TPC Benchmark C

Full Disclosure Report

SQL Anywhere 16

on

Dell PowerEdge T620

with

Microsoft Windows 2012 Standard x64

FDR Version 1.0

November 24, 2014

Special notices

SAP believes that the information in this document is accurate as of the publication date. The information in this document is subject to change without notice. SAP assumes no responsibility for any errors that may appear in this document. The pricing information in this document is believed to accurately reflect the current prices as of the publication date. However, SAP provides no warranty of the pricing information in this document.

Benchmark results are highly dependent upon workload, specific application requirements, and system design and implementation. Relative system performance will vary as a result of these and other factors. Therefore, TPC Benchmark C should not be used as a substitute for a specific customer application benchmark when critical capacity planning and/or product evaluation decisions are contemplated. All performance data contained in this report were obtained in a rigorously controlled environment. Results obtained in other operating environments may vary significantly. SAP does not warrant or represent that a user can or will achieve similar performance expressed in transactions per minute (tpmC) or normalized price/performance ($/tpmC). No warranty of system performance or price/performance is expressed or implied in this report.

SAP, Sybase, iAnywhere Solutions, and SQL Anywhere are registered trademarks of SAP, Inc. or its subsidiaries.

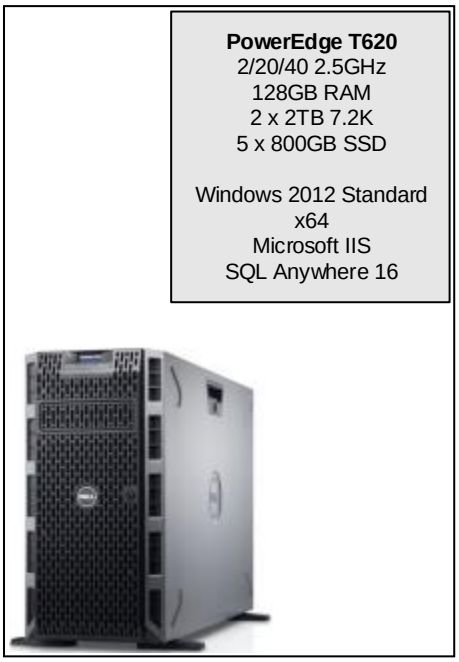
PowerEdge is a registered trademark of Dell Inc.

Microsoft and Windows 2012 are registered trademarks of Microsoft Corporation.

TPC Benchmark C is a trademark of the Transaction Processing Performance Council.

All other brand or product names mentioned herein must be considered trademarks or registered trademarks of their respective owners.

	Dell PowerEdge T620			**TPC-C Version 5.11** **TPC Pricing 1.7.0** Report Date **November 24, 2014**
Total System Cost	TPC-C Throughput		Price/Performance	Availability Date
$21,160.12 USD	**112,890 tpmC**		**$0.19 USD/tpmC**	**24-Nov-2014**
Server Processors/cores/threads	Database Manager	Operating System	Other Software	Number of Users
2/20/40 Intel Xeon E5-2670 v2 2.5GHz	**SQL Anywhere 16**	**Microsoft Windows 2012 Standard x64**	**Microsoft IIS Microsoft COM+**	**90,000**



System Components	Quantity	Description
Processors/cores/threads	2/20/40	Intel Xeon E5-2670 v2 2.5GHz 25MB L3 Cache
Memory	8	16GB RDIMM
Disk Controller	1	PERC H710
Disk Drives	2	2TB 7.2Krpm SAS 6Gbps 3.5in Hot-plug
	5	800GB SSD SATA 3Gbps 2.5in Hot-plug,3.5in HYB
Total Storage	8TB	

Dell PowerEdge T620

TPC-C Version 5.11
TPC Pricing 1.7.0

Report Date
November 24, 2014

Availability Date:
November 24, 2014

Description	Part Number	Source	Unit Price	Qty	Price	
Client/Server Hardware						
PowerEdge T620		2		1	*$15,023.72*	
PowerEdge T620 Motherboard, TPM	591-BBBN		(included)	1		
Non-Mission Critical: 4-Hour 7x24 On-site Service After Problem Diagnosis, Initial Year	989-9671		(included)	1		
Non-Mission Critical: 4-Hour 7x24 On-site Service After Problem Diagnosis, 2 Year Extended	990-0311		(included)	1		
Broadcom 5720 DP 1Gb Network Interface Card	430-4423		(included)	1		
iDRAC7 Enterprise	421-5339		(included)	1		
Chassis with up to 8, 3.5" Hard Drives, Tower Configuration	318-2059		(included)	1		
T620 PERC Cable for 3.5in Chassis	331-6124		(included)	1		
Power Saving Dell Active Power Controller	330-5116		(included)	1		
RAID 1 for H710P/H710/H310 (2 HDDs)	342-3954		(included)	1		
PERC H710 Adapter RAID Controller, 512MB NV Cache, Full Height	342-4048		(included)	1		
T620 Heat Sink, 1 Proc, up to 115W	331-5602		(included)	1		
Intel Xeon E5-2670v2 2.5GHz, 25M Cache, 8.0GT/s QPI, Turbo, HT, 10C, 115W, Max Mem 1866MHz	338-BDBG		(included)	1		
T620 Heat Sink, 1 Proc, up to 115W	331-5602		(included)	1		
Intel Xeon E5-2670v2 2.5GHz, 25M Cache, 8.0GT/s QPI, Turbo, HT, 10C, 115W, Max Mem 1866MHz,2nd Proc	338-BDBV		(included)	1		
16GB RDIMM, 1600MT/s, Low Volt, Dual Rank, x4 Data Width	319-1812		(included)	8		
1600MT/s RDIMMS	331-4424		(included)	1		
2TB 7.2K RPM Near-Line SAS 6Gbps 3.5in Hot-plug Hard Drive	342-2100		(included)	2		
DVD-ROM, SATA, Internal	313-6765		(included)	1		
Casters for PowerEdge Tower Chassis	330-4121		(included)	1		
Dual, Hot-plug, Redundant Power Supply (1+1), 750W	331-4605		(included)	1		
Power Cord, NEMA 5-15P to C13, 15 amp, wall plug, 10 feet / 3 meter	310-8509		(included)	2		
Optical Mouse, Two Buttons, USB, Black	331-0846		(included)	1		
Dell QuietKey Keyboard, No Hot Keys, English, No Palmrest, ESG	331-2254		(included)	1		
Dell 17 Monitor - E1715S	480-ACFR		(included)	1		
Windows Server 2012R2 Standard, Media, FI Standard Ed Downgrade image, Eng	634-BBOZ		(included)	1		
800GB Solid State Drive SATA Read Intensive MLC 3Gbps 2.5in Hot-plug Drive-Limited Warranty,CusKit	342-6080		(included)	5		

Dell PowerEdge T620

TPC-C Version 5.11
TPC Pricing 1.7.0

Report Date
November 24, 2014

Availability Date:
November 24, 2014

Client/Server Software						
Microsoft Visual Studio 2013		4	$499.00	1	$499.00	
Microsoft Support (1 incident)		3	$259.00	1	$259.00	
SQL Anywhere 16 (Workgroup)		1	$3,240.00	1	$3,240.00	
SQL Anywhere Support (3yr)		1	$712.80	3	$2,138.40	
Total					**$21,160.12**	

Pricing Notes

1) https://store.sap.com
2) Dell, Inc. - Quote 695609049 (see Appendix D)
3) http://support2.microsoft.com/gp/microsoft-support-options
4) http://www.microsoftstore.com/store/msusa/en_US/pdp/Visual-Studio-Professional-2013/productID.284832200

Three year cost of ownership: $21,160.12 USD

Benchmark rating: 112,890.54 tpmC

Price/Performance: $0.19 USD

Benchmark implementation and results independantly audited by Francois Raab of InfoSizing Inc. (www.sizing.com)

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 pricing 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.

	**Dell PowerEdge T620**			**TPC-C Version 5.11** **TPC Pricing 1.7.0** Report Date **November 24, 2014** Availability Date: **November 24, 2014**
MQTh, computed Maximum Qualified Throughput		**112,890.54 tpmC**		
Response Times (seconds)	**Min**	**Average**	**90th**	**Max**
New-Order	0.11	0.24	0.58	9.97
Payment	0.11	0.22	0.55	9.64
Order-Status	0.11	0.24	0.56	9.66
Delivery (interactive portion)	0.11	0.11	0.11	4.17
Delivery (deferred portion)	0	0.05	0.08	9.49
Stock-Level	0.11	0.23	0.53	9.59
Menu	0.11	0.11	0.11	4.64
Transaction Mix	**Percent**	**Number**		
New-Order	44.96%	13,546,865		
Payment	43.01%	12,959,244		
Order-Status	4.01%	1,208,187		
Delivery	4.01%	1,208,207		
Stock-Level	4.01%	1,208,223		
Keying Times (seconds)	**Min**	**Average**	**Max**	
New-Order	18	18.08	26.75	
Payment	3.01	3.06	11.75	
Order-Status	2.01	2.06	10.75	
Delivery	2.01	2.06	10.75	
Stock-Level	2.01	2.06	10.75	
Think Times (seconds)	**Min**	**Average**	**Max**	
New-Order	0.02	12.12	128.25	
Payment	0.02	12.11	127.83	
Order-Status	0.02	10.1	100.51	
Delivery	0.02	5.1	54.17	
Stock-Level	0.02	5.1	55.79	
Test Duration				
Ramp-up time				10 min
Measurement Interval				120 min
Checkpoints in Measurement Interval				4
Checkpoint interval				29.5 min
Number of transactions (all types) completed during Measurement Interval				30,130,726

Abstract

We describe a TPC-C benchmark run sponsored by SAP, run using SQL Anywhere 16 on a Dell PowerEdge T620 server with Microsoft Windows 2012 Standard x64. We describe the configuration and pricing of the benchmark system, the method used to measure its performance and verify its properties, and the performance results achieved.

With this configuration, we were able to achieve 112,890 tpmC at a price/performance of $ 0.19 USD/ tpmC. The priced system will available as of November 24, 2014.

This benchmark publication has been independently audited by François Raab of InfoSizing, Inc. (www.sizing.com).

Table of Contents

Preface

The TPC Benchmark C was developed by the Transaction Processing Performance Council (TPC). The TPC was founded to define transaction processing benchmarks and to disseminate objective, verifiable performance data to the industry. This full disclosure report is based on the TPC Benchmark C Standard Specifications Version 5.11, released February, 2010.

TPC Benchmark C Overview

The TPC describes this benchmark in Clause 0.1 of the specifications as follows:

TPC Benchmark™ C (TPC-C) is an OLTP workload. It is a mixture of read-only and update intensive transactions that simulate the activities found in complex OLTP application environments. It does so by exercising a breadth of system components associated with such environments, which are characterized by:

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

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 TPC-C results must include the tpmC rate, the associated price-per-tpmC, and the availability date of the priced configuration.

To be compliant with the optional TPC-Energy standard, the additional primary metric, expressed as watts-per-tpmC must be reported. The requirements of the TPC-Energy Specification can be found at www.tpc.org.

Although these specifications express implementation in terms of a relational data model with conventional locking scheme, the database may be implemented using any commercially available database management system (DBMS), database server, file system, or other data repository that

provides a functionally equivalent implementation. The terms "table", "row", and "column" are used in this document only as examples of logical data structures.

TPC-C uses terminology and metrics that are similar to other benchmarks, originated by the TPC or others. Such similarity in terminology does not in any way imply that TPC-C results are comparable to other benchmarks. The only benchmark results comparable to TPC-C are other TPC-C results conformant with the same revision.

Despite the fact that this benchmark offers a rich environment that emulates many OLTP applications, this benchmark does not reflect the entire range of OLTP requirements. In addition, the extent to which a customer can achieve the results reported by a vendor is highly dependent on how closely TPC-C approximates the customer application. The relative performance of systems derived from this benchmark does not necessarily hold for other workloads or environments. Extrapolations to any other environment are not recommended.

Benchmark results are highly dependent upon workload, specific application requirements, and 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.

General Items

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 F contains all source code used in the benchmark.

Benchmark Sponsor

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

This benchmark is sponsored by SAP, Inc. The benchmark kit was developed by SAP, and the benchmark was carried out at SAP research facilities in Waterloo, Ontario, Canada.

Assistance with hardware pricing was provided by Dell, Inc.

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 C highlights some of the configuration settings used in the benchmark. Appendix E contains a complete dump of all configuration settings used by benchmark components.

Configuration Diagrams

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

- *Number and type of processors*
- *Size of allocated memory, and any specific mapping/partitioning of memory unique to the test*
- *Number and type of disk units (and controllers, if applicable)*
- *Number of channels or bus connections to disk units, including the protocol type*

• Number of LAN (e.g. Ethernet) connections, including routers, work stations, terminals, etc, that were physically used in the test or are incorporated into the pricing structure (see Clause 8.1.8)

• Type and run-time execution location of software components (e.g. DBMS, client processes, transaction monitors, software drivers, etc)

The priced configuration is identical to the one tested, with the exceptions that the benchmarked configuration contains one less SSD hard drive and it used a different keyboard, mouse, and video display than the ones included in the priced configuration.

Figure 1a. Benchmarked configuration.

PowerEdge T620
2/20/40 2.5GHz
128GB RAM
2 x 2TB 7.2K
4 x 800GB SSD

Windows 2012 Standard
x64
Microsoft IIS
SQL Anywhere 16

Figure 2b. Priced configuration.

PowerEdge T620
2/20/40 2.5GHz
128GB RAM
2 x 2TB 7.2K
5 x 800GB SSD

Windows 2012 Standard
x64
Microsoft IIS
SQL Anywhere 16

Clause 1: Logical Data Base Design Related Items

Table Definitions

Listings must be provided for all table definition statements and all other statements used to setup the data base.

Appendix D contains a SQL script generated by unloading the schema of the benchmarked database. This script could be used to rebuild the database schema.

Database Organization

The physical organization of tables and indices, within the data base, must be disclosed.

The system consists of four logical volumes as shown in Table 1.

Table 1. Physical storage organization.

Contents	Location	Configuration	Windows drive letter(s)	Total size reported by Windows
2 x 2000 GB SAS	T620	RAID 1	C: (O/S and log partition)	1862.5 GB
5 x 800 GB SSD	T620	RAID 0	D:	3723.15 GB

The database consists of four files as shown in Table 2.

Table 2. Database file locations.

Name	Location	Description
tpcc.db	D:\tpcc_data\tpcc.db	System dbspace, catalog, checkpoint log
misc.dbs	D:\tpcc_data\misc.dbs	All TPCC tables, except stock and customer; and all TPCC indexes, including stock and customer
cs.dbs	D:\tpcc_data\cs.dbs	Stock and customer tables (but not their indexes)
tpcc.log	C:\tpcc_data\tpcc.log	Transaction log

One complete backup copy of the database is stored C:\backup.

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 data base 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.

The capability of the configuration to support insert and delete operations was verified by the execution of the dbcheck.sql and dbinsert.sql scripts, which checks that rows can be inserted and deleted into all tables in the manner required by Clause 1.4.11. There is no restriction on the number of insertions up to the limits defined in Clause 1.4.11.

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.

No partitioning is used in this configuration.

Replication

Replication tables, if used, must be disclosed (see Clause 1.4.6).

No replication is used in this configuration.

Table Attributes

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

No additional or duplicate attributes to any table were used in this configuration.

Clause 2: Transaction & Terminal Profiles Related Items

Verification for the Random Number Generator

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

Random numbers in this configuration are generated by the popular public-domain MT19937 algorithm. Details of this algorithm are available from http://www.math.sci.hiroshima-u.ac.jp/~m-mat/MT/VERSIONS/C-LANG/mt19937-64.c

Input/Output Screens

The actual layouts of the terminal input/output screens must be disclosed.

All input/output screens follow the requirements of the standard, as demonstrated to the auditor during the on-site portion of the audit.

Priced Terminal Features

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

The features of the emulated terminals were verified by the auditor during the on-site portion of the benchmark audit.

Presentation Managers

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

The terminal input and output screens are rendered by a web browser from plain HTML. No special presentation managers or extra functionality is used.

Transaction Statistics

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

The transaction statistics are shown in

Table 3.

Table 3. Transaction statistics.

Transaction	Function	Value
New Order	Home warehouse items	99.00%
	Remote warehouse items	1.00%
	Rolled back transactions	1.01%
	Average lines per order	10.0
Payment	Home warehouse	85.00%
	Remote warehouse	15.00%
	Lookup by name	59.98%
Order Status	Lookup by name	60.06%
Delivery	Skipped transactions	0

Queuing Mechanism of Delivery

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

The web server is started with a number of delivery worker threads; this number can be controlled by a registry setting. When a delivery request is processed by the web server, it is inserted into an array, and the next waiting delivery worker is signaled. The web server then immediately returns a message to the user informing them that the delivery has been queued. Meanwhile, the delivery worker will look up the delivery value in the array and submit it to the database server, waiting while it is processed, and then recording its completion time.

Clause 3: Transaction and System Properties

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

Atomicity Requirements

The system under test must guarantee that data base 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.

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.

This was verified automatically by the atom.py script.

Atomicity of Aborted Transactions

Perform the Payment transaction for a randomly selected warehouse, district, and customer (by customer number) a 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.

This was verified automatically by the atom.py script.

Consistency Requirements

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

Verify that the data base 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 specification defines 12 consistency conditions of which the following four are required to be explicitly demonstrated:

Consistency conditions 1 through 4 were verified automatically by first running the consist.py script against a fresh copy of the database. Then, consist.py was run against the database again after an RTE run of 6hrs and also after each of the durability tests.

Isolation Requirements

Operations of concurrent data base 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.

The isolation requirements were verified automatically by starting a clean database and running the isol.py script. Isolation tests 7 and 8 followed case A.

Durability Requirements

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

Two tests were used to verify all of these properties.

Loss of data

The system must guarantee durability after permanent irrecoverable failure of any single durable medium containing TPC-C data base tables or recovery log data, as described in Clause 3.5.3.1

The loss of a single log disk (a disk from the array containing the database catalog and the checkpoint logs) and the loss of a single data disk (a disk from the array containing the database files) was tested using the following procedure:

1. Generate a fresh copy of the database.
2. Make a backup copy of the entire database (tpcc.db, tpcc.log, cs.dbs, misc.dbs); in c:\backup.
3. Start the server with dbsvc.exe –u tpcc_dbsrv
4. Run count_orders_before.bat to count the number of orders in the database.
5. Start the RTE with the –dump txn flag and let it run for 5 minutes; verify that the reported throughput is at least 11,289 tpmC.
6. Remove one of the RAID 0 drives (D:). The system reports disk I/O failure and halts.
7. Shut down the database and web server.
8. Replace the removed physical disk with a newly formatted disk.
9. Reboot the machine.
10. Using the BIOS RAID Manager or the Dell system management utility, recreate virtual drive D: spanning the 4 SSD drives.
11. In Windows, create and format a drive for D:.

12. Copy the backed up files in C:\backup to their respective locations on D. (Do not copy the backed-up log file).

13. Rename c:\tpcc_data\tpcc.log to c:\tpcc_data\tpcc2.log

14. Run: dbsrv16 tpcc.db –a c:\tpcc\tpcc2.log

15. Start the server with dbsvc.exe –u tpcc_dbsrv

16. Run count_orders_after.bat to count the number of orders in the recovered database.

17. Run consist.py to verify consistency conditions 1 through 4.

18. On the RTE machine, run convertorders.py to generate a flat file containing all orders reported as committed to the driver.

19. Copy the result file from step 18 to the SUT and run durability.sql to ensure all orders reported as commited by the driver are present in the recovered database. The final query in the script should return zero.

Loss of Log / Loss of power / loss of memory / system failure

The system must guarantee durability after a system-wide failure, or failure of main memory, or loss of power to the system.

All of these conditions were tested simultaneously using the following procedure:

1. Generate a fresh copy of the database.

2. Start the server with dbsvc.exe –u tpcc_dbsrv

3. Start the RTE and let it run for about 5 minutes to warm up the server.

4. Stop the RTE.

5. Connect with dbisql and issue a manual checkpoint.

6. Run count_orders_before.bat to count the number of orders in the database.

7. Restart the RTE with transaction URL dumping and let it run for 5 minutes (verifying that the average throughput is above 101,600 tpmC and that at least 60,000 new order transactions have completed.

8. Pull one of the RAID 1 drives.

9. Let the RTE run for 5 more minutes

10. Kill the power to the entire SUT by turning off the powerbar to which all SUT components are connected.

11. After 10 seconds, reenable the power and restart the system.

12. Once the machine has rebooted, run start_server.bat to start the database server.
13. Run count_orders_after.bat to count the number of orders in the recovered database.
14. On the RTE machine, run convertorders.py to generate a flat file containing all orders reported as committed to the client.
15. Copy the file from step 14 to the SUT and run durability.sql to ensure all orders reported as committed by the client are present in the recovered database. The final query in the script should return zero.

Clause 4: Scaling and Data Base Population Related Items

Cardinality of Tables

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

The cardinality of each table in the database at the start of the benchmark run was as follows:

Table 4. Table cardinality at start of benchmark.

Table	Cardinality
customer	270,000,000
district	90,000
history	270,000,000
item	100,000
new_orders	81,000,000
order_line	2,699,991,176
orders	270,000,000
stock	9,000,000,000
warehouse	9,000

Distribution of Tables and Logs

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

See Table 1 and Table 2.

Data Base Model Implemented

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

A standard row-based relational database model was implemented by the SQL Anywhere DBMS.

Partitions/Replications Mapping

The mapping of data base partitions/replications must be explicitly described.

No partitions or replication was used in this benchmark configuration.

60-Day Space Calculations

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

The details of this calculation are included in Appendix A.

Clause 5: Performance Metrics and Response Time Related Items

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 5. Transaction response times.

Transaction	Average	90th Percentile	Maximum
New Order	0.24	0.58	9.97
Payment	0.22	0.55	9.64
Order Status	0.24	0.56	9.66
Delivery (interactive)	0.11	0.11	4.17
Delivery (deferred)	0.05	0.08	9.49
Stock Level	0.23	0.53	9.59
Menu	0.11	0.11	4.64

Keying and Think Times

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

Table 6. Keying times.

Type	Minimum	Average	Maximum
New Order	18.00	18.08	26.75
Payment	3.01	3.06	11.75
Order Status	2.01	2.06	10.75
Delivery (interactive)	2.01	2.06	10.75
Stock Level	2.01	2.06	10.75

Table 7. Think times.

Type	Minimum	Average	Maximum
New Order	0.02	12.12	128.25
Payment	0.02	12.11	127.83
Order Status	0.02	10.10	100.51
Delivery (interactive)	0.02	5.10	54.17
Stock Level	0.02	5.10	55.79

Response Time Frequency Distribution

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

Figure 3. New order response time.

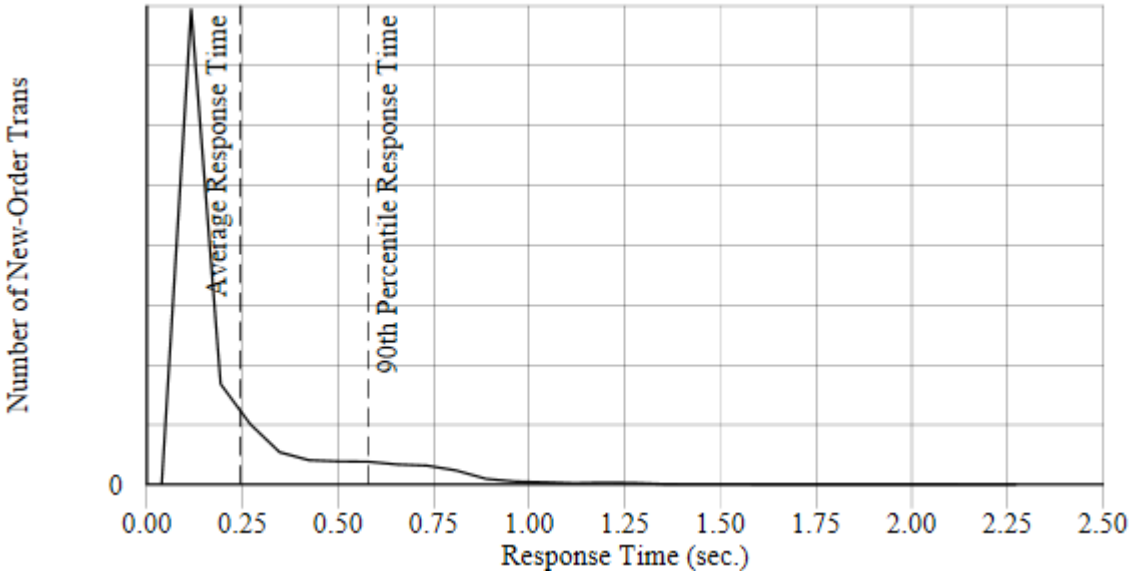


Figure 4. Payment response time.

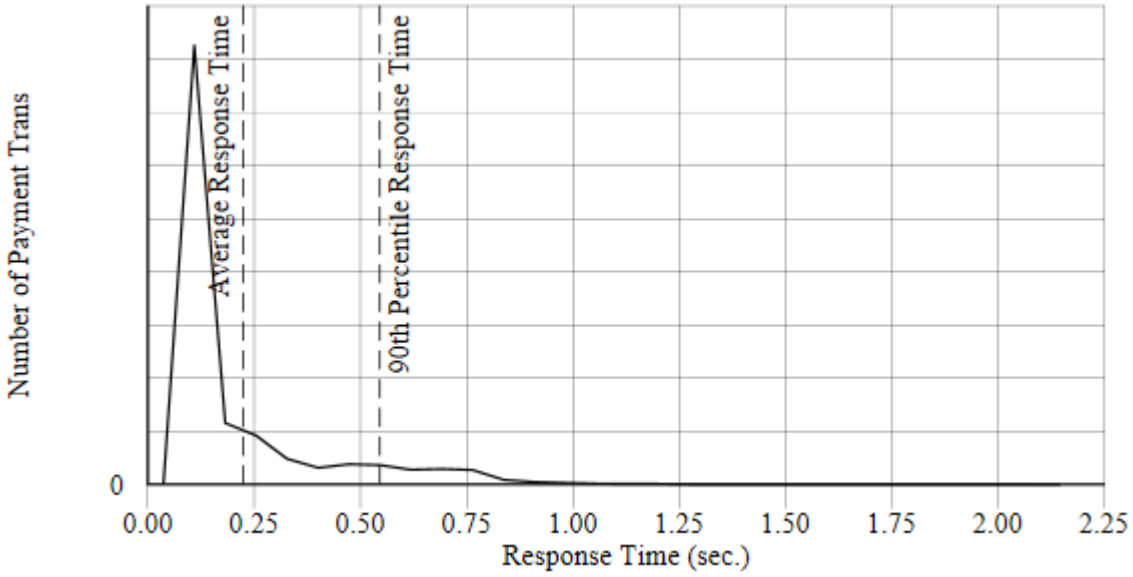


Figure 5. Order status response time.

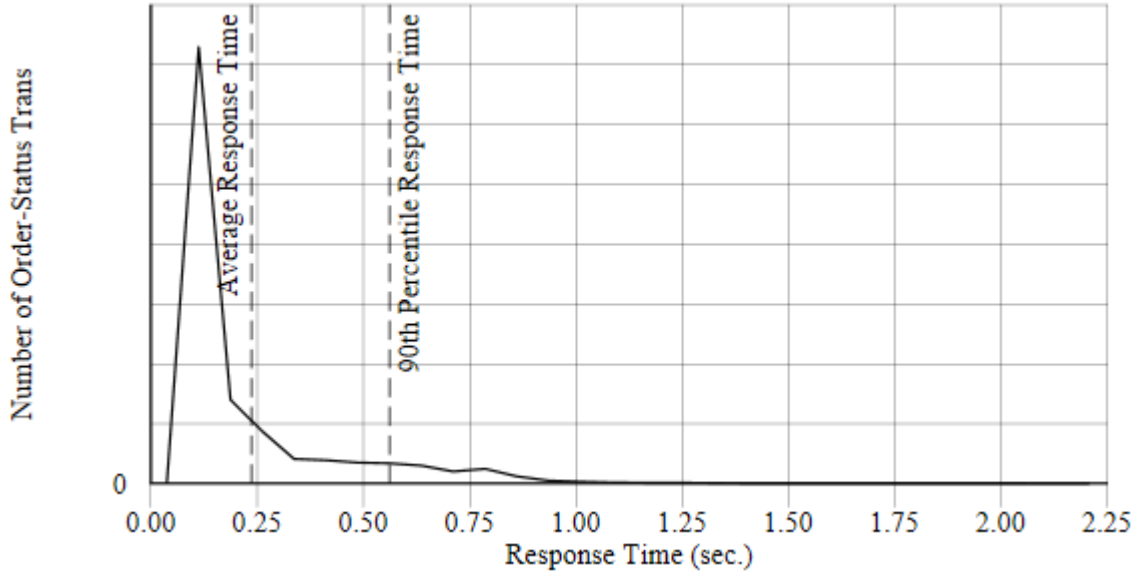


Figure 6. Interactive delivery response time.

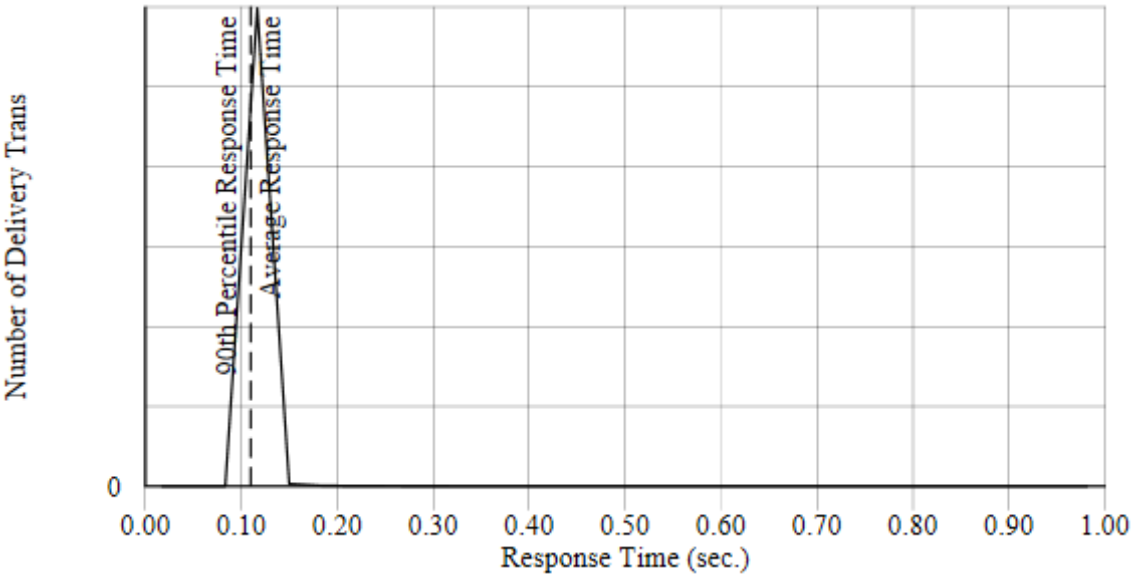


Figure 7. Deferred delivery response time.

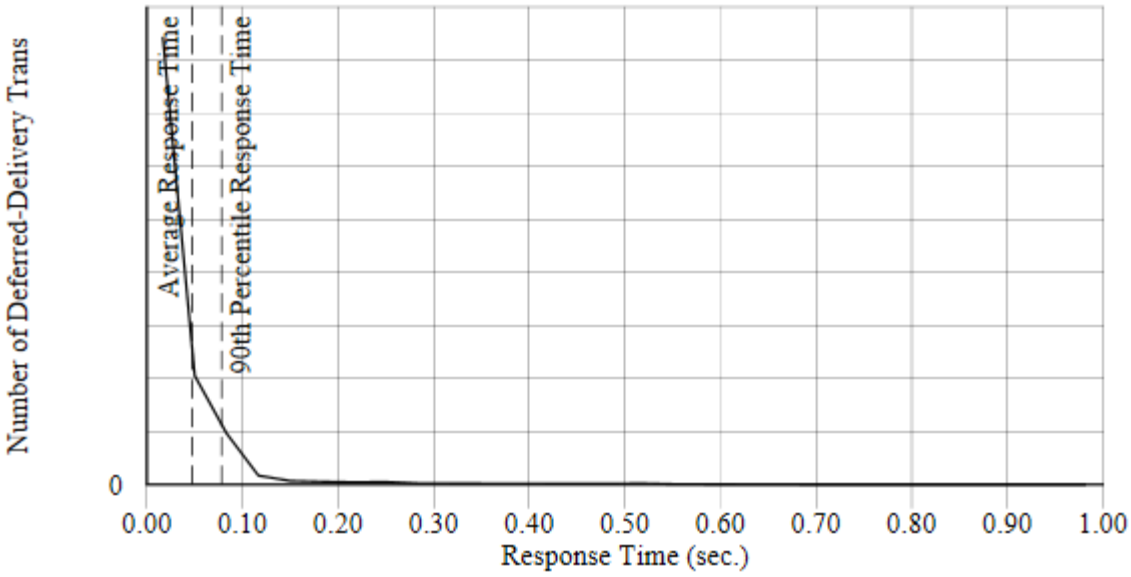
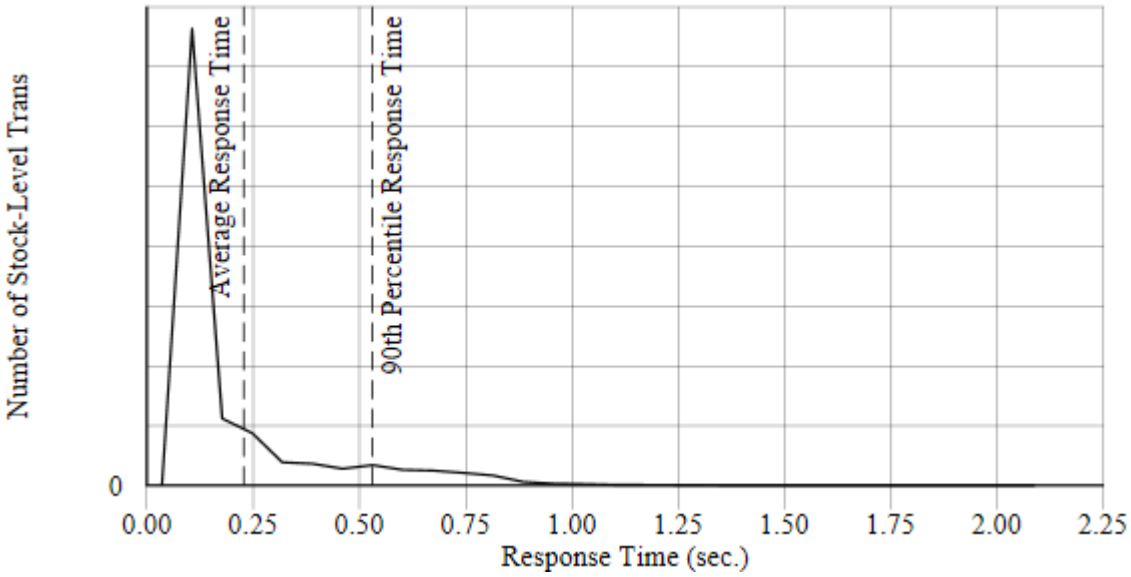


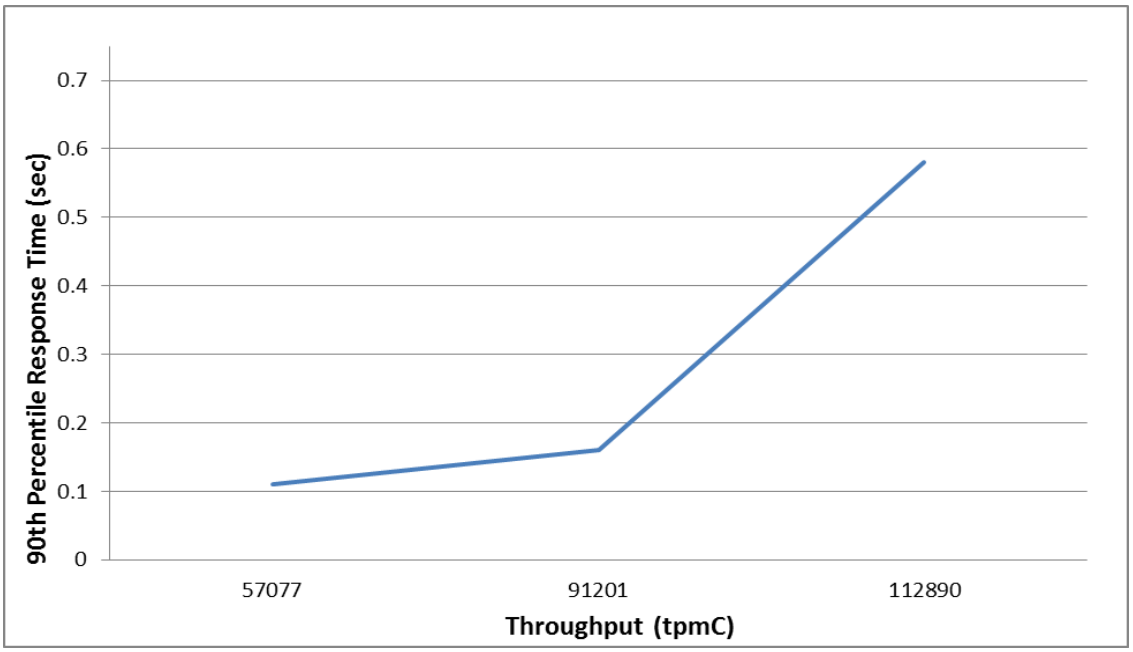
Figure 8. Stock level response time.



New Order Response Time Versus Throughput

A graph must be plotted at approximately 50%, 80%,and 100% of reported throughput rate (additional data points are optional).

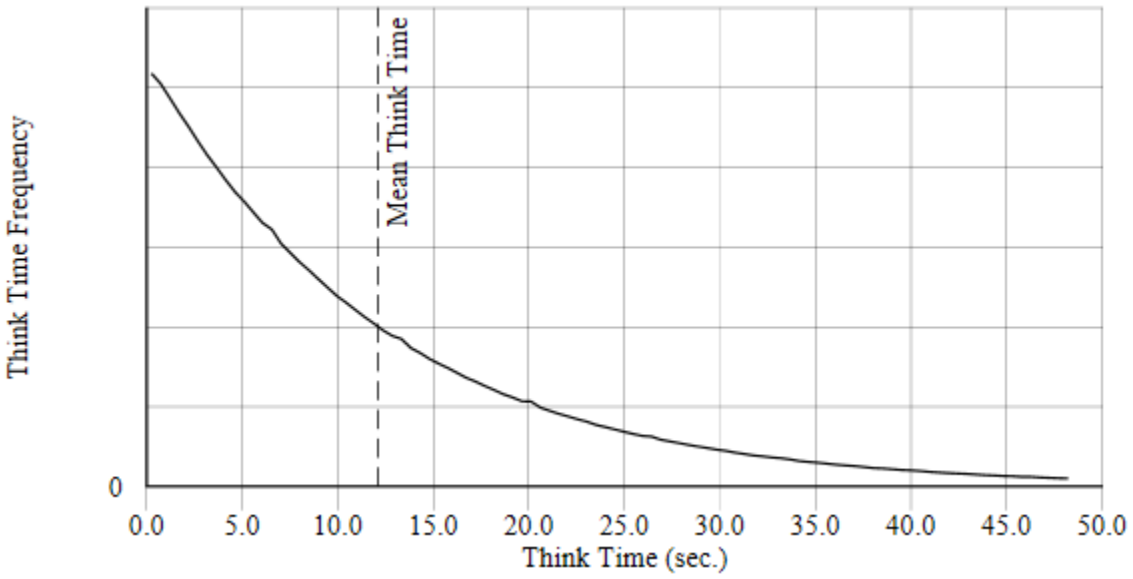
Figure 9. Response time vs. throughput.



Think Time Frequency Distribution

A graph of the think time frequency distribution must be reported for the New-Order transaction.

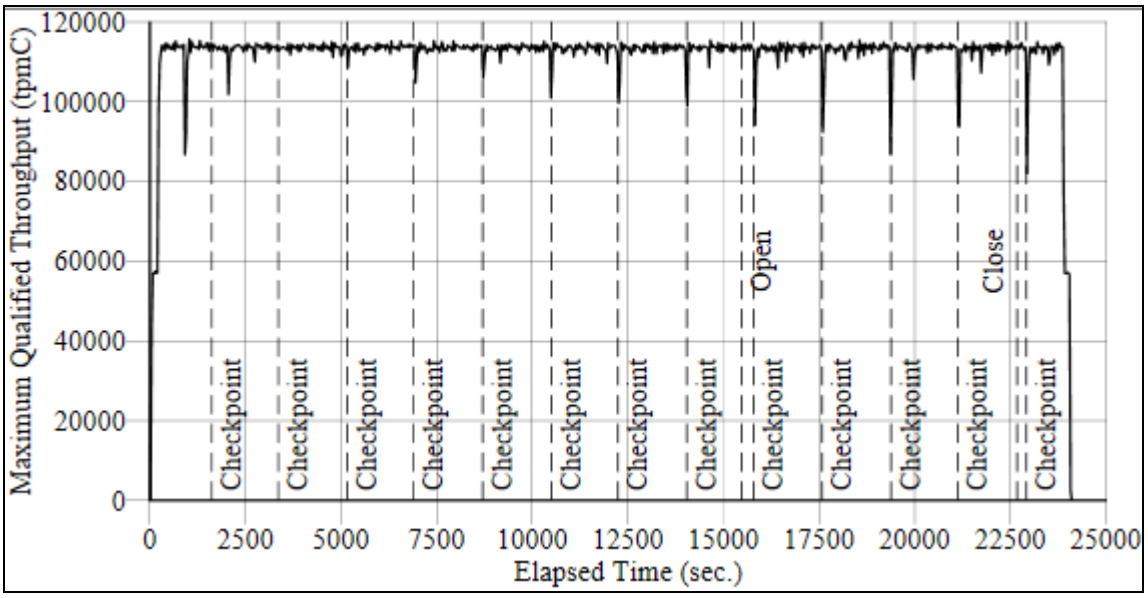
Figure 10. Think time distribution.



Throughput versus Elapsed Time

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

Figure 11. Throughput over ramp-up and measurement interval.



The markers labelled “Open” and “Closed” indicate the beginning and end of the reported measurement interval.

Steady State Determination

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

The benchmark was run against the system for a 6.5 hour period. The measurement interval was then selected as a 120-minute window over this run. It was verified that the system sustained a throughput that was at least 98% of the reported throughput over the 6.5 hour period for which the benchmark was run.

Work Performed During Steady State

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

Transaction Flow

When a user logs into the system, they are first presented with a welcome screen where they can enter their home warehouse and district id. The terminal is then assigned to that location for its active duration. Each terminal session will timeout after five minutes if it is not in use.

The user is presented with a menu listing the types of transactions. The user then picks a transaction, which causes a request to be sent from the user’s browser to the web-server. The web-server returns a

form where the user can fill in the details of the particular transaction type they have selected. This is then submitted to the web-server as a URL.

Each request arriving at the web-server is assigned to a worker thread. Delivery requests are assigned to the pool of delivery threads, while all other transaction types use a common pool of worker threads. Each worker thread parses the data submitted to it and fills in an in-memory structure with these parameters, then makes a call to the COM+ interface to perform the transaction. A COM+ object is then assigned to the thread, and the COM+ object submits the transaction to the database via ODBC. When the transaction completes, its return values (including success or failure) are filled into the appropriate data structures and passed back to the web-server worker thread. This thread then generates the HTML for the response and returns it to the user terminal.

Database Transaction

The database portion of each transaction type is implemented as a single stored procedure. These stored procedures include their own logic for commits (on success) or rollback (on failure).

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 benchmark was 120 minutes.

Transaction Mix

The method of regulation of the transaction mix (e.g., card decks or weighted random distribution) must be described. If weighted distribution is used and the RTE adjusts the weights associated with each transaction type, the maximum adjustments to the weight from the initial value must be disclosed. (8.1.6.13)

The RTE is hard-coded with the desired transaction mix. As terminals complete their think time, the RTE selects the next transaction type so that the correct proportion is maintained.

Percentage of Total Mix

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

The mix of transactions during the measurement interval are shown in Table 8.

Table 8. Transaction mix.

Transaction Type	Mix
New Order	44.96%
Payment	43.01%
Order Status	4.01%
Delivery	4.01%
Stock Level	4.01%

Checkpoints

The number of checkpoints in the Measurement Interval, the time in seconds from the start of the Measurement Interval to the first checkpoint, and the Checkpoint Interval must be disclosed.

The checkpoints performed during the measurement interval are shown below. The times given are accurate to within 1 second, as obtained from the timestamps in the database server console log.

Table 9. Checkpoint times.

Event	Start time	End time	Duration
Checkpoint #1	3:05:49	3:32:38	27 min
Checkpoint #1	3:35:26	4:01:41	26 min
Checkpoint #3	4:05:04	4:30:46	25 min
Checkpoint #4	4:34:43	4:59:51	25 min
Measurement Interval	3:00:00	5:00:00	120 min

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

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.

The RTE is a custom piece of software written for this benchmark. The source code listing of it can be found in Appendix F.

The RTE on client machine 1 was run during the benchmark with the following command line:

```
tpccrte.exe –h 192.168.5.5 –r 10000 –n 50000000 –w 9000 –t 2000 –f 1 –l 4500
```

(Connect to the web server, running at the provided ip address, displaying a performance metric every 10K transactions, running 50 million transactions for warehouses 1 to 4500 of 9000 warehouses in total using 2000 threads).

The RTE on client machine 2 was run during the benchmark with the following command line:

```
tpccrte.exe –h 192.168.5.5 –r 10000 –n 50000000 –w 9000 –t 2000 –f 4501 –l 9000
```

(Connect to the web server, running at the provided ip address, displaying a performance metric every 10K transactions, running 50 million transactions for warehouses 4501 to 9000 of 9000 warehouses in total using 2000 threads).

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.

No components were emulated in this benchmark configuration.

Network Bandwidth

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

The 2 RTEs were connected to the SUT (client and server on a single machine) by a 1Gbit switch.

Operator Intervention

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

The database server was started, and the client was then started. No operator intervention was required during the 6.5 hour period for which the benchmark was run.

Clause 7: Pricing Related Items

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.

All hardware and software components, and their associated support costs, are reported in the executive summary of this report. All third-party quotations are included in Appendix B.

Three Year Cost of System Configuration

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

All hardware and software components, and their associated support costs, are reported in the executive summary of this report. All third-party quotations are included in Appendix B.

Availability Dates

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

All hardware components, Microsoft Windows 2012 and Visual Studio 2012, and SQL Anywhere 16 are currently available at the time of report publication.

Statement of tpmC and Price/Performance

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

Maximum qualified throughput	112,890 tpmC
Total system cost	$21,691.74USD
Price per tpmC	$0.19 USD/ tpmC
Availability date	November 24, 2014

Clause 9: 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.

This benchmark has been audited by François Raab of InfoSizing, Inc. (www.sizing.com)

The letter of attestation is included on the next two pages.

InfoSizing
The Right Metric For Sizing IT

TPC Transaction Processing Performance Council
Certified Auditors

Jason Hinsperger
Senior Product Manager
SAP Canada, Inc.
445 Wes Graham Way
Waterloo, Ontario N2L6R2

November 20, 2014

I verified the TPC Benchmark C (TPC-C™ v5.11.0) performance of the following configuration:

Platform:	Dell PowerEdge T620 with SQL Anywhere 16
Operating System:	Microsoft Windows 2012 Standard x64
Database Manager:	SAP SQL Anywhere 16
Other Software:	Microsoft IIS & COM+

The results were:

Performance Metric	**112,890.58 tpmC**
Number of Users	90,000

Server	**Dell PowerEdge T620**		
CPUs	2 x Intel Xeon E5-2670 v2 Processor (2.5GHz, 25MB cache)		
Memory	128 GB		
Disks	***Qty***	***Size***	***Type***
	5	800 GB	SATA 2.5" SSD
	2	2 TB	SAS 3.5" HDD 7.2Krpm

In my opinion, these performance results were produced in compliance with the TPC requirements for the benchmark.

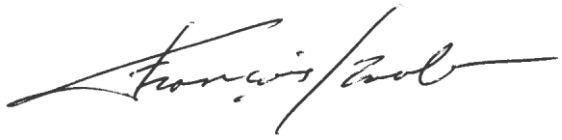
The following verification items were given special attention:

- 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 120 minutes
- Four checkpoints were executed during the measurement interval
- The 60 day storage requirement was correctly computed
- The system pricing was verified for major components and maintenance

Additional Audit Notes:

None.

Respectfully Yours,



François Raab, President

APPENDIX A: SPACE CALCULATION

Note : Numbers are in KBytes unless otherwise specified						
Warehouses	9000	**tpmC**	112,890.54	**tpmC/W**	12.54	
#wh used in test	9000					
Table	**Rows**	**Data Sizes**	**Index Size**	**5% Space**	**8H Space**	**Total Space**
'customer'	270,000,000	144,827,568	9,294,288	3,082,437		157,204,293
'district'	90,000	8,304	464	438		9,206
'history'	270,000,000	12,650,292	-		2,067,978	14,718,270
'item'	100,000	7,968	112	162		8,242
'new_order'	81,000,000	1,006,320	40,200		360,000	1,406,520
'order_line'	2,699,991,176	199,874,672	12,547,280		34,725,205	247,147,157
'orders'	270,000,000	7,746,316	6,193,040		2,278,705	16,218,061
'stock'	9,000,000,000	300,000,080	3,484,896	6,069,700		309,554,676
'warehouse'	9,000	800	16	41		857
Totals		666,122,320	31,560,296	9,152,777	39,431,888	746,267,282
Segment No.	**Segment Name**	**SizePages**	**Segment Size**	**Needed**	**Overhead**	**Not Needed**
0	'system'	6,135	24,540	60,000	600	-35,460
15	'temporary'	2,537	10,148	15,000	150	-4,852
1	'cs_space'	114,436,320	457,745,280	458,519,582	4,585,196	-774,302
2	'misc_space'	60,037,236	240,148,944	292,415,289	2,924,153	-52,266,345
Totals			697,928,912	751,009,871	7,510,099	-53,080,959
Dynamic space	220,271,280	Sum of Data for Order, Order-Line and History (excluding free extents)				
Static space	494,074,212	Data + Index + 5% Space + Overhead - Dynamic space				
Free space	36,664,379	Total Seg. Size - Dynamic Space - Static Space - Not Needed				
Daily growth	44,207,189	(Dynamic space/W * 62.5)* tpmC				
Daily spread	0	Free space - 1.5 * Daily growth (zero if negative)				
60 day (KB)	3,146,505,545	Static space + 60 (Daily growth + Daily spread)				
60 day (GB)	3,000.74	Excludes OS, Paging and RDBMS Logs				
Size of configured space for data (GB)	**3,723.15**					
Log growth	Experiment:	Number NO	44,887,237	Log size (B)	58,796,736,512	
Log (KB) per N-O txn	1.27918					
8 Hour Log (GB)	66.2					
Size of configured space for log (GB)	**1,862.5**					

APPENDIX B: THIRD PARTY PRICING

QUOTATION

Quote #: **695609049**
Customer #:
Contract #:
CustomerAgreement #:
Quote Date: **11/19/2014**

Date:
11/19/2014 **Customer Name:**

Thanks for choosing Dell! Your quote is detailed below; please review the quote for product and informational accuracy. If you find errors or desire certain changes please contact your sales professional as soon as possible.

Sales Professional Information			
SALES REP:	BILL BURR	**PHONE:**	1800 - 6958133
Email Address:	Bill_Burr@Dell.com	**Phone Ext:**	7250089

GROUP: 1 QUANTITY: 1 SYSTEM PRICE: $8,817.77 GROUP TOTAL: $8,817.77

Description	Quantity
PowerEdge T620, Intel Xeon E-26XX Processors (210-ABVO)	1
PowerEdge T620 Motherboard, TPM (591-BBBN)	1
Dell ProSupport. For tech support, visit http://support.dell.com/ProSupport or call 1-800-945-3355 (989-3439)	1
Non-Mission Critical: 4-Hour 7x24 On-site Service After Problem Diagnosis, Initial Year (989-9671)	1
Non-Mission Critical: 4-Hour 7x24 On-site Service After Problem Diagnosis, 2 Year Extended (990-0311)	1
ProSupport: 7x24 HW / SW Tech Support and Assistance, 3 Year (990-0351)	1
Dell Hardware Limited Warranty Plus On Site Service Initial Year (990-1521)	1
Dell Hardware Limited Warranty Plus On Site Service Extended Year (990-1631)	1
On-Site Installation Declined (900-9997)	1
Proactive Maintenance Service Declined (926-2979)	1
PowerEdge T620 Shipping (331-5592)	1
Broadcom 5720 DP 1Gb Network Interface Card (430-4423)	1

iDRAC7 Enterprise (421-5339)	1
Chassis with up to 8, 3.5" Hard Drives, Tower Configuration (318-2059)	1
T620 PERC Cable for 3.5in Chassis (331-6124)	1
No Bezel (313-0869)	1
Power Saving Dell Active Power Controller (330-5116)	1
RAID 1 for H710P/H710/H310 (2 HDDs) (342-3954)	1
PERC H710 Adapter RAID Controller, 512MB NV Cache, Full Height (342-4048)	1
T620 Heat Sink, 1 Proc, up to 115W (331-5602)	1
Intel Xeon E5-2670v2 2.5GHz, 25M Cache, 8.0GT/s QPI, Turbo, HT, 10C, 115W, Max Mem 1866MHz (338-BDBG)	1
T620 Heat Sink, 1 Proc, up to 115W (331-5602)	1
Intel Xeon E5-2670v2 2.5GHz, 25M Cache, 8.0GT/s QPI, Turbo, HT, 10C, 115W, Max Mem 1866MHz,2nd Proc (338-BDBV)	1
16GB RDIMM, 1600MT/s, Low Volt, Dual Rank, x4 Data Width (319-1812)	8
1600MT/s RDIMMS (331-4424)	1
Performance Optimized (331-4428)	1
2TB 7.2K RPM Near-Line SAS 6Gbps 3.5in Hot-plug Hard Drive (342-2100)	2
No System Documentation, No OpenManage DVD Kit (310-5171)	1
DVD-ROM, SATA, Internal (313-6765)	1
Casters for PowerEdge Tower Chassis (330-4121)	1
Dual, Hot-plug, Redundant Power Supply (1+1), 750W (331-4605)	1
Power Cord, NEMA 5-15P to C13, 15 amp, wall plug, 10 feet / 3 meter (310-8509)	2
Optical Mouse, Two Buttons, USB, Black (331-0846)	1
Dell QuietKey Keyboard, No Hot Keys, English, No Palmrest, ESG (331-2254)	1
Dell 17 Monitor - E1715S (480-ACFR)	1
Windows Server 2012R2 Standard Edition,Factory Installed, No Media, 2 Socket, 2 VMs,NO CALs (618-BBDS)	1

Windows Server 2012R2 Standard, Media, FI Standard Ed Downgrade image, Eng (634-BBOZ)	1
State Environmental Fee for display 15 inches, less than 35 inches (600-0277)	1

SOFTWARE & ACCESSORIES		**GROUP TOTAL: $6,209.95**	
Product	**Quantity**	**Unit Price**	**Total**
800GB Solid State Drive SATA Read Intensive MLC 3Gbps 2.5in Hot-plug Drive-Limited Warranty,CusKit (342-6080)	5	$1,241.99	$6,209.95

***Total Purchase Price:**	**$16,323.18**
Product Subtotal:	$15,023.72
Tax:	$1,200.46
Shipping & Handling:	$95.00
State Environmental Fee:	$4.00
Shipping Method:	STANDARD GROUND

(Amount denoted in $)*

Statement of Conditions

The information in this document is believed to be accurate. However, Dell assumes no responsibility for inaccuracies, errors, or omissions, and shall not be liable for direct, indirect, special, incidental, or consequential damages resulting from any such error or omission. Dell is not responsible for pricing or other errors, and reserves the right to cancel orders arising from such errors.

Dell may make changes to this proposal including changes or updates to the products and services described, including pricing, without notice or obligation.

Terms of Sale

This quote is valid for 30 days unless otherwise stated. Unless you have a separate written agreement with Dell that specifically applies to this order, your order will be subject to and governed by the following agreements, each of which are incorporated herein by reference and available in hardcopy from Dell at your request:

If this purchase is for your internal use only: Dell's Commercial Terms of Sale (www.dell.com/CTS), which incorporate Dell's U.S. Return Policy (www.dell.com/returnpolicy) and Warranty (www.dell.com/warrantyterms).

If this purchase is intended for resale: Dell's Reseller Terms of Sale (www.dell.com/resellerterms).

If this purchase includes services: in addition to the foregoing applicable terms, Dell's Service Terms (www.dell.com/servicecontracts/global).

If this purchase includes software: in addition to the foregoing applicable terms, your use of the software is subject to the license terms accompanying the software, and in the absence of such terms, then use of the Dell-branded application software is subject to the Dell End User License Agreement - Type A (www.dell.com/AEULA) and use of the Dell-branded system software is subject to the Dell End User License Agreement - Type S (www.dell.com/SEULA).

You acknowledge having read and agree to be bound by the foregoing applicable terms in their entirety. Any terms and conditions set forth in your purchase order or any other correspondence that are in addition to, inconsistent or in conflict with, the foregoing applicable online terms will be of no force or effect unless specifically agreed to in a writing signed by Dell that expressly references such terms.

Pricing, Taxes, and Additional Information

All product, pricing, and other information is valid for U.S. customers and U.S. addresses only, and is based on the latest information available and may be subject to change. Dell reserves the right to cancel quotes and orders arising from pricing or other errors. Sales tax on products shipped is based on your "Ship To" address, and for software downloads is based on your "Bill To" address. Please indicate any tax-exempt status on your PO, and fax your exemption certificate, including your Customer Number, to the Dell Tax Department at 800-433-9023. Please ensure that your tax-exemption certificate reflects the correct Dell entity name: Dell Marketing L.P. Note: All tax quoted above is an estimate; final taxes will be listed on the invoice. If you have any questions regarding tax please send an e-mail to Tax_Department@dell.com.

For certain products shipped to end-users in California, a State Environmental Fee will be applied to your invoice. Dell encourages customers to dispose of electronic equipment properly.

All information supplied to VMWARE AP for the purpose of this proposal is to be considered confidential information belonging to Dell.

About Dell

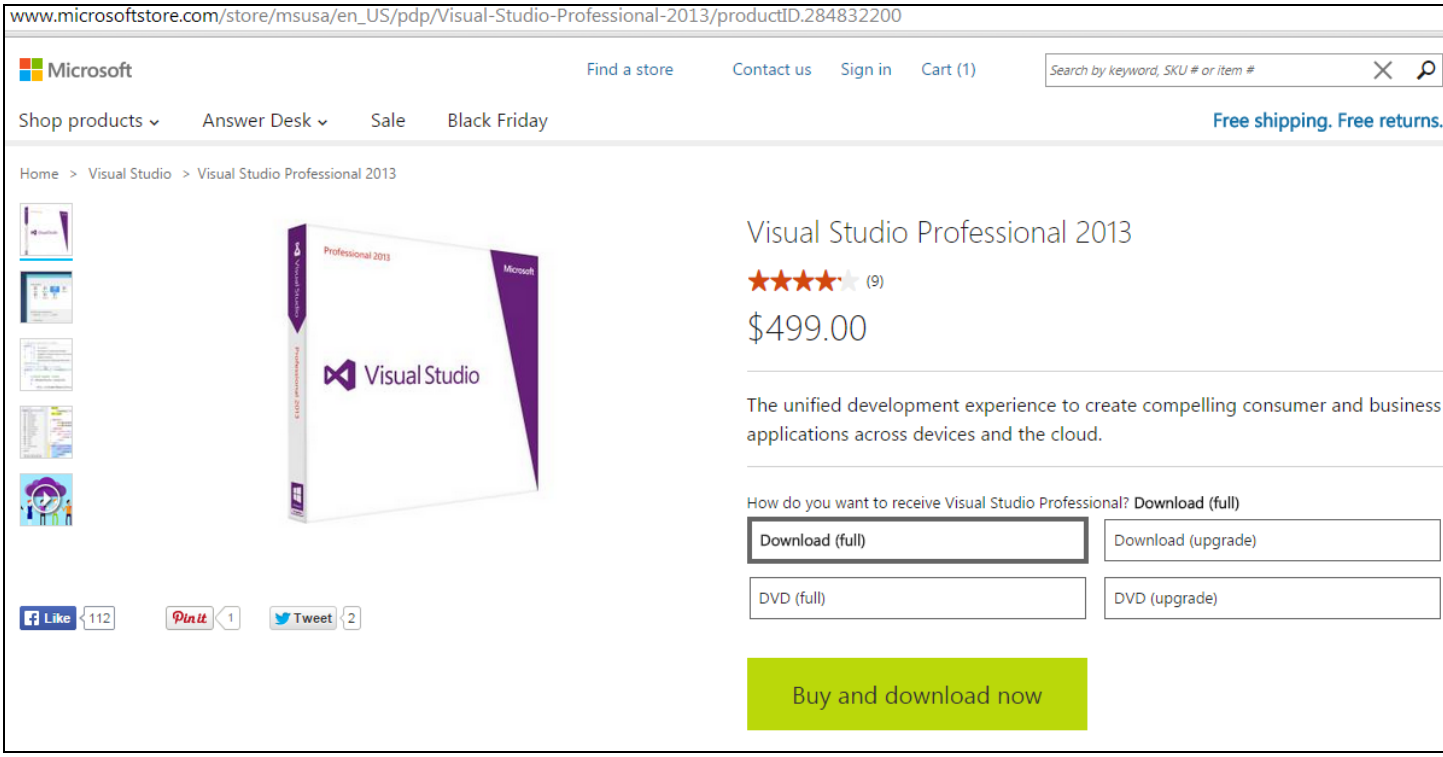
Dell Inc. listens to customers and delivers innovative technology and services they trust and value. Uniquely enabled by its direct business model, Dell is a leading global systems and services company and No. 34 on the Fortune 500. For more information, visit www.dell.com.

Privacy Policy

Dell respects your privacy. Across our business, around the world, Dell will collect, store, and use customer information only to support and enhance our relationship with your organization, for example, to process your purchase, provide service and support, and share product, service, and company news and offerings with you. Dell does not sell your personal information. For a complete statement of our Global Privacy Policy, please visit dell.com/privacy.

Microsoft Visual Studio Professional

Microsoft Visual Studio Professional is purchasable online from the Microsoft Store:



APPENDIX C: TUNING HIGHLIGHTS

This appendix highlights parameters that were tuned differently from their default values and that are believed to be the most significant to achieving the benchmark result. See Appendix E for a complete list of all configuration settings.

Database options

```
SET OPTION "PUBLIC"."max_plans_cached"='100'
go
```

Database command line

dbsrv16.exe -x tcpip -n tpcc_einstein -ca 0 -ch 115g -cl 115g -gr 9999 -gc 59 -gn 500 -gna 0 -gt 1 -o c:\tpcc_output_logs\srv.out d:\tpcc_data\tpcc.db

Operating system

Page file: 3000MB fixed size, drive C:

TPCCIsapi.dll

ISAPIWorkerThreads: 100
ISAPIDeliveryThreads: 50
ISAPIMaxDeliveries: 5000000
ISAPIDMaxTerminals: 150000

COM

Tpcccom.TPCCTran.1 component: Minimum pool size = Maximum pool size = 7000

Virtual disks:

All disks: Write Through cache policy;

APPENDIX D: DATABASE DESIGN

```
--
-- This script file reloads a database that was unloaded using "dbunload".
--
-- (Version:  16.0.0.2020)
--


-- Database file: d:\tpcc_data\tpcc.db
-- Database CHAR collation: 1252LATIN1, NCHAR collation: UCA
-- Connection Character Set: windows-1252
--
-- CREATE DATABASE command: CREATE DATABASE 'd:\\tpcc_data\\tpcc.db' LOG ON 'c:\\tpcc_data\\tpcc.log' PAGE SIZE 4096
COLLATION '1252LATIN1(CaseSensitivity=Ignore)' NCHAR COLLATION
'UCA(CaseSensitivity=Ignore;AccentSensitivity=Ignore;PunctuationSensitivity=Primary)' BLANK PADDING OFF JCONNECT OFF
CHECKSUM ON SYSTEM PROC AS DEFINER OFF
--

SET OPTION date_order          = 'YMD'
go

SET OPTION PUBLIC.preserve_source_format = 'OFF'
go

SET TEMPORARY OPTION tsql_outer_joins = 'ON'
go

SET TEMPORARY OPTION st_geometry_describe_type = 'binary'
go

SET TEMPORARY OPTION st_geometry_on_invalid = 'Ignore'
go

SET OPTION PUBLIC.reserved_keywords = ''
go


------------------------------------------------
--   Create dbspaces
------------------------------------------------

CREATE DBSPACE "cs_space" AS 'd:\\tpcc_data\\cs.dbs'
go

CREATE DBSPACE "misc_space" AS 'd:\\tpcc_data\\misc.dbs'
go

------------------------------------------------
--   Create users
------------------------------------------------

GRANT CONNECT TO "DBA" IDENTIFIED BY sql
go


------------------------------------------------
--   Create role definitions
------------------------------------------------

GRANT ROLE "SYS_AUTH_DBA_ROLE" TO "DBA" WITH ADMIN OPTION WITH NO SYSTEM PRIVILEGE INHERITANCE
go

GRANT ROLE "SYS_AUTH_DBA_ROLE" TO "dbo" WITH ADMIN OPTION WITH NO SYSTEM PRIVILEGE INHERITANCE
```

```
go

GRANT ROLE "SYS_AUTH_RESOURCE_ROLE" TO "DBA" WITH NO SYSTEM PRIVILEGE INHERITANCE
go

GRANT ROLE "SYS_AUTH_RESOURCE_ROLE" TO "dbo" WITH NO SYSTEM PRIVILEGE INHERITANCE
go

------------------------------------------------
--   Create dbspace permissions
------------------------------------------------

begin
   for dbspaces as dbcurs cursor for
         select privilege_type, dbspace_name, user_name
                  from SYS.SYSDBSPACEPERM p
                  join SYS.SYSDBSPACE d on p.dbspace_id = d.dbspace_id
                  join SYS.SYSUSER u on u.user_id = p.grantee
   do
         execute immediate 'revoke ' + if privilege_type = 1 then 'CREATE' else 'UNKNOWN' endif + ' on "' + dbspace_name + '"
from "' + user_name + '"'
   end for;
end

go

grant CREATE on "system" to "PUBLIC"
go

grant CREATE on "temporary" to "PUBLIC"
go

grant CREATE on "cs_space" to "PUBLIC"
go

grant CREATE on "misc_space" to "PUBLIC"
go


------------------------------------------------
--   Create tables
------------------------------------------------

CREATE TABLE "DBA"."warehouse" (
    "w_id"                      smallint NOT NULL
   ,"w_name"                       char(10) NOT NULL
   ,"w_street_1"                   char(20) NOT NULL
   ,"w_street_2"                   char(20) NOT NULL
   ,"w_city"                      char(20) NOT NULL
   ,"w_state"                      char(2) NOT NULL
   ,"w_zip"                       char(9) NOT NULL
   ,"w_tax"                       numeric(4,4) NOT NULL
   ,"w_ytd"                       numeric(12,2) NOT NULL
) IN "misc_space"
go

CREATE TABLE "DBA"."district" (
    "d_id"                      tinyint NOT NULL
   ,"d_w_id"                      smallint NOT NULL
   ,"d_name"                       char(10) NOT NULL
   ,"d_street_1"                   char(20) NOT NULL
   ,"d_street_2"                   char(20) NOT NULL
   ,"d_city"                      char(20) NOT NULL
   ,"d_state"                      char(2) NOT NULL
   ,"d_zip"                       char(9) NOT NULL
   ,"d_tax"                       numeric(4,4) NOT NULL
   ,"d_ytd"                       numeric(12,2) NOT NULL
```

```
   ,"d_next_o_id"                integer NOT NULL
) IN "misc_space"
go

CREATE TABLE "DBA"."customer" (
    "c_id"                      integer NOT NULL
   ,"c_d_id"                     tinyint NOT NULL
   ,"c_w_id"                     smallint NOT NULL
   ,"c_first"                   char(16) NOT NULL
   ,"c_middle"                    char(2) NOT NULL
   ,"c_last"                    char(16) NOT NULL
   ,"c_street_1"                  char(20) NOT NULL
   ,"c_street_2"                  char(20) NOT NULL
   ,"c_city"                    char(20) NOT NULL
   ,"c_state"                    char(2) NOT NULL
   ,"c_zip"                     char(9) NOT NULL
   ,"c_phone"                     char(16) NOT NULL
   ,"c_since"                   "datetime" NOT NULL
   ,"c_credit"                   char(2) NOT NULL
   ,"c_credit_lim"                numeric(12,2) NOT NULL
   ,"c_discount"                  numeric(4,4) NOT NULL
   ,"c_balance"                   numeric(12,2) NOT NULL
   ,"c_ytd_payment"                 numeric(12,2) NOT NULL
   ,"c_payment_cnt"                 smallint NOT NULL
   ,"c_delivery_cnt"               smallint NOT NULL
   ,"c_data"                    char(500) NOT NULL NO INDEX INLINE 0 PREFIX 0 COMPRESSED
) IN "cs_space"
go

CREATE TABLE "DBA"."history" (
    "h_c_id"                    integer NOT NULL
   ,"h_c_d_id"                   tinyint NOT NULL
   ,"h_c_w_id"                   smallint NOT NULL
   ,"h_d_id"                    tinyint NOT NULL
   ,"h_w_id"                    smallint NOT NULL
   ,"h_date"                    "datetime" NOT NULL
   ,"h_amount"                    numeric(6,2) NOT NULL
   ,"h_data"                    char(24) NOT NULL
) IN "misc_space"
go

CREATE TABLE "DBA"."new_order" (
    "no_o_id"                   integer NOT NULL
   ,"no_d_id"                   tinyint NOT NULL
   ,"no_w_id"                   smallint NOT NULL
) IN "misc_space"
go

CREATE TABLE "DBA"."orders" (
    "o_id"                      integer NOT NULL
   ,"o_d_id"                     tinyint NOT NULL
   ,"o_w_id"                     smallint NOT NULL
   ,"o_c_id"                    integer NOT NULL
   ,"o_entry_d"                   "datetime" NOT NULL
   ,"o_carrier_id"                tinyint NULL
   ,"o_ol_cnt"                   tinyint NOT NULL
   ,"o_all_local"                tinyint NOT NULL
) IN "misc_space"
go

CREATE TABLE "DBA"."order_line" (
    "ol_o_id"                   integer NOT NULL
   ,"ol_d_id"                   tinyint NOT NULL
   ,"ol_w_id"                    smallint NOT NULL
   ,"ol_number"                    tinyint NOT NULL
   ,"ol_i_id"                   integer NOT NULL
   ,"ol_supply_w_id"                smallint NOT NULL
```

```
   ,"ol_delivery_d"                "datetime" NULL
   ,"ol_quantity"                  smallint NOT NULL
   ,"ol_amount"                     numeric(6,2) NOT NULL
   ,"ol_dist_info"                 char(24) NOT NULL INLINE 0 PREFIX 0
) IN "misc_space"
go

CREATE TABLE "DBA"."item" (
    "i_id"                        integer NOT NULL
   ,"i_im_id"                      integer NOT NULL
   ,"i_name"                       char(24) NOT NULL
   ,"i_price"                     numeric(5,2) NOT NULL
   ,"i_data"                       char(50) NOT NULL
) IN "misc_space"
go

CREATE TABLE "DBA"."stock" (
    "s_i_id"                      integer NOT NULL
   ,"s_w_id"                       smallint NOT NULL
   ,"s_quantity"                    smallint NOT NULL
   ,"s_dist_01"                    char(24) NOT NULL
   ,"s_dist_02"                    char(24) NOT NULL
   ,"s_dist_03"                    char(24) NOT NULL
   ,"s_dist_04"                    char(24) NOT NULL
   ,"s_dist_05"                    char(24) NOT NULL
   ,"s_dist_06"                    char(24) NOT NULL
   ,"s_dist_07"                    char(24) NOT NULL
   ,"s_dist_08"                    char(24) NOT NULL
   ,"s_dist_09"                    char(24) NOT NULL
   ,"s_dist_10"                    char(24) NOT NULL
   ,"s_ytd"                       integer NOT NULL
   ,"s_order_cnt"                   smallint NOT NULL
   ,"s_remote_cnt"                   smallint NOT NULL
   ,"s_data"                       char(50) NOT NULL
) IN "cs_space"
go

CREATE GLOBAL TEMPORARY TABLE "DBA"."neworder_results" (
    "li_no"                       smallint NOT NULL
   ,"i_name"                       char(24) NOT NULL
   ,"s_quantity"                    smallint NOT NULL
   ,"b_g"                         char(1) NOT NULL
   ,"price"                       numeric(5,2) NOT NULL
   ,"amount"                        numeric(10,2) NOT NULL
) NOT TRANSACTIONAL
go

commit work
go


------------------------------------------------
--   Create indexes
------------------------------------------------

call sa_unload_display_table_status( 17738, 1, 13, 'DBA', 'warehouse' )
go

CREATE UNIQUE CLUSTERED INDEX "warehouse_c1" ON "DBA"."warehouse"
    ( "w_id" )
 IN "misc_space"
go

call sa_unload_display_table_status( 17738, 2, 13, 'DBA', 'district' )
go

CREATE UNIQUE CLUSTERED INDEX "district_c1" ON "DBA"."district"
```

```
    ( "d_w_id","d_id" )
 IN "misc_space"
go

call sa_unload_display_table_status( 17738, 3, 13, 'DBA', 'customer' )
go

CREATE UNIQUE CLUSTERED INDEX "customer_c1" ON "DBA"."customer"
    ( "c_w_id","c_d_id","c_id" )
 IN "cs_space"
go

CREATE UNIQUE INDEX "customer_nc1" ON "DBA"."customer"
    ( "c_w_id","c_d_id","c_last","c_first","c_id" )
 IN "cs_space"
go

call sa_unload_display_table_status( 17738, 4, 13, 'DBA', 'history' )
go

call sa_unload_display_table_status( 17738, 5, 13, 'DBA', 'new_order' )
go

CREATE UNIQUE CLUSTERED INDEX "new_order_c1" ON "DBA"."new_order"
    ( "no_w_id","no_d_id","no_o_id" )
 IN "misc_space"
go

call sa_unload_display_table_status( 17738, 6, 13, 'DBA', 'orders' )
go

CREATE UNIQUE CLUSTERED INDEX "orders_c1" ON "DBA"."orders"
    ( "o_w_id" DESC,"o_d_id" DESC,"o_id" DESC )
 IN "misc_space"
go

call sa_unload_display_table_status( 17738, 7, 13, 'DBA', 'order_line' )
go

CREATE UNIQUE CLUSTERED INDEX "order_line_c1" ON "DBA"."order_line"
    ( "ol_w_id","ol_d_id","ol_o_id","ol_number" )
 IN "misc_space"
go

call sa_unload_display_table_status( 17738, 8, 13, 'DBA', 'item' )
go

CREATE UNIQUE CLUSTERED INDEX "item_c1" ON "DBA"."item"
    ( "i_id" )
 IN "misc_space"
go

call sa_unload_display_table_status( 17738, 9, 13, 'DBA', 'stock' )
go

CREATE UNIQUE CLUSTERED INDEX "stock_c1" ON "DBA"."stock"
    ( "s_w_id","s_i_id" )
 IN "cs_space"
go

call sa_unload_display_table_status( 17738, 10, 13, 'DBA', 'neworder_results' )
go

call sa_unload_display_table_status( 17738, 11, 13, 'DBA', 'completion' )
go

call sa_unload_display_table_status( 17738, 12, 13, 'DBA', 'firstline' )
```

```
go

call sa_unload_display_table_status( 17738, 13, 13, 'DBA', 'temp_w' )
go

commit work
go


------------------------------------------------
--   Create procedures
------------------------------------------------

commit
go


create procedure "DBA"."tpcc_delivery"(
  in @w_id smallint,
  in @o_carrier_id smallint )
begin
  declare @d_id tinyint;
  declare @o_id integer;
  declare @c_id integer;
  declare @oid1 integer;
  declare @oid2 integer;
  declare @oid3 integer;
  declare @oid4 integer;
  declare @oid5 integer;
  declare @oid6 integer;
  declare @oid7 integer;
  declare @oid8 integer;
  declare @oid9 integer;
  declare @oid10 integer;
  declare @ol_total numeric(12,2);
  set @d_id = 0;
  while(@d_id < 10) loop
    begin
      declare "neword" dynamic scroll cursor for
        select top 1
          "no_o_id"
          from "new_order" with(holdlock)
          where "no_w_id" = @w_id and "no_d_id" = @d_id
          order by "no_w_id" asc,"no_d_id" asc,"no_o_id" asc;
      set @d_id = @d_id+1;
      set @ol_total = 0;
      set @o_id = 0;
      /*ACID_START*/
      open "neword";
      fetch next "neword" into @o_id for update;
      close "neword";
      /*ACID_STOP*/
      if(@o_id <> 0) then
        delete from "new_order"
          where "no_w_id" = @w_id and "no_d_id" = @d_id and "no_o_id" = @o_id;
        update "orders"
          set "o_carrier_id" = @o_carrier_id,
          @c_id = "o_c_id"
          where "o_w_id" = @w_id and "o_d_id" = @d_id and "o_id" = @o_id;
        update "order_line"
          set "ol_delivery_d" = "getdate"(),
          @ol_total = @ol_total+"ol_amount"
          where "ol_w_id" = @w_id and "ol_d_id" = @d_id and "ol_o_id" = @o_id;
        update "customer"
          set "c_balance" = "c_balance"+@ol_total,
          "c_delivery_cnt" = "c_delivery_cnt"+1
          where "c_w_id" = @w_id and "c_d_id" = @d_id and "c_id" = @c_id
```

```
      end if;
      /*ACID56*/
      select case @d_id when 1 then @o_id else @oid1 end,
        case @d_id when 2 then @o_id else @oid2 end,
        case @d_id when 3 then @o_id else @oid3 end,
        case @d_id when 4 then @o_id else @oid4 end,
        case @d_id when 5 then @o_id else @oid5 end,
        case @d_id when 6 then @o_id else @oid6 end,
        case @d_id when 7 then @o_id else @oid7 end,
        case @d_id when 8 then @o_id else @oid8 end,
        case @d_id when 9 then @o_id else @oid9 end,
        case @d_id when 10 then @o_id else @oid10 end
        into @oid1,@oid2,@oid3,@oid4,@oid5,@oid6,@oid7,@oid8,@oid9,@oid10
    end
  end loop;
  commit work; /*ACID_DELAY*/
  select @oid1,
    @oid2,
    @oid3,
    @oid4,
    @oid5,
    @oid6,
    @oid7,
    @oid8,
    @oid9,
    @oid10
end
go

COMMENT TO PRESERVE FORMAT ON PROCEDURE "DBA"."tpcc_delivery" IS
{create procedure tpcc_delivery(
    in @w_id                smallint,
    in @o_carrier_id  smallint )
begin
declare @d_id               tinyint;
declare @o_id               int;
declare @c_id               int;
declare @oid1               int;
declare @oid2               int;
declare @oid3               int;
declare @oid4               int;
declare @oid5               int;
declare @oid6               int;
declare @oid7               int;
declare @oid8               int;
declare @oid9               int;
declare @oid10              int;
declare @ol_total           numeric(12,2);

    set @d_id = 0;

    while (@d_id < 10) loop
            begin

            declare neword cursor for
            select top 1
                no_o_id
            from new_order with (holdlock)
            where no_w_id = @w_id and no_d_id = @d_id
            order by no_w_id, no_d_id, no_o_id asc;

            set @d_id = @d_id + 1;
            set @ol_total = 0;
            set @o_id = 0;
            /*ACID_START*/
            open neword;
            fetch neword into @o_id for update;
```

```
            close neword;
            /*ACID_STOP*/

            if (@o_id <> 0) then
               delete new_order
                     where no_w_id = @w_id and no_d_id = @d_id and no_o_id = @o_id;

               update orders
               set o_carrier_id = @o_carrier_id,
                     @c_id = o_c_id
               where o_w_id = @w_id and o_d_id = @d_id and o_id = @o_id;

               update order_line
               set ol_delivery_d = getdate(),
                     @ol_total = @ol_total + ol_amount
               where ol_w_id = @w_id and ol_d_id = @d_id and ol_o_id = @o_id;

               update customer
               set c_balance = c_balance + @ol_total,
                     c_delivery_cnt = c_delivery_cnt + 1
               where c_w_id = @w_id and c_d_id = @d_id and c_id = @c_id;
            end if;
            /*ACID56*/

            select
               case @d_id when 1  then @o_id else @oid1  end,
               case @d_id when 2  then @o_id else @oid2  end,
               case @d_id when 3  then @o_id else @oid3  end,
               case @d_id when 4  then @o_id else @oid4  end,
               case @d_id when 5  then @o_id else @oid5  end,
               case @d_id when 6  then @o_id else @oid6  end,
               case @d_id when 7  then @o_id else @oid7  end,
               case @d_id when 8  then @o_id else @oid8  end,
               case @d_id when 9  then @o_id else @oid9  end,
               case @d_id when 10 then @o_id else @oid10 end
               into @oid1,@oid2,@oid3,@oid4,@oid5,@oid6,@oid7,@oid8,@oid9,@oid10;
      end;
      end loop;

      /*ACID_DELAY*/commit;

      select
            @oid1,
            @oid2,
            @oid3,
            @oid4,
            @oid5,
            @oid6,
            @oid7,
            @oid8,
            @oid9,
            @oid10;
end
}
go

create procedure "DBA"."tpcc_maximum"()
begin
  declare @max_w_id smallint;
  select "max"("w_id")
    into @max_w_id
    from "warehouse";
  rollback work;
  select @max_w_id
end
go
```

```
COMMENT TO PRESERVE FORMAT ON PROCEDURE "DBA"."tpcc_maximum" IS
{create procedure DBA.tpcc_maximum()
begin
  declare @max_w_id smallint;
  select max(w_id) into @max_w_id
    from warehouse;
  rollback work;
  select @max_w_id
end
}
go

create procedure "DBA"."tpcc_orderstatus"(
  in @w_id smallint,
  in @d_id tinyint,
  in @c_id integer,
  in @c_last char(16) default '' )
begin
  declare @cnt smallint;
  declare @c_first char(16);
  declare @c_middle char(2);
  declare @o_entry_d "datetime";
  declare @o_carrier_id smallint;
  declare @c_balance numeric(12,2);
  declare @o_id integer;
  if(@c_id = 0) then
    begin
      declare "c" dynamic scroll cursor for select "c_id","c_balance","c_first","c_last","c_middle"
          from "customer" with(repeatableread)
          where "c_last" = @c_last and "c_w_id" = @w_id and "c_d_id" = @d_id
          order by "c_w_id" asc,"c_d_id" asc,"c_last" asc,"c_first" asc;
      select("count"()+1)/2
        into @cnt
        from "customer" with(repeatableread)
        where "c_last" = @c_last and "c_w_id" = @w_id and "c_d_id" = @d_id;
      open "c";
      fetch absolute @cnt "c"
        into @c_id,@c_balance,@c_first,@c_last,@c_middle;
      close "c"
    end
  else
    select "c_balance",
      "c_first",
      "c_middle",
      "c_last" into @c_balance,@c_first,@c_middle,@c_last
      from "customer" with(repeatableread)
      where "c_id" = @c_id and "c_d_id" = @d_id and "c_w_id" = @w_id;
    set @cnt = @@rowcount
  end if;
  if(@cnt = 0) then
    rollback work;
    raiserror 23000 'Customer not found';
    return
  end if;
  /*ACID_START*/
  select top 1
    "o_id",
    "o_entry_d",
    "o_carrier_id" into @o_id,@o_entry_d,@o_carrier_id
    from "orders" with(holdlock)
    where "o_c_id" = @c_id and "o_d_id" = @d_id and "o_w_id" = @w_id
    order by "o_w_id" desc,"o_d_id" desc,"o_id" desc;
  /*ACID_STOP*/
  /*ACID1*/
  select "ol_supply_w_id",
    "ol_i_id",
    "ol_quantity",
```

```
    "ol_amount",
    "ol_delivery_d"
    from "order_line" with(repeatableread)
    where "ol_o_id" = @o_id and "ol_d_id" = @d_id and "ol_w_id" = @w_id;
  commit work; //only local variables are returned beyond this point - can release locks
  select @c_id,
    @c_last,
    @c_first,
    @c_middle,
    @o_entry_d,
    @o_carrier_id,
    @c_balance,
    @o_id
end
go

COMMENT TO PRESERVE FORMAT ON PROCEDURE "DBA"."tpcc_orderstatus" IS
{create procedure DBA.tpcc_orderstatus(
  in @w_id smallint,
  in @d_id tinyint,
  in @c_id integer,
  in @c_last char(16) default '' )
begin
  declare @cnt smallint;
  declare @c_first char(16);
  declare @c_middle char(2);
  declare @o_entry_d datetime;
  declare @o_carrier_id smallint;
  declare @c_balance numeric(12,2);
  declare @o_id integer;

  if(@c_id = 0) then
    begin
      declare c dynamic scroll cursor for select c_id,c_balance,c_first,c_last,c_middle
          from customer with(repeatableread)
          where c_last = @c_last and c_w_id = @w_id and c_d_id = @d_id
          order by c_w_id asc,c_d_id asc,c_last asc,c_first asc;

    select(count(*)+1)/2 into @cnt
      from customer with(repeatableread)
      where c_last = @c_last and c_w_id = @w_id and c_d_id = @d_id;

    open c;
    fetch absolute @cnt c
      into @c_id,@c_balance,@c_first,@c_last,@c_middle;
    close c
    end;

  else
    select c_balance,
      c_first,
      c_middle,
      c_last into @c_balance,@c_first,@c_middle,@c_last
      from customer with(repeatableread)
      where c_id = @c_id and c_d_id = @d_id and c_w_id = @w_id;
    set @cnt = @@rowcount
  end if;

  if(@cnt = 0) then
    rollback;
    raiserror 23000 'Customer not found';
    return
  end if;

  /*ACID_START*/
  select top 1
    o_id,
```

```
    o_entry_d,
    o_carrier_id into @o_id,@o_entry_d,@o_carrier_id
    from orders with(holdlock)
    where o_c_id = @c_id and o_d_id = @d_id and o_w_id = @w_id
    order by o_w_id desc,o_d_id desc,o_id desc;
  /*ACID_STOP*/

  select
  /*ACID1*/
    ol_supply_w_id,
    ol_i_id,
    ol_quantity,
    ol_amount,
    ol_delivery_d
    from order_line with(repeatableread)
    where ol_o_id = @o_id and ol_d_id = @d_id and ol_w_id = @w_id;

  commit; //only local variables are returned beyond this point - can release locks

  select @c_id,
    @c_last,
    @c_first,
    @c_middle,
    @o_entry_d,
    @o_carrier_id,
    @c_balance,
    @o_id;

end
}
go

create procedure "DBA"."tpcc_payment"(
  in @w_id smallint,
  in @c_w_id smallint,
  in @h_amount numeric(6,2),
  in @d_id tinyint,
  in @c_d_id tinyint,
  in @c_id integer,
  in @c_last char(16) default '' )
begin
  declare @w_name char(10);
  declare @d_name char(10);
  declare @data char(500);
  declare @c_data char(500);
  declare @w_ytd numeric(12,2);
  declare @d_ytd numeric(12,2);
  declare @cnt smallint;
  declare @val smallint;
  declare @d_id_local tinyint;
  declare @w_id_local smallint;
  declare @c_id_local integer;
  declare @datetime "datetime";
  declare @w_street_1 char(20);
  declare @w_street_2 char(20);
  declare @w_city char(20);
  declare @w_state char(2);
  declare @w_zip char(9);
  declare @d_street_1 char(20);
  declare @d_street_2 char(20);
  declare @d_city char(20);
  declare @d_state char(2);
  declare @d_zip char(9);
  declare @c_first char(16);
  declare @c_middle char(2);
  declare @c_street_1 char(20);
  declare @c_street_2 char(20);
```

```
declare @c_city char(20);
declare @c_state char(2);
declare @c_zip char(9);
declare @c_phone char(16);
declare @c_since "datetime";
declare @c_credit char(2);
declare @c_credit_lim numeric(12,2);
declare @c_discount numeric(4,4);
declare @c_balance numeric(12,2);
declare @screen_data char(200);
set @screen_data = '';
set @datetime = current timestamp;
if(@c_id = 0) then
  begin
    declare "c" dynamic scroll cursor for select "c_id"
        from "customer" with(repeatableread)
        where "c_last" = @c_last and "c_w_id" = @c_w_id and "c_d_id" = @c_d_id
        order by "c_w_id" asc,"c_d_id" asc,"c_last" asc,"c_first" asc for update;
    select "count"()
      into @cnt
      from "customer" with(repeatableread)
      where "c_last" = @c_last and "c_w_id" = @c_w_id and "c_d_id" = @c_d_id;
    set @val = (@cnt+1)/2;
    open "c";
    fetch absolute @val "c"
      into @c_id;
    update "customer"
      set @c_balance = "c_balance"-@h_amount,
      "c_balance" = "c_balance"-@h_amount,
      "c_payment_cnt" = "c_payment_cnt"+1,
      "c_ytd_payment" = "c_ytd_payment"+@h_amount,
      @c_first = "c_first",
      @c_middle = "c_middle",
      @c_last = "c_last",
      @c_street_1 = "c_street_1",
      @c_street_2 = "c_street_2",
      @c_city = "c_city",
      @c_state = "c_state",
      @c_zip = "c_zip",
      @c_phone = "c_phone",
      @c_credit = "c_credit",
      @c_credit_lim = "c_credit_lim",
      @c_discount = "c_discount",
      @c_since = "c_since",
      @c_id_local = "c_id" where current of "c";
    close "c"
  end
else
  update "customer"
    set @c_balance = "c_balance"-@h_amount,
    "c_balance" = "c_balance"-@h_amount,
    "c_payment_cnt" = "c_payment_cnt"+1,
    "c_ytd_payment" = "c_ytd_payment"+@h_amount,
    @c_first = "c_first",
    @c_middle = "c_middle",
    @c_last = "c_last",
    @c_street_1 = "c_street_1",
    @c_street_2 = "c_street_2",
    @c_city = "c_city",
    @c_state = "c_state",
    @c_zip = "c_zip",
    @c_phone = "c_phone",
    @c_credit = "c_credit",
    @c_credit_lim = "c_credit_lim",
    @c_discount = "c_discount",
    @c_since = "c_since",
    @c_id_local = "c_id"
```

```
    where "c_id" = @c_id and "c_w_id" = @c_w_id and "c_d_id" = @c_d_id
end if;
if sqlcode = -306 then // deadlock
  rollback work;
  raiserror 29000 'Deadlock updating customer';
  return
end if;
if @c_id_local is null then
  rollback work;
  raiserror 23000 'Customer not found';
  return
end if;
if(@c_credit = 'BC') then
  select "c_data" into @data from "customer" where "c_id" = @c_id
    and "c_w_id" = @c_w_id and "c_d_id" = @c_d_id;
  set
    @c_data = convert(char(5),@c_id)
    +convert(char(4),@c_d_id)
    +convert(char(5),@c_w_id)
    +convert(char(4),@d_id)
    +convert(char(5),@w_id)
    +convert(char(19),@h_amount)
    +"substring"(@data,1,458);
  update "customer" set "c_data" = @c_data
    where "c_id" = @c_id
    and "c_w_id" = @c_w_id
    and "c_d_id" = @c_d_id;
  set @screen_data = "substring"(@c_data,1,200)
end if;
update "district"
  set "d_ytd" = "d_ytd"+@h_amount,
  @d_street_1 = "d_street_1",
  @d_street_2 = "d_street_2",
  @d_city = "d_city",
  @d_state = "d_state",
  @d_zip = "d_zip",
  @d_name = "d_name",
  @d_id_local = "d_id"
  where "d_w_id" = @w_id and "d_id" = @d_id;
update "warehouse"
  set "w_ytd" = "w_ytd"+@h_amount,
  @w_street_1 = "w_street_1",
  @w_street_2 = "w_street_2",
  @w_city = "w_city",
  @w_state = "w_state",
  @w_zip = "w_zip",
  @w_name = "w_name",
  @w_id_local = "w_id"
  where "w_id" = @w_id;
insert into "history" values( @c_id_local,
  @c_d_id,
  @c_w_id,
  @d_id_local,
  @w_id_local,
  @datetime,
  @h_amount,
  @w_name
  +' '+@d_name ) ;
commit work; /*ACID_DELAY*/
select @c_id,
  @c_last,
  @datetime,
  @w_street_1,
  @w_street_2,
  @w_city,
  @w_state,
  @w_zip,
```

```
    @d_street_1,
    @d_street_2,
    @d_city,
    @d_state,
    @d_zip,
    @c_first,
    @c_middle,
    @c_street_1,
    @c_street_2,
    @c_city,
    @c_state,
    @c_zip,
    @c_phone,
    @c_since,
    @c_credit,
    @c_credit_lim,
    @c_discount,
    @c_balance,
    @screen_data
end
go

COMMENT TO PRESERVE FORMAT ON PROCEDURE "DBA"."tpcc_payment" IS
{create procedure DBA.tpcc_payment(
  in @w_id smallint,
  in @c_w_id smallint,
  in @h_amount numeric(6,2),
  in @d_id tinyint,
  in @c_d_id tinyint,
  in @c_id integer,
  in @c_last char(16) default '' )
begin
  declare @w_name char(10);
  declare @d_name char(10);
  declare @data char(500);
  declare @c_data char(500);
  declare @w_ytd numeric(12,2);
  declare @d_ytd numeric(12,2);
  declare @cnt smallint;
  declare @val smallint;
  declare @d_id_local tinyint;
  declare @w_id_local smallint;
  declare @c_id_local integer;
  declare @datetime datetime;
  declare @w_street_1 char(20);
  declare @w_street_2 char(20);
  declare @w_city char(20);
  declare @w_state char(2);
  declare @w_zip char(9);
  declare @d_street_1 char(20);
  declare @d_street_2 char(20);
  declare @d_city char(20);
  declare @d_state char(2);
  declare @d_zip char(9);
  declare @c_first char(16);
  declare @c_middle char(2);
  declare @c_street_1 char(20);
  declare @c_street_2 char(20);
  declare @c_city char(20);
  declare @c_state char(2);
  declare @c_zip char(9);
  declare @c_phone char(16);
  declare @c_since datetime;
  declare @c_credit char(2);
  declare @c_credit_lim numeric(12,2);
  declare @c_discount numeric(4,4);
  declare @c_balance numeric(12,2);
```

```
declare @screen_data char(200);

set @screen_data = '';
set @datetime = CURRENT TIMESTAMP;

if(@c_id = 0) then
  begin
    declare c dynamic scroll cursor for select c_id
        from customer with(repeatableread)
        where c_last = @c_last and c_w_id = @c_w_id and c_d_id = @c_d_id
        order by c_w_id asc,c_d_id asc,c_last asc,c_first asc for update;

  select count(*) into @cnt
    from customer with(repeatableread)
    where c_last = @c_last and c_w_id = @c_w_id and c_d_id = @c_d_id;

  set @val = (@cnt+1)/2;
open c;
fetch absolute @val c
  into @c_id;

update customer
  set @c_balance = c_balance-@h_amount,
  c_balance = c_balance-@h_amount,
  c_payment_cnt = c_payment_cnt+1,
  c_ytd_payment = c_ytd_payment+@h_amount,
  @c_first = c_first,
  @c_middle = c_middle,
  @c_last = c_last,
  @c_street_1 = c_street_1,
  @c_street_2 = c_street_2,
  @c_city = c_city,
  @c_state = c_state,
  @c_zip = c_zip,
  @c_phone = c_phone,
  @c_credit = c_credit,
  @c_credit_lim = c_credit_lim,
  @c_discount = c_discount,
  @c_since = c_since,
  @c_id_local = c_id
  where current of c;

close c;

  end;

else
  update customer
  set @c_balance = c_balance-@h_amount,
  c_balance = c_balance-@h_amount,
  c_payment_cnt = c_payment_cnt+1,
  c_ytd_payment = c_ytd_payment+@h_amount,
  @c_first = c_first,
  @c_middle = c_middle,
  @c_last = c_last,
  @c_street_1 = c_street_1,
  @c_street_2 = c_street_2,
  @c_city = c_city,
  @c_state = c_state,
  @c_zip = c_zip,
  @c_phone = c_phone,
  @c_credit = c_credit,
  @c_credit_lim = c_credit_lim,
  @c_discount = c_discount,
  @c_since = c_since,
  @c_id_local = c_id
  where c_id = @c_id and c_w_id = @c_w_id and c_d_id = @c_d_id;
```

```
end if;

if sqlcode = -306 then // deadlock
  rollback;
  raiserror 29000 'Deadlock updating customer';
  return
end if;

if @c_id_local is null then
  rollback;
  raiserror 23000 'Customer not found';
  return
end if;

if(@c_credit = 'BC') then
  select c_data into @data from customer where c_id = @c_id
    and c_w_id = @c_w_id and c_d_id = @c_d_id;
  set
    @c_data = convert(char(5),@c_id)
    +convert(char(4),@c_d_id)
    +convert(char(5),@c_w_id)
    +convert(char(4),@d_id)
    +convert(char(5),@w_id)
    +convert(char(19),@h_amount)
    +substring(@data,1,458);
  update customer set c_data = @c_data
    where c_id = @c_id
    and c_w_id = @c_w_id
    and c_d_id = @c_d_id;
  set @screen_data = substring(@c_data,1,200)
end if;

update district
  set d_ytd = d_ytd+@h_amount,
  @d_street_1 = d_street_1,
  @d_street_2 = d_street_2,
  @d_city = d_city,
  @d_state = d_state,
  @d_zip = d_zip,
  @d_name = d_name,
  @d_id_local = d_id
  where d_w_id = @w_id and d_id = @d_id;

update warehouse
  set w_ytd = w_ytd+@h_amount,
  @w_street_1 = w_street_1,
  @w_street_2 = w_street_2,
  @w_city = w_city,
  @w_state = w_state,
  @w_zip = w_zip,
  @w_name = w_name,
  @w_id_local = w_id
  where w_id = @w_id;

insert into history values( @c_id_local,
  @c_d_id,
  @c_w_id,
  @d_id_local,
  @w_id_local,
  @datetime,
  @h_amount,
  @w_name
  +' '+@d_name ) ;

/*ACID_DELAY*/commit;

select @c_id,
```

```
    @c_last,
    @datetime,
    @w_street_1,
    @w_street_2,
    @w_city,
    @w_state,
    @w_zip,
    @d_street_1,
    @d_street_2,
    @d_city,
    @d_state,
    @d_zip,
    @c_first,
    @c_middle,
    @c_street_1,
    @c_street_2,
    @c_city,
    @c_state,
    @c_zip,
    @c_phone,
    @c_since,
    @c_credit,
    @c_credit_lim,
    @c_discount,
    @c_balance,
    @screen_data
end
}
go

create procedure "DBA"."tpcc_stocklevel"(
  in @w_id smallint,
  in @d_id tinyint,
  in @threshhold smallint )
begin
  declare @o_id_low integer;
  declare @o_id_high integer;
  select("d_next_o_id"-20),
    ("d_next_o_id"-1) into @o_id_low,@o_id_high
    from "district"
    where "d_w_id" = @w_id and "d_id" = @d_id;
  select "count"(distinct("s_i_id"))
    from "stock","order_line"
    where "ol_w_id" = @w_id
    and "ol_d_id" = @d_id
    and "ol_o_id" >= @o_id_low
    and "ol_o_id" <= @o_id_high
    and "s_w_id" = "ol_w_id"
    and "s_i_id" = "ol_i_id"
    and "s_quantity" < @threshhold
end
go

COMMENT TO PRESERVE FORMAT ON PROCEDURE "DBA"."tpcc_stocklevel" IS
{create procedure DBA.tpcc_stocklevel(
  in @w_id smallint,
  in @d_id tinyint,
  in @threshhold smallint )
begin
  declare @o_id_low integer;
  declare @o_id_high integer;

  select(d_next_o_id-20),
    (d_next_o_id-1) into @o_id_low,@o_id_high
    from district
    where d_w_id = @w_id and d_id = @d_id;
```

```
  select count(distinct(s_i_id))
    from stock,order_line
    where ol_w_id = @w_id
    and ol_d_id = @d_id
    and ol_o_id >= @o_id_low
    and ol_o_id <= @o_id_high
    and s_w_id = ol_w_id
    and s_i_id = ol_i_id
    and s_quantity < @threshhold
end
}
go

create procedure "DBA"."tpcc_neworder"(
  in @w_id smallint,
  in @d_id tinyint,
  in @c_id integer,
  in @o_ol_cnt tinyint,
  in @o_all_local tinyint,
  in @i_id1 integer,
  in @s_w_id1 smallint,
  in @ol_qty1 smallint,
  in @i_id2 integer,
  in @s_w_id2 smallint,
  in @ol_qty2 smallint,
  in @i_id3 integer,
  in @s_w_id3 smallint,
  in @ol_qty3 smallint,
  in @i_id4 integer,
  in @s_w_id4 smallint,
  in @ol_qty4 smallint,
  in @i_id5 integer,
  in @s_w_id5 smallint,
  in @ol_qty5 smallint,
  in @i_id6 integer,
  in @s_w_id6 smallint,
  in @ol_qty6 smallint,
  in @i_id7 integer,
  in @s_w_id7 smallint,
  in @ol_qty7 smallint,
  in @i_id8 integer,
  in @s_w_id8 smallint,
  in @ol_qty8 smallint,
  in @i_id9 integer,
  in @s_w_id9 smallint,
  in @ol_qty9 smallint,
  in @i_id10 integer,
  in @s_w_id10 smallint,
  in @ol_qty10 smallint,
  in @i_id11 integer,
  in @s_w_id11 smallint,
  in @ol_qty11 smallint,
  in @i_id12 integer,
  in @s_w_id12 smallint,
  in @ol_qty12 smallint,
  in @i_id13 integer,
  in @s_w_id13 smallint,
  in @ol_qty13 smallint,
  in @i_id14 integer,
  in @s_w_id14 smallint,
  in @ol_qty14 smallint,
  in @i_id15 integer,
  in @s_w_id15 smallint,
  in @ol_qty15 smallint )
begin
  declare @c_discount numeric(4,4);
  declare @i_price numeric(5,2);
```

```
declare @i_name char(24);
declare @i_data char(50);
declare @o_entry_d "datetime";
declare @remote_flag integer;
declare @s_quantity smallint;
declare @s_data char(50);
declare @s_dist char(24);
declare @li_no integer;
declare @li_id integer;
declare @li_s_w_id smallint;
declare @li_qty smallint;
declare @ol_number integer;
declare @c_id_local integer;
declare @w_tax numeric(4,4);
declare @d_tax numeric(4,4);
declare @o_id integer;
declare @c_last char(16);
declare @c_credit char(2);
declare @commit_flag tinyint;
truncate table "DBA"."neworder_results";
update "district"
  set @d_tax = "d_tax",
  @o_id = "d_next_o_id",
  "d_next_o_id" = "d_next_o_id"+1,
  @o_entry_d = "getdate"(),
  @li_no = 0,
  @commit_flag = 1
  where "d_w_id" = @w_id and "d_id" = @d_id;
while(@li_no < @o_ol_cnt) loop
  set @li_no = @li_no+1;
  select case @li_no
    when 1 then @i_id1
    when 2 then @i_id2
    when 3 then @i_id3
    when 4 then @i_id4
    when 5 then @i_id5
    when 6 then @i_id6
    when 7 then @i_id7
    when 8 then @i_id8
    when 9 then @i_id9
    when 10 then @i_id10
    when 11 then @i_id11
    when 12 then @i_id12
    when 13 then @i_id13
    when 14 then @i_id14
    when 15 then @i_id15 end,
    case @li_no
    when 1 then @s_w_id1
    when 2 then @s_w_id2
    when 3 then @s_w_id3
    when 4 then @s_w_id4
    when 5 then @s_w_id5
    when 6 then @s_w_id6
    when 7 then @s_w_id7
    when 8 then @s_w_id8
    when 9 then @s_w_id9
    when 10 then @s_w_id10
    when 11 then @s_w_id11
    when 12 then @s_w_id12
    when 13 then @s_w_id13
    when 14 then @s_w_id14
    when 15 then @s_w_id15 end,
    case @li_no
    when 1 then @ol_qty1
    when 2 then @ol_qty2
    when 3 then @ol_qty3
    when 4 then @ol_qty4
```

```
    when 5 then @ol_qty5
    when 6 then @ol_qty6
    when 7 then @ol_qty7
    when 8 then @ol_qty8
    when 9 then @ol_qty9
    when 10 then @ol_qty10
    when 11 then @ol_qty11
    when 12 then @ol_qty12
    when 13 then @ol_qty13
    when 14 then @ol_qty14
    when 15 then @ol_qty15 end
    into @li_id,@li_s_w_id,@li_qty;
  select "i_price",
    "i_name",
    "i_data" into @i_price,@i_name,@i_data
    from "item" with(repeatableread)
    where "i_id" = @li_id;
  /*ACID7*/
  update "stock"
    set "s_ytd" = "s_ytd"+@li_qty,
    @s_quantity = "s_quantity"-@li_qty
    +case when("s_quantity"-@li_qty < 10) then 91 else 0 end,
    "s_quantity" = "s_quantity"-@li_qty
    +case when("s_quantity"-@li_qty < 10) then 91 else 0 end,
    "s_order_cnt" = "s_order_cnt"+1,
    "s_remote_cnt" = "s_remote_cnt"
    +case when(@li_s_w_id = @w_id) then 0 else 1 end,
    @s_data = "s_data",
    @s_dist
     = case @d_id
    when 1 then "s_dist_01"
    when 2 then "s_dist_02"
    when 3 then "s_dist_03"
    when 4 then "s_dist_04"
    when 5 then "s_dist_05"
    when 6 then "s_dist_06"
    when 7 then "s_dist_07"
    when 8 then "s_dist_08"
    when 9 then "s_dist_09"
    when 10 then "s_dist_10" end
    where "s_i_id" = @li_id and "s_w_id" = @li_s_w_id;
  if(@@rowcount > 0) then
    insert into "order_line" values
      ( @o_id,
      @d_id,
      @w_id,
      @li_no,
      @li_id,
      @li_s_w_id,
      null,
      @li_qty,
      @i_price*@li_qty,
      @s_dist ) ;
    insert into "neworder_results" values
      ( @li_no,
      @i_name,
      @s_quantity,
      case when(("locate"(@i_data,'ORIGINAL') > 0)
      and("locate"(@s_data,'ORIGINAL') > 0)) then
        'B' else 'G' end,
      @i_price,
      @i_price*@li_qty )
  else
    insert into "neworder_results" values( @li_no,'',0,'',0,0 ) ;
    set @commit_flag = 0
  end if
 end loop;
```

```
  select "c_last",
    "c_discount",
    "c_credit",
    "c_id" into @c_last,@c_discount,@c_credit,@c_id_local
    from "customer" with(repeatableread)
    where "c_id" = @c_id and "c_w_id" = @w_id and "c_d_id" = @d_id;
  insert into "orders" values
    ( @o_id,
    @d_id,
    @w_id,
    @c_id_local,
    @o_entry_d,
    null,
    @o_ol_cnt,
    @o_all_local ) ;
  insert into "new_order" values( @o_id,@d_id,@w_id ) ;
  select "w_tax"
    into @w_tax
    from "warehouse" with(repeatableread)
    where "w_id" = @w_id;
  if(@commit_flag = 1) then
    commit work /*ACID_DELAY*/
  else
    rollback work
  end if;
  select @w_tax,
    @d_tax,
    @o_id,
    @c_last,
    @c_discount,
    @c_credit,
    @o_entry_d,
    @commit_flag,
    "li_no",
    "i_name",
    "s_quantity",
    "b_g",
    "price",
    "amount"
    from "neworder_results" order by "li_no" asc
end
go

COMMENT TO PRESERVE FORMAT ON PROCEDURE "DBA"."tpcc_neworder" IS
{create procedure DBA.tpcc_neworder(
  in @w_id smallint,
  in @d_id tinyint,
  in @c_id integer,
  in @o_ol_cnt tinyint,
  in @o_all_local tinyint,
  in @i_id1 integer,
  in @s_w_id1 smallint,
  in @ol_qty1 smallint,
  in @i_id2 integer,
  in @s_w_id2 smallint,
  in @ol_qty2 smallint,
  in @i_id3 integer,
  in @s_w_id3 smallint,
  in @ol_qty3 smallint,
  in @i_id4 integer,
  in @s_w_id4 smallint,
  in @ol_qty4 smallint,
  in @i_id5 integer,
  in @s_w_id5 smallint,
  in @ol_qty5 smallint,
  in @i_id6 integer,
  in @s_w_id6 smallint,
```

```
  in @ol_qty6 smallint,
  in @i_id7 integer,
  in @s_w_id7 smallint,
  in @ol_qty7 smallint,
  in @i_id8 integer,
  in @s_w_id8 smallint,
  in @ol_qty8 smallint,
  in @i_id9 integer,
  in @s_w_id9 smallint,
  in @ol_qty9 smallint,
  in @i_id10 integer,
  in @s_w_id10 smallint,
  in @ol_qty10 smallint,
  in @i_id11 integer,
  in @s_w_id11 smallint,
  in @ol_qty11 smallint,
  in @i_id12 integer,
  in @s_w_id12 smallint,
  in @ol_qty12 smallint,
  in @i_id13 integer,
  in @s_w_id13 smallint,
  in @ol_qty13 smallint,
  in @i_id14 integer,
  in @s_w_id14 smallint,
  in @ol_qty14 smallint,
  in @i_id15 integer,
  in @s_w_id15 smallint,
  in @ol_qty15 smallint )
begin
  declare @c_discount numeric(4,4);
  declare @i_price numeric(5,2);
  declare @i_name char(24);
  declare @i_data char(50);
  declare @o_entry_d datetime;
  declare @remote_flag integer;
  declare @s_quantity smallint;
  declare @s_data char(50);
  declare @s_dist char(24);
  declare @li_no integer;
  declare @li_id integer;
  declare @li_s_w_id smallint;
  declare @li_qty smallint;
  declare @ol_number integer;
  declare @c_id_local integer;
  declare @w_tax numeric(4,4);
  declare @d_tax numeric(4,4);
  declare @o_id integer;
  declare @c_last char(16);
  declare @c_credit char(2);
  declare @commit_flag tinyint;

  truncate table DBA.neworder_results;

  update district
    set @d_tax = d_tax,
    @o_id = d_next_o_id,
    d_next_o_id = d_next_o_id+1,
    @o_entry_d = getdate(*),
    @li_no = 0,
    @commit_flag = 1
    where d_w_id = @w_id and d_id = @d_id;

  while(@li_no < @o_ol_cnt) loop
    set @li_no = @li_no+1;
    select
                case @li_no
                        when 1    then @i_id1
```

```
                when 2    then @i_id2
                when 3    then @i_id3
                when 4    then @i_id4
                when 5    then @i_id5
                when 6    then @i_id6
                when 7    then @i_id7
                when 8    then @i_id8
                when 9    then @i_id9
                when 10   then @i_id10
                when 11   then @i_id11
                when 12   then @i_id12
                when 13   then @i_id13
                when 14   then @i_id14
                when 15   then @i_id15
        end,
        case @li_no
                when 1    then @s_w_id1
                when 2    then @s_w_id2
                when 3    then @s_w_id3
                when 4    then @s_w_id4
                when 5    then @s_w_id5
                when 6    then @s_w_id6
                when 7    then @s_w_id7
                when 8    then @s_w_id8
                when 9    then @s_w_id9
                when 10   then @s_w_id10
                when 11   then @s_w_id11
                when 12   then @s_w_id12
                when 13   then @s_w_id13
                when 14   then @s_w_id14
                when 15   then @s_w_id15
        end,
        case @li_no
                when 1    then @ol_qty1
                when 2    then @ol_qty2
                when 3    then @ol_qty3
                when 4    then @ol_qty4
                when 5    then @ol_qty5
                when 6    then @ol_qty6
                when 7    then @ol_qty7
                when 8    then @ol_qty8
                when 9    then @ol_qty9
                when 10   then @ol_qty10
                when 11   then @ol_qty11
                when 12   then @ol_qty12
                when 13   then @ol_qty13
                when 14   then @ol_qty14
                when 15   then @ol_qty15
        end
        into @li_id, @li_s_w_id, @li_qty;

select i_price,
  i_name,
  i_data into @i_price,@i_name,@i_data
  from item with(repeatableread)
  where i_id = @li_id;

/*ACID7*/

update stock
  set s_ytd = s_ytd+@li_qty,
  @s_quantity = s_quantity-@li_qty
  +case when(s_quantity-@li_qty < 10) then 91 else 0 end,
  s_quantity = s_quantity-@li_qty
  +case when(s_quantity-@li_qty < 10) then 91 else 0 end,
  s_order_cnt = s_order_cnt+1,
  s_remote_cnt = s_remote_cnt
```

```
    +case when(@li_s_w_id = @w_id) then 0 else 1 end,
    @s_data = s_data,
    @s_dist
     = case @d_id
    when 1 then s_dist_01
    when 2 then s_dist_02
    when 3 then s_dist_03
    when 4 then s_dist_04
    when 5 then s_dist_05
    when 6 then s_dist_06
    when 7 then s_dist_07
    when 8 then s_dist_08
    when 9 then s_dist_09
    when 10 then s_dist_10 end
    where s_i_id = @li_id and s_w_id = @li_s_w_id;

  if(@@rowcount > 0) then
    insert into order_line
      values( @o_id,
      @d_id,
      @w_id,
      @li_no,
      @li_id,
      @li_s_w_id,
      null,
      @li_qty,
      @i_price*@li_qty,
      @s_dist ) ;

    insert into neworder_results
      values( @li_no,
      @i_name,
      @s_quantity,
      case when((locate(@i_data,'ORIGINAL') > 0)
      and(locate(@s_data,'ORIGINAL') > 0)) then
        'B' else 'G' end,
      @i_price,
      @i_price*@li_qty )
  else
    insert into neworder_results values( @li_no,'',0,'',0,0 ) ;

    set @commit_flag = 0
  end if
end loop;

select c_last,
  c_discount,
  c_credit,
  c_id into @c_last,@c_discount,@c_credit,@c_id_local
  from customer with(repeatableread)
  where c_id = @c_id and c_w_id = @w_id and c_d_id = @d_id;

insert into orders
  values( @o_id,
  @d_id,
  @w_id,
  @c_id_local,
  @o_entry_d,
  null,
  @o_ol_cnt,
  @o_all_local ) ;

insert into new_order values( @o_id,@d_id,@w_id ) ;

select w_tax into @w_tax
  from warehouse with(repeatableread)
  where w_id = @w_id;
```

```
  if(@commit_flag = 1) then
    /*ACID_DELAY*/commit;
  else
    rollback;
  end if;

  select @w_tax,
    @d_tax,
    @o_id,
    @c_last,
    @c_discount,
    @c_credit,
    @o_entry_d,
    @commit_flag,
    li_no,
    i_name,
    s_quantity,
    b_g,
    price,
    amount
    from neworder_results order by li_no;
end
}
go

begin
  declare prev_count int;
  declare new_count int;
  declare local temporary table dependent_proc (
    proc_id    unsigned int    not null,
    primary key (proc_id)
  ) in system not transactional;
  set prev_count = -1;
  lp: loop
      truncate table dependent_proc;
      insert into dependent_proc
              select proc_id from SYS.SYSPROCEDURE p
              where exists (select * from SYS.SYSPROCPARM pp
                  where pp.proc_id = p.proc_id
                  and parm_name = 'expression'
                  and parm_type = 1
                 and domain_id = 1);
      select count(*) into new_count from dependent_proc;
      if new_count = 0 or (new_count >= prev_count and prev_count >= 0) then
        leave lp;
      end if;
      set prev_count = new_count;
      for l1 as c1 cursor for
              select u.user_name, proc_name
              from SYS.SYSPROCEDURE p
                  join dependent_proc d on (d.proc_id = p.proc_id)
                  join SYS.SYSUSER u on (p.creator = u.user_id)
      do
        begin
                execute immediate with quotes on
            'alter procedure ''' || user_name || '''.''' || proc_name || ''' recompile';
          exception when others then
        end
      end for;
  end loop;
end

go

call dbo.sa_recompile_views(0)
go
```

```
------------------------------------------------
--  Set DBA password
------------------------------------------------

GRANT CONNECT TO DBA IDENTIFIED BY ENCRYPTED
'\x01\x56\x7a\xaf\xbc\x04\x7d\x32\xc5\x7b\xb0\x9d\x8f\xfe\x84\xf3\xcb\x09\x87\x3b\xeb\xc4\x38\x04\x31\x26\x34\xef\x7a\xae\
xe2\x6b\xce\x2d\xdc\xba\x75'
go


------------------------------------------------
--  Create options
------------------------------------------------

SET OPTION date_order =
go

SET OPTION PUBLIC.preserve_source_format =
go

SET OPTION "PUBLIC"."max_plans_cached"='100'
go

SET OPTION "PUBLIC"."preserve_source_format"='On'
go
```

APPENDIX E: SYSTEM CONFIGURATION

'Engine properties'
0,'ActiveReq','Active requests','1'
1,'AvailIO','Number of available I/O control blocks','0'
2,'BytesReceived','Bytes received by server','5768'
3,'BytesReceivedUncomp','Bytes received after decompression','5768'
4,'BytesSent','Bytes sent to client','19391'
5,'BytesSentUncomp','Bytes sent before compression','19391'
6,'CacheAllocated','Cache pages that have been allocated for server data structures','30599'
7,'CacheFile','Cache pages used to hold data from database files','24808'
8,'CacheFileDirty','Cache pages that are dirty (needing a write)','0'
9,'CacheFree','Number of cache pages not being used','29221979'
10,'CacheHits','Cache Hits','20633'
13,'CachePanics','Number of times the cache manager has failed to find a page to allocate','0'
14,'CachePinned','Pinned cache pages','30611'
15,'CacheRead','Cache reads','20726'
16,'CacheReplacements','Cache replacements','0'
17,'CacheScavengeVisited','Number of pages visited while scavenging for a page to allocate','55453'
18,'CacheScavenges','Number of times the cache manager has scavenged for a page to allocate','55453'
21,'CarverHeapPages','Cache pages used for carvers','0'
37,'ClientStmtCacheHits','Number of prepares not required because of the client statement cache','0'
38,'ClientStmtCacheMisses','Number of prepares in the client statement cache which were prepared again','0'
39,'Commit','Number of commit requests','11'
45,'CurrentCacheSize','Current cache size in kilobytes','120586240'
49,'Cursor','Declared cursors','4'
52,'DiskRead','Disk reads','22580'
55,'DiskReadHintScatterLimit','Imposed limit on the size (in bytes) of a scatter read hint','0'
56,'DiskRetryRead','Disk read retries','0'
57,'DiskRetryReadScatter','Disk read retries for scattered reads','0'
58,'DiskRetryWrite','Disk write retries','0'
68,'ExchangeTasks','Exchange tasks currently being executed','0'
69,'ExchangeTasksCompleted','Total number of exchange tasks that have been executed','0'
80,'FreeBuffers','Free communication buffers','7'
87,'HeapsCarver','Number of heaps used for carvers','0'
88,'HeapsLocked','Number of relocatable heaps currently locked in cache','4'
89,'HeapsQuery','Number of heaps used for query processing (hash and sort operations)','0'
90,'HeapsRelocatable','Number of relocatable heaps','15'
104,'LockedCursorPages','Number of pages used to keep cursor heaps pinned in memory','4'
105,'LockedHeapPages','Heap pages locked in cache','12'
106,'MainHeapBytes','Main heap bytes in cache','121109136'
107,'MainHeapPages','Main heap pages in cache','30484'
108,'MapPhysicalMemoryEng','Map physical memory','0'
109,'MaxCacheSize','Maximum cache size in kilobytes','120586240'
113,'MinCacheSize','Minimum cache size in kilobytes','120586240'
115,'MultiPacketsReceived','Number of multi-packet receives','0'
116,'MultiPacketsSent','Number of multi-packet sends','1'
117,'MultiPageAllocs','Number of multi-page allocations','5'
118,'CursorOpen','Open cursors','4'
119,'PacketsReceived','Packets received by server','42'
120,'PacketsReceivedUncomp','Packets received after decompression','42'
121,'PacketsSent','Packets sent to client','42'
122,'PacketsSentUncomp','Packets sent before compression','42'
124,'PeakCacheSize','Peak cache size in kilobytes','120586240'
134,'QueryHeapPages','Cache pages used for query processing (hash and sort operations)','0'
137,'QueryMemActiveCurr','The current number of requests actively using query memory','0'
138,'QueryMemActiveEst','The server''s estimate of the steady state average of the number of requests actively using query memory','0'
139,'QueryMemExtraAvail','The amount of memory available to grant beyond the base memory-intensive grant','0'
140,'QueryMemGrantFailed','The total number of times any request waited for query memory and failed to get it','0'
141,'QueryMemGrantGranted','The number of pages currently granted to requests','0'
142,'QueryMemGrantRequested','The total number of times any request attempted to acquire query memory','0'
143,'QueryMemGrantWaited','The total number of times any request waited for query memory','0'
144,'QueryMemGrantWaiting','The current number of requests waiting for query memory','0'
159,'RemoteputWait','Remote put waits','0'

160,'Req','Requests','43'
161,'RequestsReceived','Requests received by server','42'
162,'Rlbk','Rollback requests handled','0'
164,'SendFail','Failed communication sends','0'
171,'PrepStmt','Prepared statements','5'
177,'StreamsUsed','Number of engine streams in use','1'
180,'ThreadDeadlocksAvoided','The number of thread deadlock errors that were avoided by dynamically increasing the multiprogramming level','0'
181,'ThreadDeadlocksReported','The number of thread deadlock errors that were returned to applications','0'
182,'TotalBuffers','Total communication buffers','10'
184,'UniqueClientAddresses','Number of unique client network addresses connected','0'
185,'UnschReq','Unscheduled requests','0'
186,'UserDefinedCounterRate01','A rate counter that is set by the database application (counter 1)','0'
187,'UserDefinedCounterRate02','A rate counter that is set by the database application (counter 2)','0'
188,'UserDefinedCounterRate03','A rate counter that is set by the database application (counter 3)','0'
189,'UserDefinedCounterRate04','A rate counter that is set by the database application (counter 4)','0'
190,'UserDefinedCounterRate05','A rate counter that is set by the database application (counter 5)','0'
191,'UserDefinedCounterRaw01','A counter that is set by the database application (counter 1)','0'
192,'UserDefinedCounterRaw02','A counter that is set by the database application (counter 2)','0'
193,'UserDefinedCounterRaw03','A counter that is set by the database application (counter 3)','0'
194,'UserDefinedCounterRaw04','A counter that is set by the database application (counter 4)','0'
195,'UserDefinedCounterRaw05','A counter that is set by the database application (counter 5)','0'
199,'Name','Name','tpcc_einstein'
202,'PageSize','Database page size','4096'
214,'TcpIpAddresses','TCP/IP addresses that can be used to connect via TCP/IP','(fd77:55d:59d9:56b:1461:b98a:200f:d6f8):2638;10.7.161.230:2638;192.168.8.8:2638'
222,'Platform','Operating system platform','Windows2012R2'
223,'PlatformVer','Operating system platform version','Windows 2012R2 Build 9600 '
224,'MessageWindowSize','Maximum number of messages retained from server window','400'
225,'MaxMessage','Maximum message number in server window','39'
226,'Message','Message from server window','10/29 09:02:13. SQL Anywhere Network Server Version 16.0.0.2020\x0d\x0a'
227,'MessageText','Message text from server window','SQL Anywhere Network Server Version 16.0.0.2020\x0d\x0a'
228,'MessageTime','Message time from server window','2014-10-29 09:02:13.127'
230,'ProductName','Product name','SQL Anywhere'
231,'ProductVersion','Product version','16.0.0.2020'
232,'CompanyName','Company name','SAP AG or an SAP affiliate company.'
233,'LegalCopyright','Copyright notice','Copyright © 2014 SAP AG or an SAP affiliate company. All rights reserved. Use of this software is governed by the Sybase License Agreement. Refer to http://www.sybase.com/softwarelicenses.'
234,'LegalTrademarks','Legal trademarks','Sybase is a trademark of Sybase, Inc.'
235,'QuittingTime','Server quitting time','none'
236,'ConnsDisabled','Connections disabled','No'
237,'RequestLogging','Request logging','NONE'
238,'RequestLogFile','Request logging filename',''
239,'RequestLogNumFiles','Request log file count','-1'
240,'RequestLogMaxSize','Request log maximum size','0'
241,'RequestFilterConn','Request log connection filter','-1'
242,'RequestFilterDB','Request log database filter','-1'
243,'LivenessTimeout','Client liveness timeout default','120'
244,'ProfileFilterConn','Procedure profile connection filter','-1'
245,'ProfileFilterUser','Procedure profile user filter',''
248,'CharSet','Character set used for CHAR data','windows-1252'
251,'Language','Language','us_english'
255,'DefaultCollation','Default Collation','1252LATIN1'
256,'DefaultNcharCollation','Default NCHAR Collation','UCA'
267,'TimeZoneAdjustment','Time zone adjustment from UTC time in minutes','-240'
269,'NumPhysicalProcessors','Number of physical processors on server machine','2'
270,'NumPhysicalProcessorsUsed','Number of physical processors which the server will use','1'
271,'NumLogicalProcessors','Number of logical processors on server machine','40'
272,'NumLogicalProcessorsUsed','Number of logical processors which the server will use','20'
273,'LicensedCompany','Name of the licensed company','Sybase, Inc. (Waterloo)'
274,'LicensedUser','Name of the licensed user','Internal Use'
275,'LicenseType','License type','networked seat (per-seat)'
276,'LicenseCount','Number of licensed seats','10000'
277,'LicenseKey','Number of licensed seats','*****-*****-*****-*****-*****'
282,'MachineName','Name of the machine','EINSTEIN'
284,'CompactPlatformVer','Operating system platform version in compact form','W2012R2 #9600 '

285,'CommandLine','Expanded command line used to start the server','-x tcpip -n tpcc_einstein -ca 0 -ch 115g -cl 115g -gr 9999 -gc 59 -gn 500 -gna 0 -gt 1 -o c:\\tpcc_output_logs\\srv.out d:\\tpcc_data\\tpcc.db '
286,'CollectStatistics','Server and database performance statistics are collected','Yes'
288,'MessageCategoryLimit','The minimum number of console messages of each type stored by the server','400'
292,'ProcessID','Process ID','5452'
293,'ProcessCPU','Process CPU usage','2.625000'
294,'ProcessCPUUser','Process CPU User usage','1.968750'
295,'ProcessCPUSystem','Process CPU System usage','0.656250'
296,'IsNetworkServer','Is Network Server','Yes'
297,'IsRuntimeServer','Is Runtime Server','No'
298,'IsService','Is Service','Yes'
299,'BuildClient','Codeline name','dbbuild_16_maint16'
300,'BuildChange','Codeline revision','841518'
301,'BuildReproducible','Codeline state','0'
302,'BuildProduction','Build type','Yes'
312,'RememberLastStatement','Remember last statement','No'
313,'RememberLastPlan','Remember last plan','No'
316,'FunctionName','Function name','abs'
317,'FunctionMinParms','Function minimum number of parameters','1'
318,'FunctionMaxParms','Function maximum number of parameters','1'
319,'OmniIdentifier','Omni Identifier','253d7e9cdc8248b29955d762c8c328b8'
320,'IdleTimeout','Idle timeout','240'
321,'ProcessorArchitecture','Processor architecture','X86_64'
322,'NativeProcessorArchitecture','Architecture of native processor when current processor is emulated','X86_64'
323,'TempDir','Temporary directory','C:\\Users\\ADMINI~1\\AppData\\Local\\Temp'
324,'StartTime','Server start time','2014-10-29 09:02:13.111'
326,'IsIQ','Is IQ Server','No'
327,'MultiProgrammingLevel','The maximum number of tasks which the server can process concurrently','500'
330,'ConsoleLogFile','Console log filename','c:\\tpcc_output_logs\\srv.out'
337,'HttpPorts','Http Server TCP/IP port number(s)',''
338,'HttpsPorts','Https Server TCP/IP port number(s)',''
339,'HttpAddresses','TCP/IP addresses that can be used to connect via HTTP',''
340,'HttpsAddresses','TCP/IP addresses that can be used to connect via HTTPS',''
342,'HttpNumConnections','Total number of HTTP connections','0'
343,'HttpsNumConnections','Total number of HTTPS (secure) connections','0'
344,'HttpNumActiveReq','Total number of active HTTP connections','0'
345,'HttpsNumActiveReq','Total number of active HTTPS (secure) connections','0'
346,'HttpNumSessions','Total number of HTTP sessions (passive and active)','0'
354,'IsFipsAvailable','FIPS Available','Yes'
355,'FipsMode','FIPS Mode','No'
356,'IsAesniAvailable','SERVER USES AESNI','Yes'
358,'StartDBPermission','Start database permission','DBA'
366,'ServerName','Server name','tpcc_einstein'
372,'RequestTiming','Monitor the cost of requests','No'
374,'ReqCountUnscheduled','Number of times waited for scheduling',
375,'ReqCountActive','Number of requests processed',
376,'ReqCountBlockIO','Number of times waited for I/O to complete','99'
377,'ReqCountBlockLock','Number of times waited for a lock',
378,'ReqCountBlockContention','Number of times waited for atomic access',
379,'ReqTimeUnscheduled','Time spent unscheduled',
380,'ReqTimeActive','Time spent processing requests',
381,'ReqTimeBlockIO','Time spent waiting for I/O to complete','.012889714279161'
382,'ReqTimeBlockLock','Time spent waiting for a lock',
383,'ReqTimeBlockContention','Time spent waiting for atomic access',
384,'ApproximateCPUTime','Approximate CPU time used','.4999168'
385,'ConnectedTime','Total time connections have been connected','1.26453958459968'
386,'CacheSizingStatistics','Display cache sizing statistics','No'
387,'ConsoleLogMaxSize','Console log maximum size','0'
388,'DebuggingInformation','Display debugging information','No'
389,'WebClientLogging','Enable HTTP web client procedure debug logging','No'
390,'WebClientLogFile','Set HTTP web client procedure debugging log filename',''
393,'IsRsaAvailable','RSA Available','Yes'
394,'MaxConnections','The maximum number of concurrent connections to the server','32766'
397,'FirstOption','First option property number','465'
398,'LastOption','Last option property number','602'
399,'LastConnectionProperty','Last connection property number','602'
400,'LastDatabaseProperty','Last database property number','446'

401,'LastServerProperty','Last server property number','446'
407,'QueryMemPercentOfCache','The amount of memory that is available for query execution algorithms, expressed as a percent of maximum cache size','50'
408,'QueryMemPages','The amount of memory that is available for query execution algorithms, expressed as a number of pages','14638692'
409,'QueryMemGrantBase','The minimum amount of memory granted to all requests','100'
410,'QueryMemGrantBaseMI','The minimum amount of memory granted to memory-intensive requests','29277'
411,'QueryMemGrantExtra','The number of query memory pages that can be distributed beyond the base memory-intensive grant','10991567'
412,'QueryMemActiveMax','Maximum number of requests actively using query memory','125'
418,'RemoteCapability','Remote capability',
419,'MaxRemoteCapability','Maximum remote capability','300'
420,'EventTypeName','Event type name','GrowDB'
421,'EventTypeDesc','Event type description','database file extended'
422,'MaxEventType','Maximum event type','16'
424,'IsPortableDevice','Is Portable Device','No'
425,'IPAddressMonitorPeriod','IP address monitor period','0'
428,'ServerEdition','Server edition','Advanced HighAvailability InMemory FIPS'
429,'ObjectType','Object type',
431,'CurrentMultiProgrammingLevel','The current number of tasks which the server can process concurrently','500'
432,'MinMultiProgrammingLevel','The minimum number of tasks which the server can process concurrently','20'
433,'MaxMultiProgrammingLevel','The maximum number of tasks which the server can process concurrently','2000'
434,'AutoMultiProgrammingLevel','The current state of automatic tuning of number of request tasks','0'
435,'AutoMultiProgrammingLevelStatistics','The current state of showing the statistics for automatically tuning the multiprogramming level','No'
440,'DiskSandbox','Disk Sandbox','Off'
442,'ProcessorAffinity','Process Affinity','0-19'
444,'CurrentMirrorBackgroundWorkers','The number of workers currently being used for database mirroring background tasks.','0'
445,'MaxMirrorBackgroundWorkers','The highest number of workers used for database mirroring background tasks.','0'
'
'DB Properties'
0,2,'BytesReceived','Bytes received by server','6176'
0,3,'BytesReceivedUncomp','Bytes received after decompression','6260'
0,4,'BytesSent','Bytes sent to client','54569'
0,5,'BytesSentUncomp','Bytes sent before compression','54569'
0,10,'CacheHits','Cache Hits','22161'
0,11,'CacheReadIndInt','Cache index interior reads','227'
0,12,'CacheReadIndLeaf','Cache index leaf reads','1133'
0,15,'CacheRead','Cache reads','22259'
0,19,'CacheReadTable','Cache table reads','1247'
0,20,'CacheReadWorkTable','Cache work table reads','4'
0,21,'CarverHeapPages','Cache pages used for carvers','0'
0,22,'ChkptFlush','Checkpoint flushed pages','2'
0,23,'ChkptPage','Checkpoint log page images saved','2'
0,24,'CheckpointUrgency','Checkpoint Urgency','0'
0,25,'Chkpt','Checkpoints','1'
0,26,'CheckpointLogCommitToDisk','Checkpoint log commit to disk','4'
0,27,'CheckpointLogPagesInUse','Checkpoint log pages in use','2'
0,28,'CheckpointLogPagesRelocated','Checkpoint log pages relocated','0'
0,29,'CheckpointLogSavePreimage','Checkpoint log save preimage','2'
0,30,'CheckpointLogSize','Checkpoint log size in pages','128'
0,31,'CheckpointLogPagesWritten','Checkpoint log pages written','2'
0,32,'CheckpointLogWrites','Checkpoint log disk writes','2'
0,33,'CleanablePagesAdded','Cleanable Pages Added','0'
0,34,'CleanablePagesCleaned','Cleanable Pages Cleaned','0'
0,35,'CleanableRowsAdded','Cleanable Rows Added','0'
0,36,'CleanableRowsCleaned','Cleanable Rows Cleaned','0'
0,37,'ClientStmtCacheHits','Number of prepares not required because of the client statement cache','0'
0,38,'ClientStmtCacheMisses','Number of prepares in the client statement cache which were prepared again','0'
0,39,'Commit','Number of commit requests','12'
0,40,'CommitFile','Commit writes to disk','22'
0,41,'ConnCount','Number of active connections','1'
0,42,'ConnPoolCachedCount','Number of connections disconnected by the application but cached for connection pooling','0'
0,43,'ConnPoolHits','Number of pooled connections which have been reused','0'
0,44,'ConnPoolMisses','Number of pooled connections which were cached but could not be reused','0'
0,46,'CurrIO','Active disk read/write requests','253'
0,47,'CurrRead','Active disk read requests','253'

0,48,'CurrWrite','Active disk write requests','0'
0,49,'Cursor','Declared cursors','4'
0,50,'DiskReadIndInt','Disk index interior reads','8'
0,51,'DiskReadIndLeaf','Disk index leaf reads','52'
0,52,'DiskRead','Disk reads','23943'
0,53,'DiskReadHint','Disk read hints','24091'
0,54,'DiskReadHintPages','Disk read hint pages','26085'
0,57,'DiskRetryReadScatter','Disk read retries for scattered reads','0'
0,59,'DiskSyncRead','Disk reads issued synchronously','105'
0,60,'DiskSyncWrite','Disk writes issued synchronously','1'
0,61,'DiskReadTable','Disk table reads','15114'
0,62,'DiskWaitRead','Number of times the server waited for an asynchronous read','841'
0,63,'DiskWaitWrite','Number of times the server waited for an asynchronous write','5'
0,64,'DiskReadWorkTable','Disk work table reads','0'
0,65,'DiskWrite','Disk writes','10'
0,66,'DiskWriteHint','Disk write hints','6'
0,67,'DiskWriteHintPages','Disk write hint pages','6'
0,70,'ExprCacheAbandons','Number of time that the expression cache was completely abandoned due to the hit rate being too low','0'
0,71,'ExprCacheDropsToReadOnly','Number of times that the expression cache dropped to read-only status due to the hit rate being low','0'
0,72,'ExprCacheEvicts','Number of evictions from the expression cache','0'
0,73,'ExprCacheHits','Number of hits in the expression cache','0'
0,74,'ExprCacheInserts','Number of values inserted into the expression cache','0'
0,75,'ExprCacheLookups','Number of lookups done in the expression cache','0'
0,76,'ExprCacheResumesOfReadWrite','Number of times that the expression cache resumed read-write status due to the hit rate rising again','0'
0,77,'ExprCacheStarts','Number of times the expression cache was started','0'
0,78,'ExtendDB','Extend database file writes','0'
0,79,'ExtendTempWrite','Extend temporary file writes','224'
0,81,'FullCompare','Number of comparisons beyond the hash value','0'
0,82,'GetData','GETDATA requests','4'
0,83,'HashForcedPartitions','Times that a hash operator was forced to partition due to competition for memory','0'
0,84,'HashRowsFiltered','Number of probe rows rejected by bit-vector filters','0'
0,85,'HashRowsPartitioned','Number of rows written to hash work tables','0'
0,86,'HashWorkTables','Number of work tables created for hash-based operations','0'
0,87,'HeapsCarver','Number of heaps used for carvers','0'
0,88,'HeapsLocked','Number of relocatable heaps currently locked in cache','6'
0,89,'HeapsQuery','Number of heaps used for query processing (hash and sort operations)','0'
0,90,'HeapsRelocatable','Number of relocatable heaps','19'
0,91,'HttpConnPoolCachedCount','Number of database connections cached for connection pooling by HTTP','0'
0,92,'HttpConnPoolHits','Number of reused database connections accessed by HTTP that were last used by the same service in the pool','0'
0,93,'HttpConnPoolMisses','Number of database connections that could not be accessed from the pool by HTTP','0'
0,94,'HttpConnPoolSteals','Number of reused database connections accessed by HTTP that were last used by a different service in the pool','0'
0,95,'IOToRecover','Recovery I/O Estimate','4'
0,96,'IdleCheck','Idle I/O checked','10'
0,97,'IdleChkTime','Idle I/O checkpoint time','0'
0,98,'IdleChkpt','Idle I/O checkpoints','0'
0,99,'IdleWrite','Idle I/O writes','0'
0,100,'IndAdd','Number of index insertions','0'
0,101,'IndLookup','Number of index lookups','0'
0,102,'LockCount','Number of locks','0'
0,103,'LockTablePages','Lock table pages','0'
0,110,'MaxIO','Maximum active disk read/write requests','256'
0,111,'MaxRead','Maximum active disk read requests','256'
0,112,'MaxWrite','Maximum active disk write requests','2'
0,114,'MirrorServerWaits','The number of times the server waited more than 500 ms when sending log pages to copy servers','0'
0,118,'CursorOpen','Open cursors','4'
0,119,'PacketsReceived','Packets received by server','63'
0,120,'PacketsReceivedUncomp','Packets received after decompression','63'
0,121,'PacketsSent','Packets sent to client','66'
0,122,'PacketsSentUncomp','Packets sent before compression','66'
0,123,'PageRelocations','Page relocations','0'
0,125,'ProcedurePages','Procedure relocatable heap pages','24'
0,126,'QueryBypassed','Number of requests optimized by the optimizer bypass','15'

0,127,'QueryBypassedCosted','Number of requests optimized by the optimizer bypass using costing','0'
0,128,'QueryBypassedHeuristic','Number of requests optimized by the optimizer bypass using heuristics','15'
0,129,'QueryBypassedOptimized','Number of requests initially processed by the optimizer bypass and subsequently fully optimized','0'
0,130,'QueryCachePages','Pages used to cache query plans','0'
0,131,'QueryCachedPlans','Query plans stored in cache','0'
0,132,'QueryDescribedBypass','Number of describes processed by the optimizer bypass','8'
0,133,'QueryDescribedOptimizer','Number of describes processed by the optimizer','1'
0,134,'QueryHeapPages','Cache pages used for query processing (hash and sort operations)','0'
0,135,'QueryJHToJNLOptUsed','Number of times a hash join was converted to a nested loop join','0'
0,136,'QueryLowMemoryStrategy','Low memory strategies used by queries','0'
0,137,'QueryMemActiveCurr','The current number of requests actively using query memory','0'
0,140,'QueryMemGrantFailed','The total number of times any request waited for query memory and failed to get it','0'
0,141,'QueryMemGrantGranted','The number of pages currently granted to requests','0'
0,142,'QueryMemGrantRequested','The total number of times any request attempted to acquire query memory','0'
0,143,'QueryMemGrantWaited','The total number of times any request waited for query memory','0'
0,144,'QueryMemGrantWaiting','The current number of requests waiting for query memory','0'
0,145,'QueryOpened','Number of requests opened for execution','19'
0,146,'QueryOptimized','Number of requests that have been fully optimized','6'
0,147,'QueryReused','Number of requests that have been reused from the plan cache','0'
0,148,'QueryRowsFetched','Number of rows fetched using buffering','17249'
0,149,'QueryRowsMaterialized','Number of rows written to work tables during query processing','486'
0,150,'RecoveryUrgency','Recovery Urgency','0'
0,151,'RecursiveIterations','Number of iterations for recursive unions','0'
0,152,'RecursiveIterationsHash','Number of times recursive hash join used hash strategy','0'
0,153,'RecursiveIterationsNested','Number of times recursive hash join used nested loop strategy','0'
0,154,'RecursiveJNLMisses','Number of index probe cache misses for recursive hash join','0'
0,155,'RecursiveJNLProbes','Number of times recursive hash join attempted an index probe','0'
0,156,'LogFreeCommit','Transaction log group commits','0'
0,157,'LogWrite','Transaction log page writes','0'
0,158,'RelocatableHeapPages','Relocatable heap pages','75'
0,161,'RequestsReceived','Requests received by server','63'
0,162,'Rlbk','Rollback requests handled','0'
0,163,'RollbackLogPages','Rollback log pages','0'
0,165,'SnapshotCount','Number of active snapshots','0'
0,166,'SortMergePasses','Number of merge passes used during sorting','0'
0,167,'SortRowsMaterialized','Number of rows written to sort work tables','0'
0,168,'SortRunsWritten','Number of sorted runs written during sorting','0'
0,169,'SortSortedRuns','Number of sorted runs created during run formation','0'
0,170,'SortWorkTables','Number of work tables created for sorting','0'
0,171,'PrepStmt','Prepared statements','6'
0,172,'StatementDescribes','Number of statement describes','34'
0,173,'StatementPostAnnotates','Number of statements processed by semantic query transformations','19'
0,174,'StatementPostAnnotatesSimple','Number of statements processed by semantic query transformations with simple transforms','0'
0,175,'StatementPostAnnotatesSkipped','Number of statements that have skipped semantic query transformations','13'
0,176,'Prepares','Number of statement prepares','23'
0,179,'TempTablePages','Temporary table pages','4'
0,183,'TriggerPages','Trigger relocatable heap pages','0'
0,186,'UserDefinedCounterRate01','A rate counter that is set by the database application (counter 1)','0'
0,187,'UserDefinedCounterRate02','A rate counter that is set by the database application (counter 2)','0'
0,188,'UserDefinedCounterRate03','A rate counter that is set by the database application (counter 3)','0'
0,189,'UserDefinedCounterRate04','A rate counter that is set by the database application (counter 4)','0'
0,190,'UserDefinedCounterRate05','A rate counter that is set by the database application (counter 5)','0'
0,191,'UserDefinedCounterRaw01','A counter that is set by the database application (counter 1)','0'
0,192,'UserDefinedCounterRaw02','A counter that is set by the database application (counter 2)','0'
0,193,'UserDefinedCounterRaw03','A counter that is set by the database application (counter 3)','0'
0,194,'UserDefinedCounterRaw04','A counter that is set by the database application (counter 4)','0'
0,195,'UserDefinedCounterRaw05','A counter that is set by the database application (counter 5)','0'
0,196,'VersionStorePages','Version store pages','0'
0,197,'ViewPages','View relocatable heap pages','5'
0,198,'XPathCompiles','XPath Compiles','0'
0,199,'Name','Name','tpcc'
0,200,'Alias','Mounted database name','tpcc'
0,201,'File','Database file','d:\\tpcc_data\\tpcc.db'
0,202,'PageSize','Database page size','4096'
0,203,'LogName','Database log file name','c:\\tpcc_data\\tpcc.log'

0,204,'LogMirrorName','Database log mirror file name',
0,205,'TempFileName','Database temporary file name','C:\\Users\\ADMINI~1\\AppData\\Local\\Temp\\sqla0000.tmp'
0,229,'IQStore','IQ store is on/off','Off'
0,236,'ConnsDisabled','Connections disabled','Off'
0,248,'CharSet','Character set used for CHAR data','windows-1252'
0,249,'NcharCharSet','Character set used for NCHAR data','UTF-8'
0,250,'MultiByteCharSet','Multi Byte Character Set (on/off)','Off'
0,251,'Language','Language','nl,da,it,pt,de,fr,en'
0,252,'Collation','Name of collation used for CHAR data','1252LATIN1'
0,253,'NcharCollation','Name of collation used for NCHAR data','UCA'
0,254,'ReadOnly','Database read-only mode','Off'
0,257,'LTMTrunc','LTM truncation point','18446744073709551615'
0,258,'RemoteTrunc','SQL Remote truncation point','18446744073709551615'
0,259,'SyncTrunc','DBMLSync truncation point','18446744073709551615'
0,260,'GlobalDBID','Global database identifier','2147483647'
0,261,'CurrentRedoPos','Current redo position','58763183698'
0,262,'LastCommitRedoPos','Redo position at end of last commit','58763183693'
0,263,'LastWrittenRedoPos','Last redo position written to disk','58763183693'
0,264,'LastSyncedRedoPos','Last redo position written and synchronized to disk','58763183693'
0,265,'LTMGeneration','LTM generation number','0'
0,266,'ProcedureProfiling','Procedure profiling','Off'
0,278,'CaseSensitive','Case sensitivity','Off'
0,279,'AccentSensitive','Accent sensitivity for UCA collations','Off'
0,280,'BlankPadding','Blank padding','Off'
0,283,'Checksum','Page checksum','On'
0,287,'CatalogCollation','Name of collation used for catalog data','1252LATIN1'
0,289,'DriveType','Drive type where the database is located','FIXED'
0,290,'DriveBus','Bus type where the database is located','RAID'
0,291,'DriveModel','Model of drive where the database is located','DELLPERC_H7103.13'
0,305,'Encryption','Encryption type','None'
0,306,'Capabilities','Database capability bits','E00C3000D000FEDFF7FDD'
0,308,'DBFileFragments','Database file fragments','2'
0,309,'LogFileFragments','Log file fragments','237'
0,329,'AuditingTypes','Auditing types enabled','all'
0,332,'FreePages','Number of free pages in dbspace','6698'
0,333,'IOParallelism','Estimated number of simultaneous I/O operations supported by dbspace','1'
0,334,'FileSize','File size in pages','14611'
0,348,'NextScheduleTime','Next schedule time',
0,357,'IdentitySignature','Identity signature','218074730'
0,359,'CopyNodeParent','Copy node parent',
0,360,'PartnerState','Partner state',
0,361,'ArbiterState','Arbiter state',
0,362,'MirrorState','Mirror state',
0,363,'MirrorServerState','Mirror server state',
0,364,'MirrorMode','Mirror mode',
0,365,'MirrorRole','Mirror role',
0,367,'AlternateServerName','Alternate server name',
0,368,'AlternateMirrorServerName','Alternate mirror server name',
0,369,'SnapshotIsolationState','State of snapshot isolation','Off'
0,370,'SendingTracingTo','Database where diagnostic tracing is being sent',''
0,371,'ReceivingTracingFrom','Database from which diagnostic tracing is being received',''
0,374,'ReqCountUnscheduled','Number of times waited for scheduling',
0,375,'ReqCountActive','Number of requests processed',
0,376,'ReqCountBlockIO','Number of times waited for I/O to complete','104'
0,377,'ReqCountBlockLock','Number of times waited for a lock',
0,378,'ReqCountBlockContention','Number of times waited for atomic access',
0,379,'ReqTimeUnscheduled','Time spent unscheduled',
0,380,'ReqTimeActive','Time spent processing requests',
0,381,'ReqTimeBlockIO','Time spent waiting for I/O to complete','.013801484719448'
0,382,'ReqTimeBlockLock','Time spent waiting for a lock',
0,383,'ReqTimeBlockContention','Time spent waiting for atomic access',
0,384,'ApproximateCPUTime','Approximate CPU time used','.546816'
0,385,'ConnectedTime','Total time connections have been connected','2.35125853811987'
0,391,'EncryptionScope','Scope of encryption','None'
0,395,'JavaVM','The JAVA VM that will be used to run JAVA classes','C:\\Program Files\\SQL Anywhere
16\\Bin64\\jre170\\bin\\java.exe'
0,396,'Authenticated','Returns whether a database or connection is authenticated','No'

0,402,'DatabaseCleaner','State of the Database Cleaner','On'
0,403,'HasCollationTailoring','Has collation tailoring','Off'
0,406,'HasEndianSwapFix','Fixed endian swap implementation','On'
0,415,'OptionWatchList','Option watch list',''
0,416,'OptionWatchAction','Option watch action','message'
0,417,'HasNCHARLegacyCollationFix','Fixed NCHAR legacy collation implementation','On'
0,430,'SynchronizationSchemaChangeActive','Synchronization schema change active','Off'
0,436,'LastCheckpointTime','Last checkpoint time','2014-10-29 09:02:15.517'
0,437,'HasTornWriteFix','Fixed file format to allow recovery from partial writes','Off'
0,438,'WriteChecksum','Write I/O checksum','On'
0,440,'DiskSandbox','Disk Sandbox','Off'
0,441,'UTCTimestampCatalog','Timestamps stored in database catalog tables are UTC based','On'
0,443,'TimeWithoutClientConnection','The elapsed time in seconds since a CmdSeq or TDS connection existed','0'
0,446,'BackupInProgress','Indicates whether the database is currently being backed up.','Off'
'
'Default DBA Connection Properties'
1,2,'BytesReceived','Bytes received by server','7340'
1,3,'BytesReceivedUncomp','Bytes received after decompression','7424'
1,4,'BytesSent','Bytes sent to client','101451'
1,5,'BytesSentUncomp','Bytes sent before compression','101451'
1,10,'CacheHits','Cache Hits','2339'
1,11,'CacheReadIndInt','Cache index interior reads','97'
1,12,'CacheReadIndLeaf','Cache index leaf reads','752'
1,15,'CacheRead','Cache reads','2354'
1,19,'CacheReadTable','Cache table reads','425'
1,20,'CacheReadWorkTable','Cache work table reads','10'
1,21,'CarverHeapPages','Cache pages used for carvers','0'
1,37,'ClientStmtCacheHits','Number of prepares not required because of the client statement cache','0'
1,38,'ClientStmtCacheMisses','Number of prepares in the client statement cache which were prepared again','0'
1,39,'Commit','Number of commit requests','1'
1,49,'Cursor','Declared cursors','4'
1,50,'DiskReadIndInt','Disk index interior reads','0'
1,51,'DiskReadIndLeaf','Disk index leaf reads','3'
1,52,'DiskRead','Disk reads','15'
1,53,'DiskReadHint','Disk read hints','0'
1,54,'DiskReadHintPages','Disk read hint pages','0'
1,59,'DiskSyncRead','Disk reads issued synchronously','15'
1,60,'DiskSyncWrite','Disk writes issued synchronously','0'
1,61,'DiskReadTable','Disk table reads','10'
1,62,'DiskWaitRead','Number of times the server waited for an asynchronous read','0'
1,63,'DiskWaitWrite','Number of times the server waited for an asynchronous write','0'
1,64,'DiskReadWorkTable','Disk work table reads','0'
1,65,'DiskWrite','Disk writes','0'
1,66,'DiskWriteHint','Disk write hints','0'
1,67,'DiskWriteHintPages','Disk write hint pages','0'
1,70,'ExprCacheAbandons','Number of time that the expression cache was completely abandoned due to the hit rate being too low','0'
1,71,'ExprCacheDropsToReadOnly','Number of times that the expression cache dropped to read-only status due to the hit rate being low','0'
1,72,'ExprCacheEvicts','Number of evictions from the expression cache','0'
1,73,'ExprCacheHits','Number of hits in the expression cache','0'
1,74,'ExprCacheInserts','Number of values inserted into the expression cache','0'
1,75,'ExprCacheLookups','Number of lookups done in the expression cache','0'
1,76,'ExprCacheResumesOfReadWrite','Number of times that the expression cache resumed read-write status due to the hit rate rising again','0'
1,77,'ExprCacheStarts','Number of times the expression cache was started','0'
1,81,'FullCompare','Number of comparisons beyond the hash value','0'
1,82,'GetData','GETDATA requests','6'
1,83,'HashForcedPartitions','Times that a hash operator was forced to partition due to competition for memory','0'
1,84,'HashRowsFiltered','Number of probe rows rejected by bit-vector filters','0'
1,85,'HashRowsPartitioned','Number of rows written to hash work tables','0'
1,86,'HashWorkTables','Number of work tables created for hash-based operations','0'
1,87,'HeapsCarver','Number of heaps used for carvers','0'
1,88,'HeapsLocked','Number of relocatable heaps currently locked in cache','7'
1,89,'HeapsQuery','Number of heaps used for query processing (hash and sort operations)','0'
1,90,'HeapsRelocatable','Number of relocatable heaps','22'
1,100,'IndAdd','Number of index insertions','0'

1,101,'IndLookup','Number of index lookups','0'
1,102,'LockCount','Number of locks','0'
1,118,'CursorOpen','Open cursors','4'
1,119,'PacketsReceived','Packets received by server','85'
1,120,'PacketsReceivedUncomp','Packets received after decompression','86'
1,121,'PacketsSent','Packets sent to client','92'
1,122,'PacketsSentUncomp','Packets sent before compression','92'
1,126,'QueryBypassed','Number of requests optimized by the optimizer bypass','21'
1,127,'QueryBypassedCosted','Number of requests optimized by the optimizer bypass using costing','0'
1,128,'QueryBypassedHeuristic','Number of requests optimized by the optimizer bypass using heuristics','21'
1,129,'QueryBypassedOptimized','Number of requests initially processed by the optimizer bypass and subsequently fully optimized','0'
1,130,'QueryCachePages','Pages used to cache query plans','0'
1,131,'QueryCachedPlans','Query plans stored in cache','0'
1,132,'QueryDescribedBypass','Number of describes processed by the optimizer bypass','10'
1,133,'QueryDescribedOptimizer','Number of describes processed by the optimizer','1'
1,134,'QueryHeapPages','Cache pages used for query processing (hash and sort operations)','0'
1,135,'QueryJHToJNLOptUsed','Number of times a hash join was converted to a nested loop join','0'
1,136,'QueryLowMemoryStrategy','Low memory strategies used by queries','0'
1,137,'QueryMemActiveCurr','The current number of requests actively using query memory','0'
1,140,'QueryMemGrantFailed','The total number of times any request waited for query memory and failed to get it','0'
1,141,'QueryMemGrantGranted','The number of pages currently granted to requests','0'
1,142,'QueryMemGrantRequested','The total number of times any request attempted to acquire query memory','0'
1,143,'QueryMemGrantWaited','The total number of times any request waited for query memory','0'
1,144,'QueryMemGrantWaiting','The current number of requests waiting for query memory','0'
1,145,'QueryOpened','Number of requests opened for execution','25'
1,146,'QueryOptimized','Number of requests that have been fully optimized','8'
1,147,'QueryReused','Number of requests that have been reused from the plan cache','0'
1,148,'QueryRowsFetched','Number of rows fetched using buffering','824'
1,149,'QueryRowsMaterialized','Number of rows written to work tables during query processing','1124'
1,151,'RecursiveIterations','Number of iterations for recursive unions','0'
1,152,'RecursiveIterationsHash','Number of times recursive hash join used hash strategy','0'
1,153,'RecursiveIterationsNested','Number of times recursive hash join used nested loop strategy','0'
1,154,'RecursiveJNLMisses','Number of index probe cache misses for recursive hash join','0'
1,155,'RecursiveJNLProbes','Number of times recursive hash join attempted an index probe','0'
1,156,'LogFreeCommit','Transaction log group commits','0'
1,157,'LogWrite','Transaction log page writes','0'
1,161,'RequestsReceived','Requests received by server','86'
1,162,'Rlbk','Rollback requests handled','0'
1,163,'RollbackLogPages','Rollback log pages','0'
1,165,'SnapshotCount','Number of active snapshots','0'
1,166,'SortMergePasses','Number of merge passes used during sorting','0'
1,167,'SortRowsMaterialized','Number of rows written to sort work tables','0'
1,168,'SortRunsWritten','Number of sorted runs written during sorting','0'
1,169,'SortSortedRuns','Number of sorted runs created during run formation','0'
1,170,'SortWorkTables','Number of work tables created for sorting','0'
1,171,'PrepStmt','Prepared statements','7'
1,172,'StatementDescribes','Number of statement describes','42'
1,173,'StatementPostAnnotates','Number of statements processed by semantic query transformations','27'
1,174,'StatementPostAnnotatesSimple','Number of statements processed by semantic query transformations with simple transforms','0'
1,175,'StatementPostAnnotatesSkipped','Number of statements that have skipped semantic query transformations','17'
1,176,'Prepares','Number of statement prepares','27'
1,178,'TempFilePages','Number of temporary file pages used by the connection','91'
1,179,'TempTablePages','Temporary table pages','4'
1,186,'UserDefinedCounterRate01','A rate counter that is set by the database application (counter 1)','0'
1,187,'UserDefinedCounterRate02','A rate counter that is set by the database application (counter 2)','0'
1,188,'UserDefinedCounterRate03','A rate counter that is set by the database application (counter 3)','0'
1,189,'UserDefinedCounterRate04','A rate counter that is set by the database application (counter 4)','0'
1,190,'UserDefinedCounterRate05','A rate counter that is set by the database application (counter 5)','0'
1,191,'UserDefinedCounterRaw01','A counter that is set by the database application (counter 1)','0'
1,192,'UserDefinedCounterRaw02','A counter that is set by the database application (counter 2)','0'
1,193,'UserDefinedCounterRaw03','A counter that is set by the database application (counter 3)','0'
1,194,'UserDefinedCounterRaw04','A counter that is set by the database application (counter 4)','0'
1,195,'UserDefinedCounterRaw05','A counter that is set by the database application (counter 5)','0'
1,199,'Name','Name','SQL_DBC_70f4e4e1f0'
1,206,'DBNumber','Database number','0'

1,207,'UserID','User ID','DBA'
1,208,'LastReqTime','Last request time','2014-10-29 09:02:38.722'
1,209,'ReqType','Type of active request','PREFETCH'
1,210,'CommLink','Communication link','local'
1,211,'NodeAddress','Client node address',''
1,212,'ClientNodeAddress','Client node address',''
1,213,'ServerNodeAddress','Server node address',''
1,215,'Number','Connection ID','1'
1,216,'LastIdle','Ticks between requests','57'
1,217,'BlockedOn','Connection blocked on','0'
1,218,'WaitStartTime','Time connection started waiting at',''
1,219,'WaitType','Reason for connection wait',''
1,220,'LockName','Lock name blocked on','0'
1,221,'UncommitOp','Uncommitted operations','0'
1,243,'LivenessTimeout','Client liveness timeout default','0'
1,246,'CommProtocol','Communication protocol name','CmdSeq'
1,247,'ClientLibrary','Client library name','CmdSeq'
1,248,'CharSet','Character set used for CHAR data','windows-1252'
1,249,'NcharCharSet','Character set used for NCHAR data','UTF-8'
1,251,'Language','Language','us_english'
1,262,'LastCommitRedoPos','Redo position at end of last commit','58763183693'
1,267,'TimeZoneAdjustment','Time zone adjustment from UTC time in minutes','-240'
1,268,'EventName','Name of the event this connection is handling',''
1,281,'Compression','Compression enabled','Off'
1,303,'AppInfo','Application Information','IP=10.7.161.230;HOST=EINSTEIN;OSUSER=dbtest;OS="Windows 2012R2 Build 9600
";EXE="C:\\Program Files\\SQL Anywhere
16\\bin64\\dbisql.com";PID=0x18a8;THREAD=0x197c;VERSION=16.0.0.2020;API=iAnywhereJDBC;TIMEZONEADJUSTMENT=-240'
1,304,'UserAppInfo','User application information',''
1,305,'Encryption','Encryption type','None'
1,307,'UtilCmdsPermitted','Utility commands permitted','On'
1,310,'PacketSize','Packet size','7300'
1,311,'LastStatement','Last statement',''
1,314,'LastPlanText','Last text plan',''
1,315,'CurrentLineNumber','Current line number',''
1,320,'IdleTimeout','Idle timeout','0'
1,325,'TransactionStartTime','Transaction start time',''
1,328,'CurrentProcedure','Stored procedure currently being executed',''
1,331,'CommNetworkLink','Communication link (never local)','SharedMemory'
1,335,'ClientPort','Client TCP/IP port number','0'
1,336,'ServerPort','Server TCP/IP port number','0'
1,341,'HttpServiceName','Http service name',''
1,347,'LoginTime','Login time','2014-10-29 09:02:35.472'
1,349,'MessageReceived','Message received',''
1,350,'SessionTimeout','HTTP session timeout in minutes','0'
1,351,'SessionID','HTTP session ID',''
1,352,'SessionCreateTime','HTTP session create time',''
1,353,'SessionLastTime','HTTP session last request time',''
1,373,'ReqStatus','Current request status','Executing'
1,374,'ReqCountUnscheduled','Number of times waited for scheduling',
1,375,'ReqCountActive','Number of requests processed',
1,376,'ReqCountBlockIO','Number of times waited for I/O to complete','15'
1,377,'ReqCountBlockLock','Number of times waited for a lock',
1,378,'ReqCountBlockContention','Number of times waited for atomic access',
1,379,'ReqTimeUnscheduled','Time spent unscheduled',
1,380,'ReqTimeActive','Time spent processing requests',
1,381,'ReqTimeBlockIO','Time spent waiting for I/O to complete','.001526990207274'
1,382,'ReqTimeBlockLock','Time spent waiting for a lock',
1,383,'ReqTimeBlockContention','Time spent waiting for atomic access',
1,384,'ApproximateCPUTime','Approximate CPU time used','.1093632'
1,385,'ConnectedTime','Total time connections have been connected','3.266'
1,392,'LockTableOID','Object ID of locked table','0'
1,396,'Authenticated','Returns whether a database or connection is authenticated','No'
1,404,'LockRowID','Identifier of the locked row','0'
1,405,'LockIndexID','Identifier of the locked index',''
1,413,'AuthType','The authentication type used when establishing the connection','Standard'
1,414,'OSUser','The operating system user name associated with the client process','dbtest'
1,423,'IsDebugger','Is debugger connection','No'

1,426,'Progress','Progress',
1,427,'ParentConnection','Temporary connection''s parent connection',
1,439,'NumLocalTempTables','The number of local temporary tables in use by the connection','4'
1,465,'blocking','Controls response to locking conflicts','On'
1,466,'blocking_timeout','Controls the time a transaction waits to obtain a lock','0'
1,467,'blocking_others_timeout','Controls the time a transaction can block another connection','0'
1,468,'checkpoint_time','Maximum number of minutes between checkpoints','59'
1,469,'conversion_error','Controls datatype conversion errors','On'
1,470,'date_format','Controls format for DATE values','YYYY-MM-DD'
1,471,'date_order','Controls order of date components','YMD'
1,472,'isolation_level','Controls the locking isolation level','0'
1,473,'updatable_statement_isolation','The isolation level for updatable statements when using readonly-statement-snapshot','0'
1,474,'lock_rejected_rows','Reserved','Off'
1,475,'login_procedure','Procedure to be run during login','sp_login_environment'
1,476,'on_tsql_error','Controls error handling in stored procedures','Conditional'
1,477,'precision','Maximum number of digits in decimal arithmetic','30'
1,478,'recovery_time','Maximum time to allow for database recovery','9999'
1,479,'replicate_all','Allows entire database to act as primary site for Rep Server','Off'
1,480,'row_counts','Controls whether row counts are estimates or exact','Off'
1,481,'scale','Minimum number of digits after decimal point','6'
1,482,'timestamp_format','Controls format for TIMESTAMP values','YYYY-MM-DD HH:NN:SS.SSS'
1,483,'timestamp_with_time_zone_format','Controls format for TIMESTAMP WITH TIME ZONE values','YYYY-MM-DD
HH:NN:SS.SSS+HH:NN'
1,484,'time_format','Controls format for TIME values','HH:NN:SS.SSS'
1,485,'wait_for_commit','Controls when foreign key integrity is checked','Off'
1,486,'quoted_identifier','Controls interpretation of strings enclosed in double quotes','On'
1,487,'allow_nulls_by_default','Controls NULL values for new columns','On'
1,488,'cooperative_commits','Controls when COMMITs are written to disk','On'
1,489,'cooperative_commit_timeout','Controls delay before a COMMIT is written to disk','250'
1,490,'delayed_commits','Controls when server returns control to application following a COMMIT','Off'
1,491,'delayed_commit_timeout','Controls delay before server returns control while waiting to do COMMIT','500'
1,492,'non_keywords','Controls what identifiers are keywords',''
1,493,'reserved_keywords','Controls what non-default reserved keywords are enabled',''
1,494,'sql_flagger_error_level','Controls errors for SQL that is not from specified set of SQL/92','Off'
1,495,'sql_flagger_warning_level','Controls warnings for SQL that is not from specified set of SQL/92','Off'
1,496,'ansi_blanks','Controls truncation errors','Off'
1,497,'string_rtruncation','Controls truncation errors on INSERT or UPDATE','On'
1,498,'divide_by_zero_error','Controls divide-by-zero errors','On'
1,499,'ansinull','Controls interpretation of NULL values','On'
1,500,'ansi_permissions','Controls permissions checking for DELETE and UPDATE statements','On'
1,501,'close_on_endtrans','Controls closing of cursors at end of transaction','On'
1,502,'tsql_variables','Controls whether @ can be used as host variable name prefix','Off'
1,503,'chained','Controls transaction mode if BEGIN TRANSACTION not used','On'
1,504,'nearest_century','Controls interpretation of two-digit years','50'
1,505,'fire_triggers','Controls whether triggers are fired in the database','On'
1,506,'background_priority','Controls priority of current connection','Off'
1,507,'login_mode','Controls integrated and Kerberos logins','Standard'
1,508,'extern_login_credentials','Controls whether remote connections will be attempted using the logged in user''s extern login
credentials or the effective user''s extern login credentials','Effective_user'
1,509,'integrated_server_name','Server name for determining user groups',''
1,510,'connection_authentication','Authentication string for
connection','Company=Sybase;Application=DBTools;Signature=000fa55157edb8e14d818eb4fe3db41447146f1571g2a1b5949cab32c
7760419117ca3ce88770fecfd7'
1,511,'java_main_userid','Userid used by external java vm to connect to the database',''
1,512,'java_location','File path pointing to the external java vm',''
1,513,'java_vm_options','Command line options used to launch external java vm',''
1,514,'java_class_path','Additional jars and directories added to the class path when launching the external java vm',''
1,515,'suppress_tds_debugging','Controls display of TDS debug information','On'
1,516,'upgrade_database_capability','Reserved',''
1,517,'database_authentication','Authentication string for database',''
1,518,'default_timestamp_increment','Number of microseconds to add to TIMESTAMP for next value','1'
1,519,'escape_character','Reserved','On'
1,520,'prefetch','Controls prefetching of rows','Conditional'
1,521,'continue_after_raiserror','Controls behavior following a RAISERROR statement','On'
1,522,'cis_option','Controls display of debug information for remote data access','0'
1,523,'cis_rowset_size','Controls number of fetched rows returned from remote servers','50'
1,524,'ansi_close_cursors_on_rollback','Controls whether WITH HOLD cursors are closed on ROLLBACK','Off'

1,525,'max_statement_count','Maximum number of prepared statements for a connection','50'
1,526,'max_cursor_count','Maximum number of cursors allowed for a connection','50'
1,527,'min_password_length','Minimum length for new database passwords','3'
1,528,'auditing','Enables and disables auditing','Off'
1,529,'conn_auditing','Enables and disables auditing on a connection','On'
1,530,'auditing_options','Reserved','4294967295'
1,531,'tds_empty_string_is_null','Controls whether TDS connections return empty strings as NULL or one space','Off'
1,532,'optimization_level','Controls amount of effort made by the query optimizer to find an access plan','9'
1,533,'extended_join_syntax','Controls errors when using duplicate correlation names in joins','On'
1,534,'ansi_update_constraints','Controls the range of updates that are permitted','Cursors'
1,535,'optimization_goal','Optimize for first row or all rows','All-rows'
1,536,'global_database_id','Controls initial value for DEFAULT GLOBAL AUTOINCREMENT columns','2147483647'
1,537,'max_hash_size','Deprecated','10'
1,538,'prevent_article_pkey_update','Controls updates to primary keys used in publications','On'
1,539,'truncate_timestamp_values','Controls precision of TIMESTAMP values','Off'
1,540,'return_date_time_as_string','Controls how DATE, TIME and TIMESTAMP values are fetched','On'
1,541,'first_day_of_week','Sets the numbering of the days of the week','7'
1,542,'preserve_source_format','Controls preservation of source for procedures, triggers, views, events','On'
1,543,'time_zone_adjustment','Numbers of minutes to add to UTC for this time zone','-240'
1,544,'exclude_operators','Reserved',''
1,545,'user_estimates','Controls whether to respect user estimates','Override-magic'
1,546,'max_plans_cached','Maximum number of cached execution plans for a connection','100'
1,547,'sort_collation','Controls implicit use of SORTKEY on ORDER BY of characters','Internal'
1,548,'pinned_cursor_percent_of_cache','Controls amount of server cache to be used for pinning cursors','10'
1,549,'on_charset_conversion_failure','Controls error handling for character set conversion failure','Ignore'
1,550,'update_statistics','Controls collection of statistics during query execution','On'
1,551,'collect_statistics_on_dml_updates','Controls collection of statistics during INSERT/UPDATE/DELETE statement execution','On'
1,552,'max_recursive_iterations','Maximum number of recursions for common table expressions','100'
1,553,'max_query_tasks','Maximum number of tasks that may be used by a parallel execution plan for a single query','0'
1,554,'for_xml_null_treatment','Controls treatment of NULL values in queries that use FOR XML','Omit'
1,555,'subsume_row_locks','Controls when server acquires row locks for table','On'
1,556,'read_past_deleted','Controls server behavior on uncommitted deletes','On'
1,557,'temp_space_limit_check','Controls whether a connection''s temp file usage is checked against an upper limit','On'
1,558,'force_view_creation','Controls errors when CREATE VIEW references non-existent view','Off'
1,559,'dedicated_task','Dedicates a server task to the current connection','Off'
1,560,'debug_messages','Controls whether MESSAGE ... DEBUG ONLY statements are executed','Off'
1,561,'optimization_workload','Controls whether optimizing for OLAP or mixed queries','Mixed'
1,562,'odbc_distinguish_char_and_varchar','Controls whether ODBC distinguishes CHAR and VARCHAR columns','Off'
1,563,'remote_idle_timeout','Controls the time that an HTTP procedure will wait for a response from a server','15'
1,564,'ansi_substring','Controls behavior of substring function with negative start or length parameter','On'
1,565,'odbc_describe_binary_as_varbinary','Controls whether ODBC describes BINARY columns as VARBINARY','Off'
1,566,'rollback_on_deadlock','Controls whether or not a transaction does a rollback upon deadlock','On'
1,567,'log_deadlocks','Controls whether deadlocks are logged','Off'
1,568,'webservice_namespace_host','Server hostname specification for use in web services namespace',''
1,569,'webservice_sessionid_name','Specifies the identifier name for an HTTP session, default is SessionID','SessionID'
1,570,'http_session_timeout','Timeout setting for an HTTP session','30'
1,571,'http_connection_pool_basesize','HTTP connection pool base size','10'
1,572,'http_connection_pool_timeout','HTTP connection pool timeout','60'
1,573,'request_timeout','Controls the maximum time a request is allowed to execute','0'
1,574,'synchronize_mirror_on_commit','Controls whether database mirror server is synchronized on commit','Off'
1,575,'uuid_has_hyphens','Controls format for UUID values','On'
1,576,'verify_password_function','Function to be run to verify a new password conforms to password rules',''
1,577,'allow_snapshot_isolation','Controls enabling of snapshot isolation','Off'
1,578,'default_dbspace','Sets the default dbspace for table creation',''
1,579,'oem_string','Sets the OEM string in the database file header',''
1,580,'max_temp_space','Sets the maximum temp space that a connection may use','0'
1,581,'secure_feature_key','Sets the secure feature override key',''
1,582,'materialized_view_optimization','Controls the use of materialized views during query optimization','Stale'
1,583,'tsql_outer_joins','Controls whether TSQL outer joins can be used in DML statements','Off'
1,584,'post_login_procedure','Procedure to be called by the application when connecting to return
messages','dbo.sa_post_login_procedure'
1,585,'max_client_statements_cached','Maximum number of prepared statements cached by the client for a connection','10'
1,586,'query_mem_timeout','Controls the time a transaction waits to obtain query memory for query execution algorithms','-1'
1,587,'allow_read_client_file','Controls whether the use of reading data from the client machine is allowed','Off'
1,588,'allow_write_client_file','Controls whether the use of writing data to the client machine is allowed','Off'
1,589,'priority','Controls the priority level of current connection','normal'
1,590,'max_priority','Controls the maximum priority level a connection can have','normal'

1,591,'progress_messages','Controls the type of progress messages sent to the client on certain SQL statements','Formatted'
1,592,'st_geometry_describe_type','Controls how ST_Geometry values are described to the client (as CHAR, NCHAR, or BINARY)','char'
1,593,'st_geometry_astext_format','Specifies the default format to use in ST_Geometry::ST_AsText()','WKT'
1,594,'st_geometry_asbinary_format','Specifies the default format to use in ST_Geometry::ST_AsBinary()','WKB'
1,595,'st_geometry_asxml_format','Specifies the default format to use in ST_Geometry::ST_AsXML()','GML'
1,596,'st_geometry_on_invalid','Controls error handling for invalid geometry values','Error'
1,597,'st_geometry_interpolation','Specifies options to use when converting from an ST_CircularString to an ST_LineString','relative-tolerance-percent=.3'
1,598,'min_role_admins','Controls minimum number of administrators for a role','1'
1,599,'trusted_certificates_file','Specifies the file containing certificates for trusted Certificate Authorities used with outbound TLS connections from SA or IQ.',''
1,600,'db_publisher','Specifies publisher user for the database','-1'
1,601,'auto_commit_on_create_local_temp_index','Causes auto commit on creation of local temporary index','Off'
1,602,'disk_sandbox','Controls database disk sandbox','Off'
'

DB Files'
0,0,'system','d:\\tpcc_data\\tpcc.db',
15,15,'temporary', c:\\users\\administrator\\AppData\\Local\\Temp\\sqla0000.tmp',
1,1,'cs_space', 'd:\\tpcc_data\\cs.dbs',
2,2,'misc_space', 'd:\\tpcc_data\\misc.dbs',
'

Options'
'PUBLIC','blocking','On'
'PUBLIC','blocking_timeout','0'
'PUBLIC','blocking_others_timeout','0'
'PUBLIC','checkpoint_time','60'
'PUBLIC','conversion_error','On'
'PUBLIC','date_format','YYYY-MM-DD'
'PUBLIC','date_order','YMD'
'PUBLIC','isolation_level','0'
'PUBLIC','updatable_statement_isolation','0'
'PUBLIC','lock_rejected_rows','Off'
'PUBLIC','login_procedure','sp_login_environment'
'PUBLIC','on_tsql_error','Conditional'
'PUBLIC','precision','30'
'PUBLIC','recovery_time','2'
'PUBLIC','replicate_all','Off'
'PUBLIC','row_counts','Off'
'PUBLIC','scale','6'
'PUBLIC','timestamp_format','YYYY-MM-DD HH:NN:SS.SSS'
'PUBLIC','timestamp_with_time_zone_format','YYYY-MM-DD HH:NN:SS.SSS+HH:NN'
'PUBLIC','time_format','HH:NN:SS.SSS'
'PUBLIC','wait_for_commit','Off'
'PUBLIC','quoted_identifier','On'
'PUBLIC','allow_nulls_by_default','On'
'PUBLIC','cooperative_commits','On'
'PUBLIC','cooperative_commit_timeout','250'
'PUBLIC','delayed_commits','Off'
'PUBLIC','delayed_commit_timeout','500'
'PUBLIC','non_keywords',''
'PUBLIC','reserved_keywords',''
'PUBLIC','sql_flagger_error_level','W'
'PUBLIC','sql_flagger_warning_level','W'
'PUBLIC','ansi_blanks','Off'
'PUBLIC','string_rtruncation','On'
'PUBLIC','divide_by_zero_error','On'
'PUBLIC','ansinull','On'
'PUBLIC','ansi_permissions','On'
'PUBLIC','close_on_endtrans','On'
'PUBLIC','tsql_variables','Off'
'PUBLIC','chained','On'
'PUBLIC','nearest_century','50'
'PUBLIC','fire_triggers','On'
'PUBLIC','background_priority','Off'

'PUBLIC','login_mode','Standard'
'PUBLIC','extern_login_credentials','Effective_user'
'PUBLIC','integrated_server_name',''
'PUBLIC','connection_authentication',''
'PUBLIC','java_main_userid',''
'PUBLIC','java_location',''
'PUBLIC','java_vm_options',''
'PUBLIC','java_class_path',''
'PUBLIC','suppress_tds_debugging','Off'
'PUBLIC','database_authentication',''
'PUBLIC','default_timestamp_increment','1'
'PUBLIC','escape_character','On'
'PUBLIC','prefetch','Conditional'
'PUBLIC','continue_after_raiserror','On'
'PUBLIC','cis_option','0'
'PUBLIC','cis_rowset_size','50'
'PUBLIC','ansi_close_cursors_on_rollback','Off'
'PUBLIC','max_statement_count','50'
'PUBLIC','max_cursor_count','50'
'PUBLIC','min_password_length','3'
'PUBLIC','auditing','Off'
'PUBLIC','conn_auditing','On'
'PUBLIC','auditing_options','4294967295'
'PUBLIC','tds_empty_string_is_null','Off'
'PUBLIC','optimization_level','9'
'PUBLIC','extended_join_syntax','On'
'PUBLIC','ansi_update_constraints','Cursors'
'PUBLIC','optimization_goal','All-rows'
'PUBLIC','global_database_id','2147483647'
'PUBLIC','max_hash_size','10'
'PUBLIC','prevent_article_pkey_update','On'
'PUBLIC','truncate_timestamp_values','Off'
'PUBLIC','return_date_time_as_string','Off'
'PUBLIC','first_day_of_week','7'
'PUBLIC','preserve_source_format','On'
'PUBLIC','time_zone_adjustment','0'
'PUBLIC','exclude_operators',''
'PUBLIC','user_estimates','Override-magic'
'PUBLIC','max_plans_cached','100'
'PUBLIC','sort_collation','Internal'
'PUBLIC','pinned_cursor_percent_of_cache','10'
'PUBLIC','on_charset_conversion_failure','Ignore'
'PUBLIC','update_statistics','On'
'PUBLIC','collect_statistics_on_dml_updates','On'
'PUBLIC','max_recursive_iterations','100'
'PUBLIC','max_query_tasks','0'
'PUBLIC','for_xml_null_treatment','Omit'
'PUBLIC','subsume_row_locks','On'
'PUBLIC','read_past_deleted','On'
'PUBLIC','temp_space_limit_check','On'
'PUBLIC','force_view_creation','Off'
'PUBLIC','dedicated_task','Off'
'PUBLIC','debug_messages','Off'
'PUBLIC','optimization_workload','Mixed'
'PUBLIC','odbc_distinguish_char_and_varchar','Off'
'PUBLIC','remote_idle_timeout','15'
'PUBLIC','ansi_substring','On'
'PUBLIC','odbc_describe_binary_as_varbinary','Off'
'PUBLIC','rollback_on_deadlock','On'
'PUBLIC','log_deadlocks','Off'
'PUBLIC','webservice_namespace_host',''
'PUBLIC','webservice_sessionid_name','SessionID'
'PUBLIC','http_session_timeout','30'
'PUBLIC','http_connection_pool_basesize','10'
'PUBLIC','http_connection_pool_timeout','60'
'PUBLIC','request_timeout','0'
'PUBLIC','synchronize_mirror_on_commit','Off'

'PUBLIC','uuid_has_hyphens','On'
'PUBLIC','verify_password_function',''
'PUBLIC','allow_snapshot_isolation','Off'
'PUBLIC','default_dbspace',''
'PUBLIC','oem_string',''
'PUBLIC','max_temp_space','0'
'PUBLIC','secure_feature_key',''
'PUBLIC','materialized_view_optimization','Stale'
'PUBLIC','tsql_outer_joins','Off'
'PUBLIC','post_login_procedure','dbo.sa_post_login_procedure'
'PUBLIC','max_client_statements_cached','10'
'PUBLIC','query_mem_timeout','-1'
'PUBLIC','allow_read_client_file','Off'
'PUBLIC','allow_write_client_file','Off'
'PUBLIC','priority','normal'
'PUBLIC','max_priority','normal'
'PUBLIC','progress_messages','Off'
'PUBLIC','st_geometry_describe_type','CHAR'
'PUBLIC','st_geometry_astext_format','WKT'
'PUBLIC','st_geometry_asbinary_format','WKB'
'PUBLIC','st_geometry_asxml_format','GML'
'PUBLIC','st_geometry_on_invalid','Error'
'PUBLIC','st_geometry_interpolation','relative-tolerance-percent=.3'
'PUBLIC','min_role_admins','1'
'PUBLIC','trusted_certificates_file',''
'PUBLIC','db_publisher','-1'
'PUBLIC','auto_commit_on_create_local_temp_index','Off'
'PUBLIC','disk_sandbox','Off'
'PUBLIC','verify_all_columns','Off'
'PUBLIC','delete_old_logs','Off'
'PUBLIC','qualify_owners','On'
'PUBLIC','quote_all_identifiers','Off'
'PUBLIC','replication_error',''
'PUBLIC','replication_error_piece',''
'PUBLIC','subscribe_by_remote','On'
'PUBLIC','verify_threshold','1000'
'PUBLIC','blob_threshold','256'
'PUBLIC','compression','6'
'PUBLIC','external_remote_options','Off'
'PUBLIC','save_remote_passwords','On'
'PUBLIC','sr_date_format','yyyy/mm/dd'
'PUBLIC','sr_time_format','hh:nn:ss.Ssssss'
'PUBLIC','sr_timestamp_format','yyyy/mm/dd hh:nn:ss.Ssssss'
'PUBLIC','sr_timestamp_with_time_zone_format','yyyy/mm/dd hh:nn:ss.Ssssss +hh:nn'
'PUBLIC','assume_distinct_servers','Off'
'PUBLIC','ml_remote_id',''

<?xml version="1.0" encoding="UTF-8"?>
<!--

    IIS configuration sections.

    For schema documentation, see
    %windir%\system32\inetsrv\config\schema\IIS_schema.xml.

    Please make a backup of this file before making any changes to it.

-->

<configuration>

    <!--

        The <configSections> section controls the registration of sections.
        Section is the basic unit of deployment, locking, searching and
        containment for configuration settings.

```
    Every section belongs to one section group.
    A section group is a container of logically-related sections.

    Sections cannot be nested.
    Section groups may be nested.

    <section
       name=""  [Required, Collection Key] [XML name of the section]
       allowDefinition="Everywhere" [MachineOnly|MachineToApplication|AppHostOnly|Everywhere] [Level where it can be set]
       overrideModeDefault="Allow"  [Allow|Deny] [Default delegation mode]
       allowLocation="true"  [true|false] [Allowed in location tags]
    />

    The recommended way to unlock sections is by using a location tag:
    <location path="Default Web Site" overrideMode="Allow">
       <system.webServer>
          <asp />
       </system.webServer>
    </location>

-->
<configSections>
    <sectionGroup name="system.applicationHost">
       <section name="applicationPools" allowDefinition="AppHostOnly" overrideModeDefault="Deny" />
       <section name="configHistory" allowDefinition="AppHostOnly" overrideModeDefault="Deny" />
       <section name="customMetadata" allowDefinition="AppHostOnly" overrideModeDefault="Deny" />
       <section name="listenerAdapters" allowDefinition="AppHostOnly" overrideModeDefault="Deny" />
       <section name="log" allowDefinition="AppHostOnly" overrideModeDefault="Deny" />
       <section name="serviceAutoStartProviders" allowDefinition="AppHostOnly" overrideModeDefault="Deny" />
       <section name="sites" allowDefinition="AppHostOnly" overrideModeDefault="Deny" />
       <section name="webLimits" allowDefinition="AppHostOnly" overrideModeDefault="Deny" />
    </sectionGroup>

    <sectionGroup name="system.webServer">
       <section name="asp" overrideModeDefault="Deny" />
       <section name="caching" overrideModeDefault="Allow" />
       <section name="cgi" overrideModeDefault="Deny" />
       <section name="defaultDocument" overrideModeDefault="Allow" />
       <section name="directoryBrowse" overrideModeDefault="Allow" />
       <section name="fastCgi" allowDefinition="AppHostOnly" overrideModeDefault="Deny" />
       <section name="globalModules" allowDefinition="AppHostOnly" overrideModeDefault="Deny" />
       <section name="handlers" overrideModeDefault="Deny" />
       <section name="httpCompression" allowDefinition="AppHostOnly" overrideModeDefault="Deny" />
       <section name="httpErrors" overrideModeDefault="Allow" />
       <section name="httpLogging" overrideModeDefault="Deny" />
       <section name="httpProtocol" overrideModeDefault="Allow" />
       <section name="httpRedirect" overrideModeDefault="Allow" />
       <section name="httpTracing" overrideModeDefault="Deny" />
       <section name="isapiFilters" allowDefinition="MachineToApplication" overrideModeDefault="Deny" />
       <section name="modules" allowDefinition="MachineToApplication" overrideModeDefault="Deny" />
       <section name="applicationInitialization" allowDefinition="MachineToApplication" overrideModeDefault="Allow" />
       <section name="odbcLogging" overrideModeDefault="Deny" />
       <sectionGroup name="security">
          <section name="access" overrideModeDefault="Deny" />
          <section name="applicationDependencies" overrideModeDefault="Deny" />
          <sectionGroup name="authentication">
             <section name="anonymousAuthentication" overrideModeDefault="Deny" />
             <section name="basicAuthentication" overrideModeDefault="Deny" />
             <section name="clientCertificateMappingAuthentication" overrideModeDefault="Deny" />
             <section name="digestAuthentication" overrideModeDefault="Deny" />
             <section name="iisClientCertificateMappingAuthentication" overrideModeDefault="Deny" />
             <section name="windowsAuthentication" overrideModeDefault="Deny" />
          </sectionGroup>
          <section name="authorization" overrideModeDefault="Allow" />
          <section name="ipSecurity" overrideModeDefault="Deny" />
          <section name="dynamicIpSecurity" overrideModeDefault="Deny" />
```

```
            <section name="isapiCgiRestriction" allowDefinition="AppHostOnly" overrideModeDefault="Deny" />
            <section name="requestFiltering" overrideModeDefault="Allow" />
         </sectionGroup>
         <section name="serverRuntime" overrideModeDefault="Deny" />
         <section name="serverSideInclude" overrideModeDefault="Deny" />
         <section name="staticContent" overrideModeDefault="Allow" />
         <sectionGroup name="tracing">
            <section name="traceFailedRequests" overrideModeDefault="Allow" />
            <section name="traceProviderDefinitions" overrideModeDefault="Deny" />
         </sectionGroup>
         <section name="urlCompression" overrideModeDefault="Allow" />
         <section name="validation" overrideModeDefault="Allow" />
         <sectionGroup name="webdav">
            <section name="globalSettings" overrideModeDefault="Deny" />
            <section name="authoring" overrideModeDefault="Deny" />
            <section name="authoringRules" overrideModeDefault="Deny" />
         </sectionGroup>
         <section name="webSocket" overrideModeDefault="Deny" />
      </sectionGroup>
      <sectionGroup name="system.ftpServer">
         <section name="log" overrideModeDefault="Deny" allowDefinition="AppHostOnly" />
         <section name="firewallSupport" overrideModeDefault="Deny" allowDefinition="AppHostOnly" />
         <section name="caching" overrideModeDefault="Deny" allowDefinition="AppHostOnly" />
         <section name="providerDefinitions" overrideModeDefault="Deny" />
         <sectionGroup name="security">
            <section name="ipSecurity" overrideModeDefault="Deny" />
            <section name="requestFiltering" overrideModeDefault="Deny" />
            <section name="authorization" overrideModeDefault="Deny" />
            <section name="authentication" overrideModeDefault="Deny" />
         </sectionGroup>
         <section name="serverRuntime" overrideModeDefault="Deny" allowDefinition="AppHostOnly" />
      </sectionGroup>
   </configSections>

   <configProtectedData>
      <providers>
         <add name="IISWASOnlyRsaProvider" type="" description="Uses RsaCryptoServiceProvider to encrypt and decrypt"
keyContainerName="iisWasKey" cspProviderName="" useMachineContainer="true" useOAEP="false" />
         <add name="AesProvider" type="Microsoft.ApplicationHost.AesProtectedConfigurationProvider" description="Uses an AES
session key to encrypt and decrypt" keyContainerName="iisConfigurationKey" cspProviderName="" useOAEP="false"
useMachineContainer="true"
sessionKey="AQIAAA5mAAAApAAAKjkVz7g4Jyn5pvwUmsSfeSpjugh8yhMNojOnE33kcAZDZvAG7KsoTXnchxdbApRjBKBHPAJVNVNZM
7FJAnC6spi1r4iDN1tT405BvkCqxFdjMghcfaTVbhaClc7UG3ixbzqnKEwwo3iz+hX2s9/GEUOmXZy5mM4doG6RVztLYylSqaANSHN3mFHs
o5+qsdHzcVtGRr07BWHJGoO5PbBQB9ksKBBXi5q0xxK0yYFO3yasySlrHD/yWIvnA6IwuNQv/tzPXiM3vfWJgNDM2I5FAVHXQKKrShDEa
wTiY7wt1iQ18+wFto/gO62QiHLZHYMfolbkK+QI6q0Y5BA+meK2aw==" />
         <add name="IISWASOnlyAesProvider" type="Microsoft.ApplicationHost.AesProtectedConfigurationProvider"
description="Uses an AES session key to encrypt and decrypt" keyContainerName="iisWasKey" cspProviderName=""
useOAEP="false" useMachineContainer="true"
sessionKey="AQIAAA5mAAAApAAAxdwFI61ixF6u47QjcOZ3wrctPaUDamlCmf4i4D1WlgWb/rUvYtgsvk1FykM/1h3ke/IbdMSjIfGLnES4Kj
/V2Dsq6JvnJZ14VcmACjpOUWjZJ7BdrG3glgzd2cJaugS/CPiZ27UciOos1GNLkMwFqW0eQhLMxtnr0c/37izVrDSMy6Og0vL8XMakGynHCr
qK7azVXGjpCz5pRY+DLqBdC4Orf1fYyx4To5aQyaOZzCXcv5B+iKPRgfyhgb0EnBB9lBcY2QAPqc3WhuwaHmysPbpwzpmbbGCfzHmSHh
xkx8Z6zVBJFmXnpQp5ekK89X3Ct6YYXm/XkFWJMZkM2PQLRw==" />
      </providers>
   </configProtectedData>

   <system.applicationHost>

      <applicationPools>
         <add name="DefaultAppPool" queueLength="65535" autoStart="true">
            <processModel maxProcesses="1" />
            <failure rapidFailProtection="false" />
            <cpu smpAffinitized="true" smpProcessorAffinityMask="4293918720" smpProcessorAffinityMask2="4095" />
         </add>
         <applicationPoolDefaults managedRuntimeVersion="v4.0">
            <processModel identityType="ApplicationPoolIdentity" />
         </applicationPoolDefaults>
      </applicationPools>
```

```
        <!--

          The <customMetadata> section is used internally by the Admin Base Objects
          (ABO) Compatibility component. Please do not modify its content.

        -->
        <customMetadata />

        <!--

          The <listenerAdapters> section defines the protocols with which the
          Windows Process Activation Service (WAS) binds.

        -->
        <listenerAdapters>
            <add name="http" />
        </listenerAdapters>

        <log>
            <centralBinaryLogFile enabled="true" directory="%SystemDrive%\inetpub\logs\LogFiles" />
            <centralW3CLogFile enabled="true" directory="%SystemDrive%\inetpub\logs\LogFiles" />
        </log>

        <sites>
            <site name="Default Web Site" id="1" serverAutoStart="true">
                <application path="/">
                    <virtualDirectory path="/" physicalPath="%SystemDrive%\inetpub\wwwroot" />
                    <virtualDirectory path="/tpcc" physicalPath="c:\tpcc\isapi" userName="Administrator"
password="[enc:AesProvider:9N9bsD9wIybSbHSvKPwvz5PR8Ejzk7sxn8Bddw0DOqM=:enc]" />
                </application>
                <bindings>
                    <binding protocol="http" bindingInformation="*:80:" />
                </bindings>
                <limits connectionTimeout="00:10:00" />
            </site>
            <siteDefaults>
                <logFile logFormat="W3C" directory="%SystemDrive%\inetpub\logs\LogFiles" />
                <traceFailedRequestsLogging directory="%SystemDrive%\inetpub\logs\FailedReqLogFiles" />
            </siteDefaults>
            <applicationDefaults applicationPool="DefaultAppPool" />
            <virtualDirectoryDefaults allowSubDirConfig="true" />
        </sites>

        <webLimits minBytesPerSecond="0" />

    </system.applicationHost>

    <system.webServer>

        <asp />

        <caching enabled="true" enableKernelCache="true">
        </caching>

        <cgi />

        <defaultDocument enabled="true">
            <files>
                <add value="Default.htm" />
                <add value="Default.asp" />
                <add value="index.htm" />
                <add value="index.html" />
                <add value="iisstart.htm" />
            </files>
        </defaultDocument>
```

```
    <directoryBrowse enabled="false" />

    <fastCgi />

    <!--

      The <globalModules> section defines all native-code modules.
      To enable a module, specify it in the <modules> section.

    -->
    <globalModules>
      <add name="UriCacheModule" image="%windir%\System32\inetsrv\cachuri.dll" />
      <add name="FileCacheModule" image="%windir%\System32\inetsrv\cachfile.dll" />
      <add name="TokenCacheModule" image="%windir%\System32\inetsrv\cachtokn.dll" />
      <add name="HttpCacheModule" image="%windir%\System32\inetsrv\cachhttp.dll" />
      <add name="StaticCompressionModule" image="%windir%\System32\inetsrv\compstat.dll" />
      <add name="DefaultDocumentModule" image="%windir%\System32\inetsrv\defdoc.dll" />
      <add name="DirectoryListingModule" image="%windir%\System32\inetsrv\dirlist.dll" />
      <add name="ProtocolSupportModule" image="%windir%\System32\inetsrv\protsup.dll" />
      <add name="ServerSideIncludeModule" image="%windir%\System32\inetsrv\iis_ssi.dll" />
      <add name="StaticFileModule" image="%windir%\System32\inetsrv\static.dll" />
      <add name="AnonymousAuthenticationModule" image="%windir%\System32\inetsrv\authanon.dll" />
      <add name="RequestFilteringModule" image="%windir%\System32\inetsrv\modrqflt.dll" />
      <add name="CustomErrorModule" image="%windir%\System32\inetsrv\custerr.dll" />
      <add name="HttpLoggingModule" image="%windir%\System32\inetsrv\loghttp.dll" />
      <add name="IsapiModule" image="%windir%\System32\inetsrv\isapi.dll" />
      <add name="IsapiFilterModule" image="%windir%\System32\inetsrv\filter.dll" />
      <add name="CgiModule" image="%windir%\System32\inetsrv\cgi.dll" />
      <add name="FastCgiModule" image="%windir%\System32\inetsrv\iisfcgi.dll" />
    </globalModules>

    <handlers accessPolicy="Read, Script">
      <add name="ISAPI-dll" path="*.dll" verb="*" modules="IsapiModule" resourceType="File" requireAccess="Execute"
allowPathInfo="true" />
      <add name="CGI-exe" path="*.exe" verb="*" modules="CgiModule" resourceType="File" requireAccess="Execute"
allowPathInfo="true" />
      <add name="SSINC-stm" path="*.stm" verb="GET,HEAD,POST" modules="ServerSideIncludeModule" resourceType="File"
/>
      <add name="SSINC-shtm" path="*.shtm" verb="GET,HEAD,POST" modules="ServerSideIncludeModule"
resourceType="File" />
      <add name="SSINC-shtml" path="*.shtml" verb="GET,HEAD,POST" modules="ServerSideIncludeModule"
resourceType="File" />
      <add name="TRACEVerbHandler" path="*" verb="TRACE" modules="ProtocolSupportModule" requireAccess="None" />
      <add name="OPTIONSVerbHandler" path="*" verb="OPTIONS" modules="ProtocolSupportModule" requireAccess="None"
/>
      <add name="StaticFile" path="*" verb="*" modules="StaticFileModule,DefaultDocumentModule,DirectoryListingModule"
resourceType="Either" requireAccess="Read" />
    </handlers>

    <httpCompression directory="%SystemDrive%\inetpub\temp\IIS Temporary Compressed Files">
      <scheme name="gzip" dll="%Windir%\system32\inetsrv\gzip.dll" />
      <staticTypes>
        <add mimeType="text/*" enabled="true" />
        <add mimeType="message/*" enabled="true" />
        <add mimeType="application/javascript" enabled="true" />
        <add mimeType="application/atom+xml" enabled="true" />
        <add mimeType="application/xaml+xml" enabled="true" />
        <add mimeType="*/*" enabled="false" />
      </staticTypes>
    </httpCompression>

    <httpErrors lockAttributes="allowAbsolutePathsWhenDelegated,defaultPath">
      <error statusCode="401" prefixLanguageFilePath="%SystemDrive%\inetpub\custerr" path="401.htm" />
      <error statusCode="403" prefixLanguageFilePath="%SystemDrive%\inetpub\custerr" path="403.htm" />
      <error statusCode="404" prefixLanguageFilePath="%SystemDrive%\inetpub\custerr" path="404.htm" />
      <error statusCode="405" prefixLanguageFilePath="%SystemDrive%\inetpub\custerr" path="405.htm" />
      <error statusCode="406" prefixLanguageFilePath="%SystemDrive%\inetpub\custerr" path="406.htm" />
```

```
    <error statusCode="412" prefixLanguageFilePath="%SystemDrive%\inetpub\custerr" path="412.htm" />
    <error statusCode="500" prefixLanguageFilePath="%SystemDrive%\inetpub\custerr" path="500.htm" />
    <error statusCode="501" prefixLanguageFilePath="%SystemDrive%\inetpub\custerr" path="501.htm" />
    <error statusCode="502" prefixLanguageFilePath="%SystemDrive%\inetpub\custerr" path="502.htm" />
</httpErrors>

<httpLogging dontLog="false" />

<httpProtocol>
    <customHeaders>
        <clear />
    </customHeaders>
    <redirectHeaders>
        <clear />
    </redirectHeaders>
</httpProtocol>

<httpRedirect />

<httpTracing />

<isapiFilters />

<modules>
    <add name="HttpCacheModule" lockItem="true" />
    <add name="StaticCompressionModule" lockItem="true" />
    <add name="DefaultDocumentModule" lockItem="true" />
    <add name="DirectoryListingModule" lockItem="true" />
    <add name="IsapiFilterModule" lockItem="true" />
    <add name="ProtocolSupportModule" lockItem="true" />
    <add name="ServerSideIncludeModule" lockItem="true" />
    <add name="StaticFileModule" lockItem="true" />
    <add name="AnonymousAuthenticationModule" lockItem="true" />
    <add name="RequestFilteringModule" lockItem="true" />
    <add name="CustomErrorModule" lockItem="true" />
    <add name="IsapiModule" lockItem="true" />
    <add name="HttpLoggingModule" lockItem="true" />
    <add name="CgiModule" lockItem="true" />
    <add name="FastCgiModule" lockItem="true" />
</modules>

<odbcLogging />

<security>

    <access sslFlags="None" />

    <applicationDependencies />

    <authentication>

        <anonymousAuthentication enabled="true" userName="IUSR" />

        <basicAuthentication />

        <clientCertificateMappingAuthentication />

        <digestAuthentication />

        <iisClientCertificateMappingAuthentication />

        <windowsAuthentication />

    </authentication>

    <authorization />
```

```
    <ipSecurity />

    <isapiCgiRestriction>
      <add path="C:\tpcc\tpccisapicom.dll" allowed="true" />
    </isapiCgiRestriction>

    <requestFiltering>
      <fileExtensions allowUnlisted="true" applyToWebDAV="true" />
      <verbs allowUnlisted="true" applyToWebDAV="true" />
      <hiddenSegments applyToWebDAV="true">
        <add segment="web.config" />
      </hiddenSegments>
    </requestFiltering>

  </security>

  <serverRuntime />

  <serverSideInclude ssiExecDisable="false" />

  <staticContent lockAttributes="isDocFooterFileName">
    <mimeMap fileExtension=".323" mimeType="text/h323" />
    <mimeMap fileExtension=".3g2" mimeType="video/3gpp2" />
    <mimeMap fileExtension=".3gp2" mimeType="video/3gpp2" />
    <mimeMap fileExtension=".3gp" mimeType="video/3gpp" />
    <mimeMap fileExtension=".3gpp" mimeType="video/3gpp" />
    <mimeMap fileExtension=".aaf" mimeType="application/octet-stream" />
    <mimeMap fileExtension=".aac" mimeType="audio/aac" />
    <mimeMap fileExtension=".aca" mimeType="application/octet-stream" />
    <mimeMap fileExtension=".accdb" mimeType="application/msaccess" />
    <mimeMap fileExtension=".accde" mimeType="application/msaccess" />
    <mimeMap fileExtension=".accdt" mimeType="application/msaccess" />
    <mimeMap fileExtension=".acx" mimeType="application/internet-property-stream" />
    <mimeMap fileExtension=".adt" mimeType="audio/vnd.dlna.adts" />
    <mimeMap fileExtension=".adts" mimeType="audio/vnd.dlna.adts" />
    <mimeMap fileExtension=".afm" mimeType="application/octet-stream" />
    <mimeMap fileExtension=".ai" mimeType="application/postscript" />
    <mimeMap fileExtension=".aif" mimeType="audio/x-aiff" />
    <mimeMap fileExtension=".aifc" mimeType="audio/aiff" />
    <mimeMap fileExtension=".aiff" mimeType="audio/aiff" />
    <mimeMap fileExtension=".application" mimeType="application/x-ms-application" />
    <mimeMap fileExtension=".art" mimeType="image/x-jg" />
    <mimeMap fileExtension=".asd" mimeType="application/octet-stream" />
    <mimeMap fileExtension=".asf" mimeType="video/x-ms-asf" />
    <mimeMap fileExtension=".asi" mimeType="application/octet-stream" />
    <mimeMap fileExtension=".asm" mimeType="text/plain" />
    <mimeMap fileExtension=".asr" mimeType="video/x-ms-asf" />
    <mimeMap fileExtension=".asx" mimeType="video/x-ms-asf" />
    <mimeMap fileExtension=".atom" mimeType="application/atom+xml" />
    <mimeMap fileExtension=".au" mimeType="audio/basic" />
    <mimeMap fileExtension=".avi" mimeType="video/avi" />
    <mimeMap fileExtension=".axs" mimeType="application/olescript" />
    <mimeMap fileExtension=".bas" mimeType="text/plain" />
    <mimeMap fileExtension=".bcpio" mimeType="application/x-bcpio" />
    <mimeMap fileExtension=".bin" mimeType="application/octet-stream" />
    <mimeMap fileExtension=".bmp" mimeType="image/bmp" />
    <mimeMap fileExtension=".c" mimeType="text/plain" />
    <mimeMap fileExtension=".cab" mimeType="application/vnd.ms-cab-compressed" />
    <mimeMap fileExtension=".calx" mimeType="application/vnd.ms-office.calx" />
    <mimeMap fileExtension=".cat" mimeType="application/vnd.ms-pki.seccat" />
    <mimeMap fileExtension=".cdf" mimeType="application/x-cdf" />
    <mimeMap fileExtension=".chm" mimeType="application/octet-stream" />
    <mimeMap fileExtension=".class" mimeType="application/x-java-applet" />
    <mimeMap fileExtension=".clp" mimeType="application/x-msclip" />
    <mimeMap fileExtension=".cmx" mimeType="image/x-cmx" />
    <mimeMap fileExtension=".cnf" mimeType="text/plain" />
    <mimeMap fileExtension=".cod" mimeType="image/cis-cod" />
```

```
        <mimeMap fileExtension=".cpio" mimeType="application/x-cpio" />
        <mimeMap fileExtension=".cpp" mimeType="text/plain" />
        <mimeMap fileExtension=".crd" mimeType="application/x-mscardfile" />
        <mimeMap fileExtension=".crl" mimeType="application/pkix-crl" />
        <mimeMap fileExtension=".crt" mimeType="application/x-x509-ca-cert" />
        <mimeMap fileExtension=".csh" mimeType="application/x-csh" />
        <mimeMap fileExtension=".css" mimeType="text/css" />
        <mimeMap fileExtension=".csv" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".cur" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".dcr" mimeType="application/x-director" />
        <mimeMap fileExtension=".deploy" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".der" mimeType="application/x-x509-ca-cert" />
        <mimeMap fileExtension=".dib" mimeType="image/bmp" />
        <mimeMap fileExtension=".dir" mimeType="application/x-director" />
        <mimeMap fileExtension=".disco" mimeType="text/xml" />
        <mimeMap fileExtension=".dll" mimeType="application/x-msdownload" />
        <mimeMap fileExtension=".dll.config" mimeType="text/xml" />
        <mimeMap fileExtension=".dlm" mimeType="text/dlm" />
        <mimeMap fileExtension=".doc" mimeType="application/msword" />
        <mimeMap fileExtension=".docm" mimeType="application/vnd.ms-word.document.macroEnabled.12" />
        <mimeMap fileExtension=".docx" mimeType="application/vnd.openxmlformats-
officedocument.wordprocessingml.document" />
        <mimeMap fileExtension=".dot" mimeType="application/msword" />
        <mimeMap fileExtension=".dotm" mimeType="application/vnd.ms-word.template.macroEnabled.12" />
        <mimeMap fileExtension=".dotx" mimeType="application/vnd.openxmlformats-
officedocument.wordprocessingml.template" />
        <mimeMap fileExtension=".dsp" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".dtd" mimeType="text/xml" />
        <mimeMap fileExtension=".dvi" mimeType="application/x-dvi" />
        <mimeMap fileExtension=".dvr-ms" mimeType="video/x-ms-dvr" />
        <mimeMap fileExtension=".dwf" mimeType="drawing/x-dwf" />
        <mimeMap fileExtension=".dwp" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".dxr" mimeType="application/x-director" />
        <mimeMap fileExtension=".eml" mimeType="message/rfc822" />
        <mimeMap fileExtension=".emz" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".eot" mimeType="application/vnd.ms-fontobject" />
        <mimeMap fileExtension=".eps" mimeType="application/postscript" />
        <mimeMap fileExtension=".etx" mimeType="text/x-setext" />
        <mimeMap fileExtension=".evy" mimeType="application/envoy" />
        <mimeMap fileExtension=".exe" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".exe.config" mimeType="text/xml" />
        <mimeMap fileExtension=".fdf" mimeType="application/vnd.fdf" />
        <mimeMap fileExtension=".fif" mimeType="application/fractals" />
        <mimeMap fileExtension=".fla" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".flr" mimeType="x-world/x-vrml" />
        <mimeMap fileExtension=".flv" mimeType="video/x-flv" />
        <mimeMap fileExtension=".gif" mimeType="image/gif" />
        <mimeMap fileExtension=".gtar" mimeType="application/x-gtar" />
        <mimeMap fileExtension=".gz" mimeType="application/x-gzip" />
        <mimeMap fileExtension=".h" mimeType="text/plain" />
        <mimeMap fileExtension=".hdf" mimeType="application/x-hdf" />
        <mimeMap fileExtension=".hdml" mimeType="text/x-hdml" />
        <mimeMap fileExtension=".hhc" mimeType="application/x-oleobject" />
        <mimeMap fileExtension=".hhk" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".hhp" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".hlp" mimeType="application/winhlp" />
        <mimeMap fileExtension=".hqx" mimeType="application/mac-binhex40" />
        <mimeMap fileExtension=".hta" mimeType="application/hta" />
        <mimeMap fileExtension=".htc" mimeType="text/x-component" />
        <mimeMap fileExtension=".htm" mimeType="text/html" />
        <mimeMap fileExtension=".html" mimeType="text/html" />
        <mimeMap fileExtension=".htt" mimeType="text/webviewhtml" />
        <mimeMap fileExtension=".hxt" mimeType="text/html" />
        <mimeMap fileExtension=".ico" mimeType="image/x-icon" />
        <mimeMap fileExtension=".ics" mimeType="text/calendar" />
        <mimeMap fileExtension=".ief" mimeType="image/ief" />
        <mimeMap fileExtension=".iii" mimeType="application/x-iphone" />
```

```
<mimeMap fileExtension=".inf" mimeType="application/octet-stream" />
<mimeMap fileExtension=".ins" mimeType="application/x-internet-signup" />
<mimeMap fileExtension=".isp" mimeType="application/x-internet-signup" />
<mimeMap fileExtension=".IVF" mimeType="video/x-ivf" />
<mimeMap fileExtension=".jar" mimeType="application/java-archive" />
<mimeMap fileExtension=".java" mimeType="application/octet-stream" />
<mimeMap fileExtension=".jck" mimeType="application/liquidmotion" />
<mimeMap fileExtension=".jcz" mimeType="application/liquidmotion" />
<mimeMap fileExtension=".jfif" mimeType="image/pjpeg" />
<mimeMap fileExtension=".jpb" mimeType="application/octet-stream" />
<mimeMap fileExtension=".jpe" mimeType="image/jpeg" />
<mimeMap fileExtension=".jpeg" mimeType="image/jpeg" />
<mimeMap fileExtension=".jpg" mimeType="image/jpeg" />
<mimeMap fileExtension=".js" mimeType="application/javascript" />
<mimeMap fileExtension=".json" mimeType="application/json" />
<mimeMap fileExtension=".jsx" mimeType="text/jscript" />
<mimeMap fileExtension=".latex" mimeType="application/x-latex" />
<mimeMap fileExtension=".lit" mimeType="application/x-ms-reader" />
<mimeMap fileExtension=".lpk" mimeType="application/octet-stream" />
<mimeMap fileExtension=".lsf" mimeType="video/x-la-asf" />
<mimeMap fileExtension=".lsx" mimeType="video/x-la-asf" />
<mimeMap fileExtension=".lzh" mimeType="application/octet-stream" />
<mimeMap fileExtension=".m13" mimeType="application/x-msmediaview" />
<mimeMap fileExtension=".m14" mimeType="application/x-msmediaview" />
<mimeMap fileExtension=".m1v" mimeType="video/mpeg" />
<mimeMap fileExtension=".m2ts" mimeType="video/vnd.dlna.mpeg-tts" />
<mimeMap fileExtension=".m3u" mimeType="audio/x-mpegurl" />
<mimeMap fileExtension=".m4a" mimeType="audio/mp4" />
<mimeMap fileExtension=".m4v" mimeType="video/mp4" />
<mimeMap fileExtension=".man" mimeType="application/x-troff-man" />
<mimeMap fileExtension=".manifest" mimeType="application/x-ms-manifest" />
<mimeMap fileExtension=".map" mimeType="text/plain" />
<mimeMap fileExtension=".mdb" mimeType="application/x-msaccess" />
<mimeMap fileExtension=".mdp" mimeType="application/octet-stream" />
<mimeMap fileExtension=".me" mimeType="application/x-troff-me" />
<mimeMap fileExtension=".mht" mimeType="message/rfc822" />
<mimeMap fileExtension=".mhtml" mimeType="message/rfc822" />
<mimeMap fileExtension=".mid" mimeType="audio/mid" />
<mimeMap fileExtension=".midi" mimeType="audio/mid" />
<mimeMap fileExtension=".mix" mimeType="application/octet-stream" />
<mimeMap fileExtension=".mmf" mimeType="application/x-smaf" />
<mimeMap fileExtension=".mno" mimeType="text/xml" />
<mimeMap fileExtension=".mny" mimeType="application/x-msmoney" />
<mimeMap fileExtension=".mov" mimeType="video/quicktime" />
<mimeMap fileExtension=".movie" mimeType="video/x-sgi-movie" />
<mimeMap fileExtension=".mp2" mimeType="video/mpeg" />
<mimeMap fileExtension=".mp3" mimeType="audio/mpeg" />
<mimeMap fileExtension=".mp4" mimeType="video/mp4" />
<mimeMap fileExtension=".mp4v" mimeType="video/mp4" />
<mimeMap fileExtension=".mpa" mimeType="video/mpeg" />
<mimeMap fileExtension=".mpe" mimeType="video/mpeg" />
<mimeMap fileExtension=".mpeg" mimeType="video/mpeg" />
<mimeMap fileExtension=".mpg" mimeType="video/mpeg" />
<mimeMap fileExtension=".mpp" mimeType="application/vnd.ms-project" />
<mimeMap fileExtension=".mpv2" mimeType="video/mpeg" />
<mimeMap fileExtension=".ms" mimeType="application/x-troff-ms" />
<mimeMap fileExtension=".msi" mimeType="application/octet-stream" />
<mimeMap fileExtension=".mso" mimeType="application/octet-stream" />
<mimeMap fileExtension=".mvb" mimeType="application/x-msmediaview" />
<mimeMap fileExtension=".mvc" mimeType="application/x-miva-compiled" />
<mimeMap fileExtension=".nc" mimeType="application/x-netcdf" />
<mimeMap fileExtension=".nsc" mimeType="video/x-ms-asf" />
<mimeMap fileExtension=".nws" mimeType="message/rfc822" />
<mimeMap fileExtension=".ocx" mimeType="application/octet-stream" />
<mimeMap fileExtension=".oda" mimeType="application/oda" />
<mimeMap fileExtension=".odc" mimeType="text/x-ms-odc" />
<mimeMap fileExtension=".ods" mimeType="application/oleobject" />
```

```
        <mimeMap fileExtension=".oga" mimeType="audio/ogg" />
        <mimeMap fileExtension=".ogg" mimeType="video/ogg" />
        <mimeMap fileExtension=".ogv" mimeType="video/ogg" />
        <mimeMap fileExtension=".one" mimeType="application/onenote" />
        <mimeMap fileExtension=".onea" mimeType="application/onenote" />
        <mimeMap fileExtension=".onetoc" mimeType="application/onenote" />
        <mimeMap fileExtension=".onetoc2" mimeType="application/onenote" />
        <mimeMap fileExtension=".onetmp" mimeType="application/onenote" />
        <mimeMap fileExtension=".onepkg" mimeType="application/onenote" />
        <mimeMap fileExtension=".osdx" mimeType="application/opensearchdescription+xml" />
        <mimeMap fileExtension=".otf" mimeType="font/otf" />
        <mimeMap fileExtension=".p10" mimeType="application/pkcs10" />
        <mimeMap fileExtension=".p12" mimeType="application/x-pkcs12" />
        <mimeMap fileExtension=".p7b" mimeType="application/x-pkcs7-certificates" />
        <mimeMap fileExtension=".p7c" mimeType="application/pkcs7-mime" />
        <mimeMap fileExtension=".p7m" mimeType="application/pkcs7-mime" />
        <mimeMap fileExtension=".p7r" mimeType="application/x-pkcs7-certreqresp" />
        <mimeMap fileExtension=".p7s" mimeType="application/pkcs7-signature" />
        <mimeMap fileExtension=".pbm" mimeType="image/x-portable-bitmap" />
        <mimeMap fileExtension=".pcx" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".pcz" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".pdf" mimeType="application/pdf" />
        <mimeMap fileExtension=".pfb" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".pfm" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".pfx" mimeType="application/x-pkcs12" />
        <mimeMap fileExtension=".pgm" mimeType="image/x-portable-graymap" />
        <mimeMap fileExtension=".pko" mimeType="application/vnd.ms-pki.pko" />
        <mimeMap fileExtension=".pma" mimeType="application/x-perfmon" />
        <mimeMap fileExtension=".pmc" mimeType="application/x-perfmon" />
        <mimeMap fileExtension=".pml" mimeType="application/x-perfmon" />
        <mimeMap fileExtension=".pmr" mimeType="application/x-perfmon" />
        <mimeMap fileExtension=".pmw" mimeType="application/x-perfmon" />
        <mimeMap fileExtension=".png" mimeType="image/png" />
        <mimeMap fileExtension=".pnm" mimeType="image/x-portable-anymap" />
        <mimeMap fileExtension=".pnz" mimeType="image/png" />
        <mimeMap fileExtension=".pot" mimeType="application/vnd.ms-powerpoint" />
        <mimeMap fileExtension=".potm" mimeType="application/vnd.ms-powerpoint.template.macroEnabled.12" />
        <mimeMap fileExtension=".potx" mimeType="application/vnd.openxmlformats-officedocument.presentationml.template"
/>
        <mimeMap fileExtension=".ppam" mimeType="application/vnd.ms-powerpoint.addin.macroEnabled.12" />
        <mimeMap fileExtension=".ppm" mimeType="image/x-portable-pixmap" />
        <mimeMap fileExtension=".pps" mimeType="application/vnd.ms-powerpoint" />
        <mimeMap fileExtension=".ppsm" mimeType="application/vnd.ms-powerpoint.slideshow.macroEnabled.12" />
        <mimeMap fileExtension=".ppsx" mimeType="application/vnd.openxmlformats-officedocument.presentationml.slideshow"
/>
        <mimeMap fileExtension=".ppt" mimeType="application/vnd.ms-powerpoint" />
        <mimeMap fileExtension=".pptm" mimeType="application/vnd.ms-powerpoint.presentation.macroEnabled.12" />
        <mimeMap fileExtension=".pptx" mimeType="application/vnd.openxmlformats-
officedocument.presentationml.presentation" />
        <mimeMap fileExtension=".prf" mimeType="application/pics-rules" />
        <mimeMap fileExtension=".prm" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".prx" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".ps" mimeType="application/postscript" />
        <mimeMap fileExtension=".psd" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".psm" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".psp" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".pub" mimeType="application/x-mspublisher" />
        <mimeMap fileExtension=".qt" mimeType="video/quicktime" />
        <mimeMap fileExtension=".qtl" mimeType="application/x-quicktimeplayer" />
        <mimeMap fileExtension=".qxd" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".ra" mimeType="audio/x-pn-realaudio" />
        <mimeMap fileExtension=".ram" mimeType="audio/x-pn-realaudio" />
        <mimeMap fileExtension=".rar" mimeType="application/octet-stream" />
        <mimeMap fileExtension=".ras" mimeType="image/x-cmu-raster" />
        <mimeMap fileExtension=".rf" mimeType="image/vnd.rn-realflash" />
        <mimeMap fileExtension=".rgb" mimeType="image/x-rgb" />
        <mimeMap fileExtension=".rm" mimeType="application/vnd.rn-realmedia" />
```

```
<mimeMap fileExtension=".rmi" mimeType="audio/mid" />
<mimeMap fileExtension=".roff" mimeType="application/x-troff" />
<mimeMap fileExtension=".rpm" mimeType="audio/x-pn-realaudio-plugin" />
<mimeMap fileExtension=".rtf" mimeType="application/rtf" />
<mimeMap fileExtension=".rtx" mimeType="text/richtext" />
<mimeMap fileExtension=".scd" mimeType="application/x-msschedule" />
<mimeMap fileExtension=".sct" mimeType="text/scriptlet" />
<mimeMap fileExtension=".sea" mimeType="application/octet-stream" />
<mimeMap fileExtension=".setpay" mimeType="application/set-payment-initiation" />
<mimeMap fileExtension=".setreg" mimeType="application/set-registration-initiation" />
<mimeMap fileExtension=".sgml" mimeType="text/sgml" />
<mimeMap fileExtension=".sh" mimeType="application/x-sh" />
<mimeMap fileExtension=".shar" mimeType="application/x-shar" />
<mimeMap fileExtension=".sit" mimeType="application/x-stuffit" />
<mimeMap fileExtension=".sldm" mimeType="application/vnd.ms-powerpoint.slide.macroEnabled.12" />
<mimeMap fileExtension=".sldx" mimeType="application/vnd.openxmlformats-officedocument.presentationml.slide" />
<mimeMap fileExtension=".smd" mimeType="audio/x-smd" />
<mimeMap fileExtension=".smi" mimeType="application/octet-stream" />
<mimeMap fileExtension=".smx" mimeType="audio/x-smd" />
<mimeMap fileExtension=".smz" mimeType="audio/x-smd" />
<mimeMap fileExtension=".snd" mimeType="audio/basic" />
<mimeMap fileExtension=".snp" mimeType="application/octet-stream" />
<mimeMap fileExtension=".spc" mimeType="application/x-pkcs7-certificates" />
<mimeMap fileExtension=".spl" mimeType="application/futuresplash" />
<mimeMap fileExtension=".spx" mimeType="audio/ogg" />
<mimeMap fileExtension=".src" mimeType="application/x-wais-source" />
<mimeMap fileExtension=".ssm" mimeType="application/streamingmedia" />
<mimeMap fileExtension=".sst" mimeType="application/vnd.ms-pki.certstore" />
<mimeMap fileExtension=".stl" mimeType="application/vnd.ms-pki.stl" />
<mimeMap fileExtension=".sv4cpio" mimeType="application/x-sv4cpio" />
<mimeMap fileExtension=".sv4crc" mimeType="application/x-sv4crc" />
<mimeMap fileExtension=".svg" mimeType="image/svg+xml" />
<mimeMap fileExtension=".svgz" mimeType="image/svg+xml" />
<mimeMap fileExtension=".swf" mimeType="application/x-shockwave-flash" />
<mimeMap fileExtension=".t" mimeType="application/x-troff" />
<mimeMap fileExtension=".tar" mimeType="application/x-tar" />
<mimeMap fileExtension=".tcl" mimeType="application/x-tcl" />
<mimeMap fileExtension=".tex" mimeType="application/x-tex" />
<mimeMap fileExtension=".texi" mimeType="application/x-texinfo" />
<mimeMap fileExtension=".texinfo" mimeType="application/x-texinfo" />
<mimeMap fileExtension=".tgz" mimeType="application/x-compressed" />
<mimeMap fileExtension=".thmx" mimeType="application/vnd.ms-officetheme" />
<mimeMap fileExtension=".thn" mimeType="application/octet-stream" />
<mimeMap fileExtension=".tif" mimeType="image/tiff" />
<mimeMap fileExtension=".tiff" mimeType="image/tiff" />
<mimeMap fileExtension=".toc" mimeType="application/octet-stream" />
<mimeMap fileExtension=".tr" mimeType="application/x-troff" />
<mimeMap fileExtension=".trm" mimeType="application/x-msterminal" />
<mimeMap fileExtension=".ts" mimeType="video/vnd.dlna.mpeg-tts" />
<mimeMap fileExtension=".tsv" mimeType="text/tab-separated-values" />
<mimeMap fileExtension=".ttf" mimeType="application/octet-stream" />
<mimeMap fileExtension=".tts" mimeType="video/vnd.dlna.mpeg-tts" />
<mimeMap fileExtension=".txt" mimeType="text/plain" />
<mimeMap fileExtension=".u32" mimeType="application/octet-stream" />
<mimeMap fileExtension=".uls" mimeType="text/iuls" />
<mimeMap fileExtension=".ustar" mimeType="application/x-ustar" />
<mimeMap fileExtension=".vbs" mimeType="text/vbscript" />
<mimeMap fileExtension=".vcf" mimeType="text/x-vcard" />
<mimeMap fileExtension=".vcs" mimeType="text/plain" />
<mimeMap fileExtension=".vdx" mimeType="application/vnd.ms-visio.viewer" />
<mimeMap fileExtension=".vml" mimeType="text/xml" />
<mimeMap fileExtension=".vsd" mimeType="application/vnd.visio" />
<mimeMap fileExtension=".vss" mimeType="application/vnd.visio" />
<mimeMap fileExtension=".vst" mimeType="application/vnd.visio" />
<mimeMap fileExtension=".vsto" mimeType="application/x-ms-vsto" />
<mimeMap fileExtension=".vsw" mimeType="application/vnd.visio" />
<mimeMap fileExtension=".vsx" mimeType="application/vnd.visio" />
```

```
      <mimeMap fileExtension=".vtx" mimeType="application/vnd.visio" />
      <mimeMap fileExtension=".wav" mimeType="audio/wav" />
      <mimeMap fileExtension=".wax" mimeType="audio/x-ms-wax" />
      <mimeMap fileExtension=".wbmp" mimeType="image/vnd.wap.wbmp" />
      <mimeMap fileExtension=".wcm" mimeType="application/vnd.ms-works" />
      <mimeMap fileExtension=".wdb" mimeType="application/vnd.ms-works" />
      <mimeMap fileExtension=".webm" mimeType="video/webm" />
      <mimeMap fileExtension=".wks" mimeType="application/vnd.ms-works" />
      <mimeMap fileExtension=".wm" mimeType="video/x-ms-wm" />
      <mimeMap fileExtension=".wma" mimeType="audio/x-ms-wma" />
      <mimeMap fileExtension=".wmd" mimeType="application/x-ms-wmd" />
      <mimeMap fileExtension=".wmf" mimeType="application/x-msmetafile" />
      <mimeMap fileExtension=".wml" mimeType="text/vnd.wap.wml" />
      <mimeMap fileExtension=".wmlc" mimeType="application/vnd.wap.wmlc" />
      <mimeMap fileExtension=".wmls" mimeType="text/vnd.wap.wmlscript" />
      <mimeMap fileExtension=".wmlsc" mimeType="application/vnd.wap.wmlscriptc" />
      <mimeMap fileExtension=".wmp" mimeType="video/x-ms-wmp" />
      <mimeMap fileExtension=".wmv" mimeType="video/x-ms-wmv" />
      <mimeMap fileExtension=".wmx" mimeType="video/x-ms-wmx" />
      <mimeMap fileExtension=".wmz" mimeType="application/x-ms-wmz" />
      <mimeMap fileExtension=".woff" mimeType="font/x-woff" />
      <mimeMap fileExtension=".wps" mimeType="application/vnd.ms-works" />
      <mimeMap fileExtension=".wri" mimeType="application/x-mswrite" />
      <mimeMap fileExtension=".wrl" mimeType="x-world/x-vrml" />
      <mimeMap fileExtension=".wrz" mimeType="x-world/x-vrml" />
      <mimeMap fileExtension=".wsdl" mimeType="text/xml" />
      <mimeMap fileExtension=".wtv" mimeType="video/x-ms-wtv" />
      <mimeMap fileExtension=".wvx" mimeType="video/x-ms-wvx" />
      <mimeMap fileExtension=".x" mimeType="application/directx" />
      <mimeMap fileExtension=".xaf" mimeType="x-world/x-vrml" />
      <mimeMap fileExtension=".xaml" mimeType="application/xaml+xml" />
      <mimeMap fileExtension=".xap" mimeType="application/x-silverlight-app" />
      <mimeMap fileExtension=".xbap" mimeType="application/x-ms-xbap" />
      <mimeMap fileExtension=".xbm" mimeType="image/x-xbitmap" />
      <mimeMap fileExtension=".xdr" mimeType="text/plain" />
      <mimeMap fileExtension=".xht" mimeType="application/xhtml+xml" />
      <mimeMap fileExtension=".xhtml" mimeType="application/xhtml+xml" />
      <mimeMap fileExtension=".xla" mimeType="application/vnd.ms-excel" />
      <mimeMap fileExtension=".xlam" mimeType="application/vnd.ms-excel.addin.macroEnabled.12" />
      <mimeMap fileExtension=".xlc" mimeType="application/vnd.ms-excel" />
      <mimeMap fileExtension=".xlm" mimeType="application/vnd.ms-excel" />
      <mimeMap fileExtension=".xls" mimeType="application/vnd.ms-excel" />
      <mimeMap fileExtension=".xlsb" mimeType="application/vnd.ms-excel.sheet.binary.macroEnabled.12" />
      <mimeMap fileExtension=".xlsm" mimeType="application/vnd.ms-excel.sheet.macroEnabled.12" />
      <mimeMap fileExtension=".xlsx" mimeType="application/vnd.openxmlformats-officedocument.spreadsheetml.sheet" />
      <mimeMap fileExtension=".xlt" mimeType="application/vnd.ms-excel" />
      <mimeMap fileExtension=".xltm" mimeType="application/vnd.ms-excel.template.macroEnabled.12" />
      <mimeMap fileExtension=".xltx" mimeType="application/vnd.openxmlformats-officedocument.spreadsheetml.template" />
      <mimeMap fileExtension=".xlw" mimeType="application/vnd.ms-excel" />
      <mimeMap fileExtension=".xml" mimeType="text/xml" />
      <mimeMap fileExtension=".xof" mimeType="x-world/x-vrml" />
      <mimeMap fileExtension=".xpm" mimeType="image/x-xpixmap" />
      <mimeMap fileExtension=".xps" mimeType="application/vnd.ms-xpsdocument" />
      <mimeMap fileExtension=".xsd" mimeType="text/xml" />
      <mimeMap fileExtension=".xsf" mimeType="text/xml" />
      <mimeMap fileExtension=".xsl" mimeType="text/xml" />
      <mimeMap fileExtension=".xslt" mimeType="text/xml" />
      <mimeMap fileExtension=".xsn" mimeType="application/octet-stream" />
      <mimeMap fileExtension=".xtp" mimeType="application/octet-stream" />
      <mimeMap fileExtension=".xwd" mimeType="image/x-xwindowdump" />
      <mimeMap fileExtension=".z" mimeType="application/x-compress" />
      <mimeMap fileExtension=".zip" mimeType="application/x-zip-compressed" />
    </staticContent>

    <tracing>

      <traceFailedRequests />
```

```
        <traceProviderDefinitions />

      </tracing>

      <urlCompression />

      <validation />

   </system.webServer>
   <location path="Default Web Site/tpcc">
      <system.webServer>
         <handlers accessPolicy="Read, Execute, Script">
            <remove name="ISAPI-dll" />
            <remove name="CGI-exe" />
            <remove name="StaticFile" />
            <remove name="TRACEVerbHandler" />
            <remove name="SSINC-stm" />
            <remove name="SSINC-shtml" />
            <remove name="SSINC-shtm" />
            <remove name="OPTIONSVerbHandler" />
            <add name="tpcc-isapi" path="*" verb="*" modules="IsapiModule" scriptProcessor="C:\tpcc\tpccisapicom.dll"
resourceType="Unspecified" requireAccess="Execute" preCondition="bitness64" />
         </handlers>
      </system.webServer>
   </location>
   <location path="Default Web Site">
      <system.webServer>
         <httpLogging dontLog="true" />
      </system.webServer>
   </location>

</configuration>""


Microsoft DiskPart version 6.3.9600

Copyright (C) 1999-2013 Microsoft Corporation.
On computer: EINSTEIN

  Disk ###  Status         Size     Free     Dyn  Gpt
  --------  -------------  -------  -------  ---  ---
  Disk 0    Online         2978 GB      0 B        *
  Disk 1    Online         1862 GB      0 B

  Volume ###  Ltr  Label        Fs     Type        Size     Status     Info
  ----------  ---  -----------  -----  ----------  -------  ---------  --------
  Volume 0     D   New Volume   NTFS   Partition   2978 GB  Healthy
  Volume 1     C                NTFS   Partition   1862 GB  Healthy    System
 Controller  PERC H710 Adapter (Slot 4)

Controllers
ID                                    : 0
Status                                : Ok
Name                                  : PERC H710 Adapter
Slot ID                               : PCI Slot 4
State                                 : Ready
Firmware Version                      : 21.2.0-0007
Latest Available Firmware Version     : Not Applicable
Driver Version                        : 6.600.21.08
Minimum Required Driver Version       : Not Applicable
Storport Driver Version               : 6.3.9600.16384
Minimum Required Storport Driver Version  : Not Applicable
Number of Connectors                  : 2
Rebuild Rate                          : 30%
BGI Rate                              : 30%
Check Consistency Rate                : 30%
```

Reconstruct Rate : 30%
Alarm State : Not Applicable
Cluster Mode : Not Applicable
SCSI Initiator ID : Not Applicable
Cache Memory Size : 512 MB
Patrol Read Mode : Auto
Patrol Read State : Stopped
Patrol Read Rate : 30%
Patrol Read Iterations : 21
Abort Check Consistency on Error : Disabled
Allow Revertible Hot Spare and Replace Member : Enabled
Load Balance : Not Applicable
Auto Replace Member on Predictive Failure : Disabled
Redundant Path view : Not Applicable
CacheCade Capable : Yes
Persistent Hot Spare : Disabled
Encryption Capable : Yes
Encryption Key Present : No
Encryption Mode : None
Preserved Cache : Not Applicable
Spin Down Unconfigured Drives : Disabled
Spin Down Hot Spares : Disabled
Spin Down Configured Drives : Disabled
Automatic Disk Power Saving (Idle C) : Disabled
Start Time (HH:MM) : Not Applicable
Time Interval for Spin Up (in Hours) : Not Applicable
T10 Protection Information Capable : No
List of Physical Disks on Controller PERC H710 Adapter (Slot 4)

Controller PERC H710 Adapter (Slot 4)
ID : 0:1:0
Status : Ok
Name : Physical Disk 0:1:0
State : Online
Power Status : Not Applicable
Bus Protocol : SATA
Media : SSD
Part of Cache Pool : Not Applicable
Remaining Rated Write Endurance : 96%
Failure Predicted : No
Revision : D201DL12
Driver Version : Not Applicable
Model Number : Not Applicable
T10 PI Capable : No
Certified : Yes
Encryption Capable : No
Encrypted : Not Applicable
Progress : Not Applicable
Mirror Set ID : Not Applicable
Capacity : 744.63 GB (799535005696 bytes)
Used RAID Disk Space : 744.63 GB (799535005696 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL(tm)
Product ID : INTEL SSDSC2BB800G4T
Serial No. : BTWL4116010Y800RGN
Part Number : CN0VDPRV4811243J0047A02
Negotiated Speed : 3.00 Gbps
Capable Speed : 6.00 Gbps
PCIe Maximum Link Width : Not Applicable
PCIe Negotiated Link Width : Not Applicable
Sector Size : 512B
Device Write Cache : Not Applicable
Manufacture Day : Not Available
Manufacture Week : Not Available
Manufacture Year : Not Available
SAS Address : 4433221107000000

ID : 0:1:1
Status : Ok
Name : Physical Disk 0:1:1
State : Online
Power Status : Not Applicable
Bus Protocol : SATA
Media : SSD
Part of Cache Pool : Not Applicable
Remaining Rated Write Endurance : 96%
Failure Predicted : No
Revision : D201DL12
Driver Version : Not Applicable
Model Number : Not Applicable
T10 PI Capable : No
Certified : Yes
Encryption Capable : No
Encrypted : Not Applicable
Progress : Not Applicable
Mirror Set ID : Not Applicable
Capacity : 744.63 GB (799535005696 bytes)
Used RAID Disk Space : 744.63 GB (799535005696 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL(tm)
Product ID : INTEL SSDSC2BB800G4T
Serial No. : BTWL41160103800RGN
Part Number : CN0VDPRV4811243J0031A02
Negotiated Speed : 3.00 Gbps
Capable Speed : 6.00 Gbps
PCIe Maximum Link Width : Not Applicable
PCIe Negotiated Link Width : Not Applicable
Sector Size : 512B
Device Write Cache : Not Applicable
Manufacture Day : Not Available
Manufacture Week : Not Available
Manufacture Year : Not Available
SAS Address : 4433221106000000

ID : 0:1:2
Status : Ok
Name : Physical Disk 0:1:2
State : Online
Power Status : Spun Up
Bus Protocol : SATA
Media : HDD
Part of Cache Pool : Not Applicable
Remaining Rated Write Endurance : Not Applicable
Failure Predicted : No
Revision : FL1D
Driver Version : Not Applicable
Model Number : Not Applicable
T10 PI Capable : No
Certified : Yes
Encryption Capable : No
Encrypted : Not Applicable
Progress : Not Applicable
Mirror Set ID : Not Applicable
Capacity : 1,862.50 GB (1999844147200 bytes)
Used RAID Disk Space : 1,862.50 GB (1999844147200 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL(tm)
Product ID : TOSHIBA MG03ACA200
Serial No. : 34G2K8Z6F
Part Number : PH0TNTM57557143G1888A00
Negotiated Speed : 3.00 Gbps

Capable Speed : 3.00 Gbps
PCIe Maximum Link Width : Not Applicable
PCIe Negotiated Link Width : Not Applicable
Sector Size : 512B
Device Write Cache : Not Applicable
Manufacture Day : Not Available
Manufacture Week : Not Available
Manufacture Year : Not Available
SAS Address : 4433221105000000

ID : 0:1:3
Status : Ok
Name : Physical Disk 0:1:3
State : Online
Power Status : Spun Up
Bus Protocol : SATA
Media : HDD
Part of Cache Pool : Not Applicable
Remaining Rated Write Endurance : Not Applicable
Failure Predicted : No
Revision : FL1D
Driver Version : Not Applicable
Model Number : Not Applicable
T10 PI Capable : No
Certified : Yes
Encryption Capable : No
Encrypted : Not Applicable
Progress : Not Applicable
Mirror Set ID : Not Applicable
Capacity : 1,862.50 GB (1999844147200 bytes)
Used RAID Disk Space : 1,862.50 GB (1999844147200 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL(tm)
Product ID : TOSHIBA MG03ACA200
Serial No. : 34G3K890F
Part Number : PH0TNTM57557143G187QA00
Negotiated Speed : 3.00 Gbps
Capable Speed : 3.00 Gbps
PCIe Maximum Link Width : Not Applicable
PCIe Negotiated Link Width : Not Applicable
Sector Size : 512B
Device Write Cache : Not Applicable
Manufacture Day : Not Available
Manufacture Week : Not Available
Manufacture Year : Not Available
SAS Address : 4433221104000000

ID : 0:1:6
Status : Non-Critical
Name : Physical Disk 0:1:6
State : Online
Power Status : Not Applicable
Bus Protocol : SATA
Media : SSD
Part of Cache Pool : Not Applicable
Remaining Rated Write Endurance : Not Applicable
Failure Predicted : No
Revision : D2010370
Driver Version : Not Applicable
Model Number : Not Applicable
T10 PI Capable : No
Certified : No
Encryption Capable : No
Encrypted : Not Applicable
Progress : Not Applicable
Mirror Set ID : Not Applicable

Capacity : 744.63 GB (799535005696 bytes)
Used RAID Disk Space : 744.63 GB (799535005696 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID :
Product ID : INTEL SSDSC2BB800G4
Serial No. : CVWL420300DX800RGN
Part Number : Not Available
Negotiated Speed : 6.00 Gbps
Capable Speed : 6.00 Gbps
PCIe Maximum Link Width : Not Applicable
PCIe Negotiated Link Width : Not Applicable
Sector Size : 512B
Device Write Cache : Not Applicable
Manufacture Day : Not Available
Manufacture Week : Not Available
Manufacture Year : Not Available
SAS Address : 4433221101000000

ID : 0:1:7
Status : Non-Critical
Name : Physical Disk 0:1:7
State : Online
Power Status : Not Applicable
Bus Protocol : SATA
Media : SSD
Part of Cache Pool : Not Applicable
Remaining Rated Write Endurance : Not Applicable
Failure Predicted : No
Revision : D2010370
Driver Version : Not Applicable
Model Number : Not Applicable
T10 PI Capable : No
Certified : No
Encryption Capable : No
Encrypted : Not Applicable
Progress : Not Applicable
Mirror Set ID : Not Applicable
Capacity : 744.63 GB (799535005696 bytes)
Used RAID Disk Space : 744.63 GB (799535005696 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID :
Product ID : INTEL SSDSC2BB800G4
Serial No. : CVWL4202043H800RGN
Part Number : Not Available
Negotiated Speed : 6.00 Gbps
Capable Speed : 6.00 Gbps
PCIe Maximum Link Width : Not Applicable
PCIe Negotiated Link Width : Not Applicable
Sector Size : 512B
Device Write Cache : Not Applicable
Manufacture Day : Not Available
Manufacture Week : Not Available
Manufacture Year : Not Available
SAS Address : 4433221100000000
List of Virtual Disks on Controller PERC H710 Adapter (Slot 4)

Controller PERC H710 Adapter (Slot 4)
ID : 0
Status : Ok
Name : Database
State : Ready
Hot Spare Policy violated : Not Applicable
Encrypted : No
Layout : RAID-0
Size : 2,978.50 GB (3198140022784 bytes)

T10 Protection Information Status : No
Associated Fluid Cache State : Not Applicable
Device Name : Windows Disk 0
Bus Protocol : SATA
Media : SSD
Read Policy : Read Ahead
Write Policy : Write Through
Cache Policy : Not Applicable
Stripe Element Size : 64 KB
Disk Cache Policy : Enabled

ID : 1
Status : Ok
Name : system
State : Ready
Hot Spare Policy violated : Not Assigned
Encrypted : No
Layout : RAID-1
Size : 1,862.50 GB (1999844147200 bytes)
T10 Protection Information Status : No
Associated Fluid Cache State : Not Applicable
Device Name : Windows Disk 1
Bus Protocol : SATA
Media : HDD
Read Policy : Read Ahead
Write Policy : Write Through
Cache Policy : Not Applicable
Stripe Element Size : 64 KB
Disk Cache Policy : Enabled
Invalid controller value. Read, controller=1
Valid values for controller are: 0
Invalid controller value. Read, controller=1
Valid values for controller are: 0

System Summary

Software Profile

Systems Management
Name : Dell OpenManage Systems Management Software (64-Bit)
Version : 7.4.0
Description : Systems Management Software
Contains : Apache Tomcat Webserver 7.0.39
: Broadcom SNMP Agent 17.6.4
: Common Storage Module 4.4.0
: Data Engine 7.4.0
: Hardware Application Programming Interface 7.4.0
: Instrumentation Service 7.4.0
: Instrumentation Service Integration Layer 7.4.0
: Intel SNMP Agent 1.118.0
: Inventory Collector 7.4.0
: OMACS 7.4.0
: Operating System Logging 7.4.0
: Oracle Java Runtime Environment 1.7.0_21
: Remote Access Controller Managed Node 7.4.0
: Server Administrator Common Framework 7.4.0
: Server Administrator Core files 7.4.0
: Server Administrator Instrumentation files 7.4.0
: Server Instrumentation SNMP Module 7.4.0
: Server Instrumentation WMI Module 7.4.0
: Storage Management 4.4.0

Operating System

Name : Microsoft Windows Server 2012 R2, Standard x64 Edition
Version : Version 6.3 (Build 9600) (x64) Server Full Installation
System Time : Wed Oct 29 09:02:55 2014
System Bootup Time : Sat Oct 18 23:39:47 2014

System

System
Host Name : EINSTEIN
System Location : Please set the value
Lifecycle Controller : Enabled

Main System Chassis

Chassis Information
Chassis Model : PowerEdge T620
Chassis Service Tag : F2P1K02
Express Service Code : 32814730370
Chassis Lock : Present
Chassis Asset Tag : 540290 4
Node Id : F2P1K02

Remote Access Information
Remote Access Device : iDRAC7 Express
vFlash Media : Absent

Processor 1
Processor Brand : Intel(R) Xeon(R) CPU E5-2670 v2 @ 2.50GHz
Processor Version : Model 62 Stepping 4
Voltage : 1200 mV

Processor 2
Processor Brand : Intel(R) Xeon(R) CPU E5-2670 v2 @ 2.50GHz
Processor Version : Model 62 Stepping 4
Voltage : 1200 mV

Memory
Total Installed Capacity : 131072 MB
Memory Available to the OS : 131027 MB
Total Maximum Capacity : 1572864 MB
Memory Array Count : 1

Memory Array 1
Location : System Board or Motherboard
Use : System Memory
Installed Capacity : 131072 MB
Maximum Capacity : 1572864 MB
Slots Available : 24
Slots Used : 8
ECC Type : Multibit ECC

Slot PCI1
Adapter : [Not Occupied]
Type : PCI E Gen 3
Data Bus Width : 8x or x8
Speed : [Not Obtained, see card documentation]
Slot Length : Long
Voltage Supply : 3.3 Volts

Slot PCI2
Adapter : [Not Occupied]
Type : PCI E Gen 3
Data Bus Width : 16x or x16
Speed : [Not Obtained, see card documentation]
Slot Length : Long

Voltage Supply : 3.3 Volts

Slot PCI3
Adapter : [Not Occupied]
Type : PCI E Gen 2 X8
Data Bus Width : 4x or x4
Speed : [Not Obtained, see card documentation]
Slot Length : Long
Voltage Supply : 3.3 Volts

Slot PCI4
Adapter : PERC H710 Adapter
Type : PCI E Gen 3
Data Bus Width : 16x or x16
Speed : [Not Obtained, see card documentation]
Slot Length : Long
Voltage Supply : 3.3 Volts

Slot PCI5
Adapter : NetXtreme BCM5720 Gigabit Ethernet PCIe
Type : PCI E Gen 3
Data Bus Width : 16x or x16
Speed : [Not Obtained, see card documentation]
Slot Length : Long
Voltage Supply : 3.3 Volts

Slot PCI6
Adapter : [Not Occupied]
Type : PCI E Gen 3
Data Bus Width : 8x or x8
Speed : [Not Obtained, see card documentation]
Slot Length : Long
Voltage Supply : 3.3 Volts

Slot PCI7
Adapter : [Not Occupied]
Type : PCI E Gen 3
Data Bus Width : 16x or x16
Speed : [Not Obtained, see card documentation]
Slot Length : Long
Voltage Supply : 3.3 Volts

BIOS Information
Manufacturer : Dell Inc.
Version : 2.2.2
Release Date : 01/16/2014

Firmware Information
Name : iDRAC7
Version : 1.56.55 (Build 5)

Firmware Information
Name : Lifecycle Controller 2
Version : 1.4.0.128

Remote Access Controller

Remote Access Controller Information
Product : iDRAC7 Express
IP Address : 192.168.0.120
IP Subnet : 255.255.255.0
IP Gateway : 192.168.0.1
IPv6 Address 1 : ::
IPv6 Address 2 : ::
IPv6 Gateway : ::

```
--------------
Network Data
--------------
Network Interface 0
IP Address              : 10.7.161.230
Subnet Mask             : 255.255.252.0
Default Gateway         : 10.7.160.1
MAC Address             : F0-1F-AF-EC-24-C2

Network Interface 1
IP Address              : 192.168.8.8
Subnet Mask             : 255.255.0.0
Default Gateway         : 192.168.5.5
MAC Address             : F0-1F-AF-EC-24-C3

Network Interface 2
IP Address              : [No Value]
MAC Address             : 00-0A-F7-52-7B-D0

Network Interface 3
IP Address              : [No Value]
MAC Address             : 00-0A-F7-52-7B-D1

--------------------
Storage Enclosures
--------------------
Storage Enclosures
Name                    : Backplane
```

APPENDIX F: SOURCE CODE

```
/*c:\original\kit\comclient.cpp*/
// ********************************************************************************
// Copyright(c) 2001-2014 SAP SE or an SAP affiliate company. All rights reserved.
// ********************************************************************************

// Implement COM+ client used by ISAPI dll through ITxn interface.
// The COM+ component is implemented in tpcccom.cpp.


#define _WIN32_DCOM
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include "txn.hpp"
#include "utils.hpp"
#include "tpcccom.h" // Generated by MIDL
#include "tpcccom_i.c"

#define ERROR_FILE                      TPCC_OUTPUT_DIRECTORY "\\com_error.txt"

class COMTxn: public ITxn {
public:
    COMTxn();
    ~COMTxn( void );

    void new_order();
    void payment();
    void order_status();
    void delivery();
    void stock_level();
    void get_maximum();

private:
    void hurl( char *diagnostic, HRESULT hr );

private:
    ITPCCTran * _conn;                  // component object
};


ITxn * new_Txn()
/**************/
{
    return new COMTxn();
}

void free_Txn( ITxn * txn )
/*************************/
{
    COMTxn *  com_txn = (COMTxn *) txn;
    delete com_txn;
}


a_bool init_Txn()
/***************/
{
    return( TRUE );
```

```
}

void fini_Txn()
/*************/
{
}

COMTxn::COMTxn( void )
/********************/
{
   HRESULT          hr;
   IUnknown *       pUnknown;

   // initialize COM for this thread
   hr = CoInitializeEx( NULL, COINIT_MULTITHREADED );
   if( FAILED( hr ) ) {
          hurl( "Unable to initialize COM", hr );
   }

   // load component and get IUnknown object
   hr = CoCreateInstance( CLSID_TPCCTran,
                                  NULL,
//                                CLSCTX_INPROC_SERVER,
//                                CLSCTX_LOCAL_SERVER,
                                  CLSCTX_SERVER,
                                  IID_IUnknown,
                                  (void**)&pUnknown );
   if( FAILED( hr ) ) {
          CoUninitialize();
          hurl( "Unable to create COM coclass with CLSID_TPCCTran. "
               "Ensure tpcccomasa.dll and tpcccomps.dll have been "
               "registered with regsvr32 on both the client and "
               "server machines and that the iASTPCC COM+ application has "
               "been configured with Admistrative Tools > Component Services. ",
               hr );
   }

   // create ITPCCTran object
   hr = pUnknown->QueryInterface( IID_ITPCCTran, (void **)&_conn );
   pUnknown->Release();

   if( FAILED( hr ) ) {
          CoUninitialize();
          hurl( "Failed to create COM coclass with interface ITPCCTran. "
               "The COM component was found, but the transaction object "
               "failed to be created. "
               "Ensure tpcccomps.dll has been registered with "
               "regsvr32 on both the client and server machines.",
               hr );
   }
   _conn->initialization_complete();
}

COMTxn::~COMTxn()
/***************/
{
   _conn->Release();
}

void COMTxn::new_order()
/**********************/
{
   HRESULT          hr;

   hr = _conn->new_order( sizeof( a_new_order ), (UCHAR *)&_params );
   if( FAILED( hr ) ) {
          hurl( "Error on new_order.", hr );
```

```
    }
}

void COMTxn::payment()
/*********************/
{
    HRESULT          hr;

    hr = _conn->payment( sizeof( a_payment ), (UCHAR *)&_params );
    if( FAILED( hr ) ) {
            hurl( "Error on payment.", hr );
    }
}

void COMTxn::order_status()
/*************************/
{
    HRESULT          hr;

    hr = _conn->order_status( sizeof( an_order_status ), (UCHAR *)&_params );
    if( FAILED( hr ) ) {
            hurl( "Error on order_status.", hr );
    }
}

void COMTxn::delivery()
/*********************/
{
    HRESULT          hr;

    hr = _conn->delivery( sizeof( a_delivery ), (UCHAR *)&_params );
    if( FAILED( hr ) ) {
            hurl( "Error on delivery.", hr );
    }
}

void COMTxn::stock_level()
/************************/
{
    HRESULT          hr;

    hr = _conn->stock_level( sizeof( a_stock_level ), (UCHAR *)&_params );
    if( FAILED( hr ) ) {
            hurl( "Error on stock_level.", hr );
    }
}

void COMTxn::get_maximum()
/************************/
{
    HRESULT          hr;

    hr = _conn->get_maximum( sizeof( _params.max_w_id ), (UCHAR *)&_params );
    if( FAILED( hr ) ) {
            hurl( "Error on get_maximum.", hr );
    }
}

#define HURL_STR "%s  Error code is 0x%X (see winerror.h).  "          \
                 "Check " ERROR_FILE " for errors detected by the component."

void COMTxn::hurl( char *diagnostic, HRESULT hr )
/***********************************************/
{
    char * str = (char *)malloc( strlen( diagnostic ) +
                                          strlen( HURL_STR ) + 20 );
```

```
    sprintf( str, HURL_STR, diagnostic, hr );
    throw( str );
}



/*c:\original\kit\odbc.cpp*/
// ********************************************************************************
// Copyright (c) 2001-2014 SAP SE or an SAP affiliate company. All rights reserved.
// ********************************************************************************

// Define ODBC transaction class, inherited from ITxn.


#if !defined( UNIX )
#include <windows.h>          // only for Sleep
#endif
#include <stdio.h>
#include <string.h>
#include <assert.h>
//#include "clibext.h"
#include "txn.hpp"

#if defined( UNIX )
#include "unixodbc.h"
#else
#include "ntodbc.h"
#define delay( ms )  Sleep( ms )
#endif

#define   MIN_BACKOFF       2
#define   MAX_BACKOFF       1024

char * DSN = NULL;

a_bool
init_Txn()
/********/
{
    return TRUE;
}

void
fini_Txn()
/********/
{
}

struct Retry {
    unsigned                dummy;
};

class ODBC: public ITxn  {
public:
    ODBC ( char * source, char * user, char * password );
    ~ODBC( void );
    void new_order();
    void payment();
    void order_status();
    void delivery();
    void stock_level();
    void get_maximum();
    void dump_server_properties( FILE * file );
    void dump_database_properties( FILE * file );
private:
    void transact();
```

```
void hurl();
void hurl( char * diagnostic );
a_bool diagnose( SQLHANDLE henv, SQLHANDLE hdbc, SQLHANDLE hstmt );

// Wrappers for ODBC functions.
void success( RETCODE rc ) {
        if( rc != SQL_SUCCESS ) hurl();
}
SQLHANDLE alloc_handle( short type, SQLHANDLE base ) {
        SQLHANDLE alloc;
        success( SQLAllocHandle( type, base, &alloc ) );
        if( alloc == NULL ) hurl( "null handle allocated" );
        return alloc;
}
void bindParam(
        short       index,
        short       ctype,
        short       sqltype,
        int         prec,
        short       scale,
        void *      buffer,
        int         size ) {
        success( SQLBindParameter( _hstmt, index, SQL_PARAM_INPUT,
           ctype, sqltype, prec, scale, buffer, size, NULL ) );
}
void bindParam( short index, short ctype, short sqltype, void * buffer ) {
        bindParam( index, ctype, sqltype, 0, 0, buffer, 0 );
}
void bindParam( short index, unsigned long size, void * buffer ) {
        bindParam( index, SQL_C_CHAR, SQL_CHAR, size, 0, buffer, size );
}
void bindCol( short col, short type, void * value, int size = 0 ) {
        success( SQLBindCol( _hstmt, col, type, value, size, NULL) );
}
void bindColInd( short col, short type, void * value, an_indicator * indic, int size = 0 ) {
        success( SQLBindCol( _hstmt, col, type, value, size, indic) );
}
void exec( char * stmt ) {
        RETCODE rc = SQLExecDirect( _hstmt, (SQLCHAR *) stmt, SQL_NTS );
        if( rc != SQL_SUCCESS && rc != SQL_SUCCESS_WITH_INFO ) hurl();
}
void exec() {
        RETCODE rc = SQLExecute( _hstmt );
        if( rc != SQL_SUCCESS && rc != SQL_SUCCESS_WITH_INFO ) hurl();
}
void prepare( char * stmt ) {
        RETCODE rc = SQLPrepare( _hstmt, (SQLCHAR *) stmt, SQL_NTS );
        if( rc != SQL_SUCCESS && rc != SQL_SUCCESS_WITH_INFO ) hurl();
}
RETCODE fetch() {
        RETCODE rc = SQLFetch( _hstmt );
        if ( rc != SQL_SUCCESS && rc != SQL_NO_DATA ) {
           SQLRETURN          rc2;
           unsigned char   Msg[500];
           unsigned char   SqlState[6];
           SQLINTEGER         NativeError;
           short       MsgLen;
           int                    i = 1;
           int                    numrecs = 0;
           int                    colnum = 0;

           // Error handling code here is not correct; however, it seems to
           // allow the code in diagnose() to report MS SQL errors better.
           SQLGetDiagField(SQL_HANDLE_STMT,_hstmt,1,SQL_DIAG_NUMBER,&numrecs,sizeof(numrecs),NULL);
           SQLGetDiagField(SQL_HANDLE_STMT,_hstmt,1,SQL_DIAG_COLUMN_NUMBER,&colnum,sizeof(colnum),NULL);
           while( (rc2 = SQLGetDiagRec(SQL_HANDLE_STMT, _hstmt, i, SqlState,
```

```
                    &NativeError, Msg, sizeof(Msg), &MsgLen) ) != SQL_NO_DATA ) {
                    // This string is not currently displayed.
                    Msg[MsgLen] = '\0';
                    i++;
                }
                hurl();
            }
            return( rc );
    }
    RETCODE fetch( char * stmt ) {
            RETCODE rc;
            exec( stmt ); rc = fetch(); SQLFreeStmt( _hstmt, SQL_CLOSE );
            return( rc );
    }
    void setStmtAttr( long attr, void * value, long length ) {
            success( SQLSetStmtAttr( _hstmt, attr, value, length ) );
    }
    void setRowDesc( void * desc ) {
            setStmtAttr( SQL_ATTR_APP_ROW_DESC, desc, SQL_IS_POINTER );
    }
private:
    char            _diagnostic[400];
    char *          _context;
    unsigned long   _error;
    unsigned long   _backoff;
    Retry           _retry;

    SQLHENV                 _henv;
    SQLHDBC                 _hdbc;
    SQLHSTMT                _hstmt;             // the current hstmt

    // new order specific fields
    SQLHSTMT                _hstmt_new_order;
    SQLHDESC                _desc_new_order_cols1;
    SQLHDESC                _desc_new_order_cols2;
    SQLUINTEGER             _no_bind_offset;
    short           _no_ol_li_no;
    char            _no_ol_i_name[I_NAME_LEN+1];
    char            _no_ol_brand_generic[BRAND_LEN+1];
    double          _no_ol_i_price;
    double          _no_ol_amount;
    a_quantity              _no_ol_stock;
    // payment specific fields
    SQLHSTMT                _hstmt_payment;
    // delivery specific fields
    SQLHSTMT                _hstmt_delivery;
    // order status specific fields
    SQLHSTMT                _hstmt_order_status;
    SQLHDESC                _desc_order_status_cols1;
    SQLHDESC                _desc_order_status_cols2;
    SQLUINTEGER             _rows_fetched;
    SQLUINTEGER             _os_bind_offset;
    // stock level specific fields
    SQLHSTMT                _hstmt_stock_level;
};

// wrapper routine for class constructor
ITxn *
new_Txn( void )
/*************/
{
#if defined( SERVER_ASA )
    char *      datasource      = "TPCC";
    char *      userid  = "DBA";
    char *      password        = "sql";
#elif defined( SERVER_MSSQL )
    char *      datasource = "MSTPCC";
```

```
    char *      userid  = "";       // define in datasource
    char *      password            = "";
#else
    #error "Unknown Server type"
#endif

    if( DSN != NULL ) {
            datasource = DSN;
    }
    return new ODBC( datasource, userid, password );
}

void free_Txn( ITxn * txn )
/************************/
{
    ODBC *  odbc = (ODBC *) txn;
    delete odbc;
}


ODBC::ODBC( char * source, char * user, char * password )
/******************************************************/
{
    RETCODE rc;
    // initialization
    _henv                   = SQL_NULL_HENV;
    _hdbc                   = SQL_NULL_HDBC;
    _hstmt                  = SQL_NULL_HSTMT;
    _hstmt_new_order        = SQL_NULL_HSTMT;
    _hstmt_payment          = SQL_NULL_HSTMT;
    _hstmt_delivery         = SQL_NULL_HSTMT;
    _hstmt_order_status     = SQL_NULL_HSTMT;
    _hstmt_stock_level      = SQL_NULL_HSTMT;
    _desc_new_order_cols1   = SQL_NULL_HDESC;
    _desc_order_status_cols1 = SQL_NULL_HDESC;
    _desc_order_status_cols2 = SQL_NULL_HDESC;

    _context = "init";

    if( SQLAllocHandle( SQL_HANDLE_ENV, SQL_NULL_HANDLE, &_henv ) != SQL_SUCCESS ) {
            hurl( "Unable to allocate environment" );
    }
    SQLSetEnvAttr( _henv, SQL_ATTR_ODBC_VERSION, (SQLPOINTER)SQL_OV_ODBC3, 0 );
    _hdbc = alloc_handle( SQL_HANDLE_DBC, _henv );

#if !defined( NO_DRIVER_MANAGER )
    if( SQLSetConnectOption( _hdbc, SQL_PACKET_SIZE, 4096 ) != SQL_SUCCESS )
        hurl();
#endif
#if defined( SERVER_ASA )
    if( SQLSetConnectOption( _hdbc, SQL_AUTOCOMMIT, SQL_AUTOCOMMIT_OFF ) != SQL_SUCCESS )
        hurl();
#endif
    if( SQLSetConnectOption( _hdbc, SQL_TXN_ISOLATION, SQL_TXN_READ_COMMITTED ) != SQL_SUCCESS )
        hurl();

    char connstr[500];
    sprintf( connstr, "DSN=%s;UID=%s;PWD=%s%s",
            source, user, password, ";PREFETCHROWS=16" );
    _context = "connect";
    rc = SQLDriverConnect( _hdbc, 0,
                (unsigned char *) connstr, (SQLSMALLINT)strlen( connstr ),
                NULL, 0, NULL,
                SQL_DRIVER_NOPROMPT );
    if( rc != SQL_SUCCESS && rc != SQL_SUCCESS_WITH_INFO )
            hurl();
```

```
   _context = "init after con";

#if defined( SERVER_MSSQL )
   _hstmt = alloc_handle( SQL_HANDLE_STMT, _hdbc );
   // T-SQL options for ASE, MS
   exec( "set nocount on set XACT_ABORT ON" );
   SQLFreeHandle(SQL_HANDLE_STMT, _hstmt);
#endif
}


ODBC::~ODBC()
/***********/
{
   _context = "fini";
   // Descriptors are automatically released when the connection is dropped.
   SQLFreeHandle( SQL_HANDLE_STMT, _hstmt_new_order );
   SQLFreeHandle( SQL_HANDLE_STMT, _hstmt_payment );
   SQLFreeHandle( SQL_HANDLE_STMT, _hstmt_delivery );
   SQLFreeHandle( SQL_HANDLE_STMT, _hstmt_order_status );
   SQLFreeHandle( SQL_HANDLE_STMT, _hstmt_stock_level );
   SQLDisconnect( _hdbc );
   SQLFreeHandle( SQL_HANDLE_DBC, _hdbc );
   if( _henv != SQL_NULL_HENV ) SQLFreeEnv( _henv );
}

a_bool
ODBC::diagnose( SQLHANDLE henv, SQLHANDLE hdbc, SQLHANDLE hstmt )
/***************************************************************/
{
   SQLRETURN         rc = 1;
   SQLCHAR           sqlstate[6] = "";
   SQLINTEGER        error = 0;
   SQLSMALLINT       msglen;


   if( hstmt == NULL && hdbc == NULL && henv == NULL ) return FALSE;

   _diagnostic[0] = '\0';

   if( hstmt != NULL ) {
         rc = SQLGetDiagRec( SQL_HANDLE_STMT, hstmt, 1, sqlstate, &error,
                  (SQLCHAR *) _diagnostic, sizeof(_diagnostic), &msglen );
   }
   if( rc != 0 && hdbc != NULL ) {
         rc = SQLGetDiagRec( SQL_HANDLE_DBC, hdbc, 1, sqlstate, &error,
                  (SQLCHAR *) _diagnostic, sizeof(_diagnostic), &msglen );
   }
   if( rc != 0 && henv != NULL ) {
         rc = SQLGetDiagRec( SQL_HANDLE_ENV, henv, 1, sqlstate, &error,
                  (SQLCHAR *) _diagnostic, sizeof(_diagnostic), &msglen );
   }
   SQLFreeStmt( _hstmt, SQL_CLOSE );
   if( rc == 0 ) {
         if( error == 1205
         ||  error == -29000  // RAISERROR issued after deadlock detected
         ||  error == -306
         ||  error == -307
         || (error == 7312 && strstr(_diagnostic, "Timeout expired")) ) {
            if( _backoff <= MAX_BACKOFF ) {
#if defined( SERVER_MSSQL )
                  SQLEndTran( SQL_HANDLE_DBC, hdbc, SQL_ROLLBACK );
#endif
                  delay( _backoff );
                  _backoff *= 2;
                  throw _retry;
            } else {
```

```
                        _backoff = MIN_BACKOFF;
                        throw _retry;
                    }
                }
    }
    // Got an unexpected error.  Rollback and throw exception.
    SQLEndTran( SQL_HANDLE_DBC, hdbc, SQL_ROLLBACK );
    if( msglen + 40 < sizeof( _diagnostic ) ) {
            sprintf( _diagnostic + msglen,
                    " [%d][%s] (in %s)",
                        error,
                        sqlstate,
                        _context );
    } else {
            _diagnostic[msglen] = '\0';
    }
    hurl( _diagnostic );
    return( FALSE );      // will never get here
}


void
ODBC::hurl()
/**********/
{
    diagnose( _henv, _hdbc, _hstmt );

    hurl( "An error was detected, but attempts to get diagnostics failed" );
}

void
ODBC::hurl( char * diagnostic )
/*****************************/
{
    // strdup the string since to ensure the string is still valid where caught
    // hurl can be called in the constructor, in which case, class buffers
    // would not otherwise be valid when caught
    throw strdup( diagnostic );
}

void
ODBC::stock_level()
/*****************/
{
    _context = "stock_level";
    a_stock_level & p = _params.stock_level;
    if( (_hstmt = _hstmt_stock_level) == SQL_NULL_HSTMT ) {
            _hstmt = _hstmt_stock_level = alloc_handle( SQL_HANDLE_STMT, _hdbc );
            unsigned i = 0;
            bindParam( ++i, SQL_C_SSHORT,   SQL_SMALLINT, &p.w_id );
            bindParam( ++i, SQL_C_UTINYINT, SQL_TINYINT,  &p.d_id );
            bindParam( ++i, SQL_C_SSHORT,   SQL_SMALLINT, &p.threshold );
            bindCol( 1, SQL_C_SLONG, &p.low_stock );
            prepare( "{call tpcc_stocklevel(?,?,?)}" );
    }

    transact();
    p.exec_status_code = eOK;
}

void
ODBC::new_order()
/***************/
{
    _context = "new_order";
    a_new_order & p = _params.new_order;
    char stmt[] =
```

```
      "{call tpcc_neworder(?,?,?,?,?,"
      "?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,"
      "?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,"
      "?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)}";
   if( (_hstmt = _hstmt_new_order) == SQL_NULL_HSTMT ) {
         _hstmt = _hstmt_new_order = alloc_handle( SQL_HANDLE_STMT, _hdbc );
         prepare( stmt );
         unsigned i;
         bindParam( i=1, SQL_C_SSHORT,   SQL_SMALLINT, &p.w_id );
         bindParam( ++i, SQL_C_UTINYINT, SQL_TINYINT,  &p.d_id );
         bindParam( ++i, SQL_C_SLONG,    SQL_INTEGER,  &p.c_id );
         bindParam( ++i, SQL_C_UTINYINT, SQL_TINYINT,  &p.o_ol_cnt );
         bindParam( ++i, SQL_C_UTINYINT, SQL_TINYINT,  &p.o_all_local );
         for( unsigned j = 0; j< MAX_OL_ITEMS; j++ ) {
            bindParam( ++i, SQL_C_SLONG,  SQL_INTEGER,  &p.ol[j].ol_i_id );
            bindParam( ++i, SQL_C_SSHORT, SQL_SMALLINT, &p.ol[j].ol_supply_w_id );
            bindParam( ++i, SQL_C_SSHORT, SQL_SMALLINT, &p.ol[j].ol_quantity );
         }

#if defined( SERVER_MSSQL )
         _desc_new_order_cols1 = alloc_handle( SQL_HANDLE_DESC, _hdbc );
         setRowDesc( _desc_new_order_cols1 );
         setStmtAttr( SQL_ATTR_ROW_BIND_OFFSET_PTR, &_no_bind_offset, SQL_IS_POINTER );

         // 1st MS result set (order items)
         bindCol( i=1, SQL_C_CHAR,   &p.ol[0].ol_i_name, sizeof(p.ol[0].ol_i_name) );
         bindCol( ++i, SQL_C_SSHORT, &p.ol[0].ol_stock );
         bindCol( ++i, SQL_C_CHAR,   &p.ol[0].ol_brand_generic, sizeof(p.ol[0].ol_brand_generic) );
         bindCol( ++i, SQL_C_DOUBLE, &p.ol[0].ol_i_price );
         bindCol( ++i, SQL_C_DOUBLE, &p.ol[0].ol_amount );

         // 2nd MS result set (general order data)
         _desc_new_order_cols2 = alloc_handle( SQL_HANDLE_DESC, _hdbc );
         setRowDesc( _desc_new_order_cols2 );
#endif

         bindCol( i=1, SQL_C_DOUBLE, &p.w_tax );
         bindCol( ++i, SQL_C_DOUBLE, &p.d_tax );
         bindCol( ++i, SQL_C_SLONG,  &p.o_id );

         bindCol( ++i, SQL_C_CHAR,   &p.c_last, sizeof(p.c_last) );
         bindCol( ++i, SQL_C_DOUBLE, &p.c_discount );
         bindCol( ++i, SQL_C_CHAR,   &p.c_credit, sizeof(p.c_credit) );
         bindCol( ++i, SQL_C_TYPE_TIMESTAMP, &p.o_entry_d );
         bindCol( ++i, SQL_C_SLONG,  &p.o_commit_flag );

#if defined( SERVER_ASA )
         // in ASA, all one result set (general order data in each row)

         // ol_li_no seems to be unused...
         bindCol( ++i, SQL_C_SSHORT, &_no_ol_li_no );

         bindCol( ++i, SQL_C_CHAR,   &_no_ol_i_name, sizeof(_no_ol_i_name) );
         bindCol( ++i, SQL_C_SSHORT, &_no_ol_stock );
         bindCol( ++i, SQL_C_CHAR,   &_no_ol_brand_generic, sizeof(_no_ol_brand_generic) );
         bindCol( ++i, SQL_C_DOUBLE, &_no_ol_i_price );
         bindCol( ++i, SQL_C_DOUBLE, &_no_ol_amount );
#endif
   }

   _backoff = MIN_BACKOFF;

   // check whether any order lines are for a remote warehouse
   p.o_all_local = 1;
   unsigned i;
   for( i = 0; i < p.o_ol_cnt; i++ ) {
      if (p.ol[i].ol_supply_w_id != p.w_id) {
```

```
            p.o_all_local = 0; // at least one remote warehouse
                break;
        }
    }
    // Set unused parameters to zero
    unsigned j;
    for( j = p.o_ol_cnt; j < MAX_OL_ITEMS; j++ ) {
            p.ol[j].ol_i_id = 0;
            p.ol[j].ol_supply_w_id = 0;
            p.ol[j].ol_quantity = 0;
    }
    for( ;; ) {
        try {
            exec();
                // Get order line results
                #if defined( SERVER_MSSQL )
                        setRowDesc( _desc_new_order_cols1 );
                #endif
            p.total_amount = 0;
                p.o_commit_flag = 1;
            for( i = 0; i < p.o_ol_cnt; i++ ) {
                        #if defined( SERVER_ASA )
                            if( fetch() != SQL_SUCCESS ) break;
                            p.ol[i].ol_li_no = _no_ol_li_no;
                            strcpy( p.ol[i].ol_i_name, _no_ol_i_name );
                            strcpy( p.ol[i].ol_brand_generic, _no_ol_brand_generic );
                            p.ol[i].ol_i_price = _no_ol_i_price;
                            p.ol[i].ol_amount = _no_ol_amount;
                            p.ol[i].ol_stock = _no_ol_stock;
                            p.total_amount += _no_ol_amount;
                        #else
                            // each line item is a seperate result set
                            _no_bind_offset = i * sizeof(p.ol[0]);
                            fetch();
                            p.total_amount += p.ol[i].ol_amount;
                            if( SQLMoreResults( _hstmt ) == SQL_ERROR ) hurl();
                        #endif
            }
                #if defined( SERVER_MSSQL )
                        // general order data is also a result set
                        setRowDesc( _desc_new_order_cols2 );
                        fetch();
                #endif
            break;
        } catch( Retry ) {
                ;
        }
    }

    if( p.o_commit_flag == 1 ) {
            SQLFreeStmt( _hstmt, SQL_CLOSE );
            p.total_amount *= ((1 + p.w_tax + p.d_tax)*(1 - p.c_discount));
            p.exec_status_code = eOK;
    } else {
            // Resume procedure to cause rollback
            #if defined( SERVER_ASA )
                if ( SQLMoreResults(_hstmt) == SQL_ERROR ) hurl();
            #endif
            SQLFreeStmt( _hstmt, SQL_CLOSE );
            p.exec_status_code = eInvalidItem;
    }
}

void
ODBC::payment()
/*************/
{
```

```
    a_payment & p = _params.payment;

    _context = "payment";
    if( (_hstmt = _hstmt_payment) == SQL_NULL_HSTMT ) {
            _hstmt = _hstmt_payment = alloc_handle( SQL_HANDLE_STMT, _hdbc );
            unsigned i;
            bindParam( i=1, SQL_C_SSHORT,   SQL_SMALLINT, &p.w_id );
            bindParam( ++i, SQL_C_SSHORT,   SQL_SMALLINT, &p.c_w_id );
            bindParam( ++i, SQL_C_DOUBLE,   SQL_NUMERIC, 6, 2, &p.h_amount, 0 );
            bindParam( ++i, SQL_C_UTINYINT, SQL_TINYINT,  &p.d_id );
            bindParam( ++i, SQL_C_UTINYINT, SQL_TINYINT,  &p.c_d_id );
            bindParam( ++i, SQL_C_SLONG,    SQL_INTEGER,  &p.c_id );
            bindParam( ++i, sizeof(p.c_last), &p.c_last );

            bindCol( i=1, SQL_C_SLONG,  &p.c_id );
            bindCol( ++i, SQL_C_CHAR,   &p.c_last,     sizeof(p.c_last) );
            bindCol( ++i, SQL_C_TYPE_TIMESTAMP, &p.h_date );
            bindCol( ++i, SQL_C_CHAR,   &p.w_street_1, sizeof(p.w_street_1) );
            bindCol( ++i, SQL_C_CHAR,   &p.w_street_2, sizeof(p.w_street_2) );
            bindCol( ++i, SQL_C_CHAR,   &p.w_city,     sizeof(p.w_city) );
            bindCol( ++i, SQL_C_CHAR,   &p.w_state,    sizeof(p.w_state) );
            bindCol( ++i, SQL_C_CHAR,   &p.w_zip,      sizeof(p.w_zip) );
            bindCol( ++i, SQL_C_CHAR,   &p.d_street_1, sizeof(p.d_street_1) );
            bindCol( ++i, SQL_C_CHAR,   &p.d_street_2, sizeof(p.d_street_2) );
            bindCol( ++i, SQL_C_CHAR,   &p.d_city,     sizeof(p.d_city) );
            bindCol( ++i, SQL_C_CHAR,   &p.d_state,    sizeof(p.d_state) );
            bindCol( ++i, SQL_C_CHAR,   &p.d_zip,      sizeof(p.d_zip) );
            bindCol( ++i, SQL_C_CHAR,   &p.c_first,    sizeof(p.c_first) );
            bindCol( ++i, SQL_C_CHAR,   &p.c_middle,   sizeof(p.c_middle) );
            bindCol( ++i, SQL_C_CHAR,   &p.c_street_1, sizeof(p.c_street_1) );
            bindCol( ++i, SQL_C_CHAR,   &p.c_street_2, sizeof(p.c_street_2) );
            bindCol( ++i, SQL_C_CHAR,   &p.c_city,     sizeof(p.c_city) );
            bindCol( ++i, SQL_C_CHAR,   &p.c_state,    sizeof(p.c_state) );
            bindCol( ++i, SQL_C_CHAR,   &p.c_zip,      sizeof(p.c_zip) );
            bindCol( ++i, SQL_C_CHAR,   &p.c_phone,    sizeof(p.c_phone) );
            bindCol( ++i, SQL_C_TYPE_TIMESTAMP, &p.c_since );
            bindCol( ++i, SQL_C_CHAR,   &p.c_credit,   sizeof(p.c_credit) );
            bindCol( ++i, SQL_C_DOUBLE, &p.c_credit_lim );
            bindCol( ++i, SQL_C_DOUBLE, &p.c_discount );
            bindCol( ++i, SQL_C_DOUBLE, &p.c_balance );
            bindCol( ++i, SQL_C_CHAR,   &p.c_data,     sizeof(p.c_data) );
            prepare( "{call tpcc_payment(?,?,?,?,?,?,?)}" );
    }

    if( p.c_id != 0 ) p.c_last[0] = 0;
    transact();
    if( p.c_id == 0 ) hurl( "Invalid customer" );
    p.exec_status_code = eOK;
}

void
ODBC::order_status()
/******************/
{
    _context = "order_status";
    an_order_status & p = _params.order_status;
    if( (_hstmt = _hstmt_order_status) == SQL_NULL_HSTMT ) {
            _hstmt = _hstmt_order_status = alloc_handle( SQL_HANDLE_STMT, _hdbc );
            unsigned i;
            bindParam( i=1, SQL_C_SSHORT,   SQL_SMALLINT, &p.w_id );
            bindParam( ++i, SQL_C_UTINYINT, SQL_TINYINT,  &p.d_id );
            bindParam( ++i, SQL_C_SLONG,    SQL_INTEGER,  &p.c_id );
            bindParam( ++i, sizeof(p.c_last), &p.c_last );

            _desc_order_status_cols1 = alloc_handle( SQL_HANDLE_DESC, _hdbc );
            setRowDesc( _desc_order_status_cols1 );
            setStmtAttr( SQL_ATTR_ROW_BIND_TYPE, (SQLPOINTER) sizeof(p.ol[0]), 0 );
```

```
        #if defined( SERVER_ASA )
            // Attempting to fetch multiple rows give "function sequence error"
            // with MS SQL.
            setStmtAttr( SQL_ATTR_ROWS_FETCHED_PTR, &_rows_fetched, 0 );
            setStmtAttr( SQL_ATTR_ROW_ARRAY_SIZE, (SQLPOINTER)MAX_OL_ITEMS, 0 );
        #else
            setStmtAttr( SQL_ATTR_ROW_BIND_OFFSET_PTR, &_os_bind_offset, SQL_IS_POINTER );
        #endif

        bindCol( i=1, SQL_C_SSHORT, &p.ol[0].ol_supply_w_id );
        bindCol( ++i, SQL_C_SLONG,  &p.ol[0].ol_i_id );
        bindCol( ++i, SQL_C_SSHORT, &p.ol[0].ol_quantity );
        bindCol( ++i, SQL_C_DOUBLE, &p.ol[0].ol_amount );
        bindColInd( ++i, SQL_C_TYPE_TIMESTAMP, &p.ol[0].ol_delivery_d,
                         &p.ol[0].ind_delivery_d );

        _desc_order_status_cols2 = alloc_handle( SQL_HANDLE_DESC, _hdbc );
        setRowDesc( _desc_order_status_cols2 );
        bindCol( i=1, SQL_C_SLONG,  &p.c_id );
        bindCol( ++i, SQL_C_CHAR,   &p.c_last,   sizeof(p.c_last) );
        bindCol( ++i, SQL_C_CHAR,   &p.c_first,  sizeof(p.c_first) );
        bindCol( ++i, SQL_C_CHAR,   &p.c_middle, sizeof(p.c_middle) );
        bindCol( ++i, SQL_C_TYPE_TIMESTAMP, &p.o_entry_d );
        bindColInd( ++i, SQL_C_SSHORT, &p.o_carrier_id, &p.ind_carrier_id );
        bindCol( ++i, SQL_C_DOUBLE, &p.c_balance );
        bindCol( ++i, SQL_C_SLONG,  &p.o_id );
        prepare( "{call tpcc_orderstatus(?,?,?,?)}" );
    }

    _backoff = MIN_BACKOFF;
    if( p.c_id != 0 ) p.c_last[0] = 0;
    for( ;; ) {
        try {
                setRowDesc( _desc_order_status_cols1 );
            exec();
                #if defined( SERVER_ASA )
                        fetch();
                #else
                        _rows_fetched = 0;
                        for( ;; ) {
                            _os_bind_offset = _rows_fetched * sizeof(p.ol[0]);
                            RETCODE rc = fetch();
                            if( rc != SQL_SUCCESS ) break;
                            ++_rows_fetched;
                        }
                #endif
            p.o_ol_cnt = (a_ol_id) _rows_fetched;
            if (p.o_ol_cnt != 0) {
                        setRowDesc( _desc_order_status_cols2 );
                if ( SQLMoreResults(_hstmt) == SQL_ERROR ) hurl();
                        fetch();
            }
            break;
        } catch( Retry ) {
                ;
        }
    }
    SQLFreeStmt( _hstmt, SQL_CLOSE );
    if ( p.o_ol_cnt == 0 ) hurl( "no such order" );
    if ( p.c_id == 0 && p.c_last[0] == 0 ) hurl( "invalid customer" );
    p.exec_status_code = eOK;
}

void
ODBC::delivery()
/**************/
{
```

```
    _context = "delivery";
    a_delivery & p = _params.delivery;
    if( (_hstmt = _hstmt_delivery) == SQL_NULL_HSTMT ) {
            _hstmt = _hstmt_delivery = alloc_handle( SQL_HANDLE_STMT, _hdbc );
            unsigned i = 0;
            bindParam( ++i, SQL_C_SSHORT, SQL_SMALLINT, &p.w_id );
            bindParam( ++i, SQL_C_SSHORT, SQL_SMALLINT, &p.o_carrier_id );
            for( i = 0; i < 10; i++ ) {
               bindCol( i+1, SQL_C_SLONG, &p.o_id[i] );
            }
            prepare( "{call tpcc_delivery(?,?)}" );
    }

    transact();
    p.exec_status_code = eOK;
}

void
ODBC::get_maximum()
/*****************/
{
    _context = "get_maximum";
    // only called once, so no need keep prepared statement
    _hstmt = alloc_handle( SQL_HANDLE_STMT, _hdbc );
    bindCol( 1, SQL_C_SSHORT, &_params.max_w_id );
    prepare( "{call tpcc_maximum}" );
    transact();
    SQLFreeHandle( SQL_HANDLE_STMT, _hstmt );
}

void
ODBC::dump_server_properties( FILE * file )
/*****************************************/
{
    long prop_id;
    char prop_name[256];
    char prop_value[256];
    an_indicator ind;

    if( file == NULL ) {
            return;
    }
    _context = "dump_server_properties";
    _hstmt = alloc_handle( SQL_HANDLE_STMT, _hdbc );
    bindCol( 1, SQL_C_SLONG, &prop_id, sizeof(prop_id) );
    bindCol( 2, SQL_C_CHAR, prop_name, sizeof(prop_name) );
    bindColInd( 4, SQL_C_CHAR, prop_value, &ind, sizeof(prop_value) );
    prepare( "{call sa_eng_properties}" );
    try {
            exec();
            for( ;; ) {
               memset( &prop_id, 0, sizeof(prop_id) );
               memset( &prop_name, 0, sizeof(prop_name) );
               memset( &prop_value, 0, sizeof(prop_value) );
               ind = 0;
               if( fetch() == SQL_NO_DATA ) {
                       break;
               }
               fprintf( file, "%d,%s,%s\n", prop_id, prop_name, ( ind < 0 ? "NULL" : prop_value ) );
            }
            fflush( file );
    } catch( Retry ) {
            ;
    }
    SQLFreeStmt( _hstmt, SQL_CLOSE );
}
```

```
void
ODBC::dump_database_properties( FILE * file )
/*****************************************/
{
    long prop_id;
    char prop_name[256];
    char prop_value[256];
    an_indicator ind;

    if( file == NULL ) {
            return;
    }
    _context = "dump_database_properties";
    _hstmt = alloc_handle( SQL_HANDLE_STMT, _hdbc );
    bindCol( 2, SQL_C_SLONG, &prop_id, sizeof(prop_id) );
    bindCol( 3, SQL_C_CHAR, prop_name, sizeof(prop_name) );
    bindColInd( 5, SQL_C_CHAR, prop_value, &ind, sizeof(prop_value) );
    prepare( "{call sa_db_properties}" );
    try {
            exec();
            for( ;; ) {
                memset( &prop_id, 0, sizeof(prop_id) );
                memset( &prop_name, 0, sizeof(prop_name) );
                memset( &prop_value, 0, sizeof(prop_value) );
                ind = 0;
                if( fetch() == SQL_NO_DATA ) {
                        break;
                }
                fprintf( file, "%d,%s,%s\n", prop_id, prop_name, ( ind < 0 ? "NULL" : prop_value ) );
            }
            fflush( file );
    } catch( Retry ) {
            ;
    }
    SQLFreeStmt( _hstmt, SQL_CLOSE );
}

void
ODBC::transact()
/**************/
{
    _backoff = MIN_BACKOFF;
    for( ;; ) {
        try {
                exec();
            fetch();
                SQLFreeStmt( _hstmt, SQL_CLOSE );
            break;
        } catch( Retry ) {
                ;
        }
    }
}




/*c:\original\kit\random.cpp*/
/* A C-program for MT19937: Integer version (1998/4/6)          */
/*  genrand() generates one pseudorandom unsigned integer (32bit) */
/* which is uniformly distributed among 0 to 2^32-1  for each     */
/* call. sgenrand(seed) set initial values to the working area    */
/* of 624 words. Before genrand(), sgenrand(seed) must be         */
/* called once. (seed is any 32-bit integer except for 0).        */
/*   Coded by Takuji Nishimura, considering the suggestions by    */
/* Topher Cooper and Marc Rieffel in July-Aug. 1997.              */
```

```
/* This library is free software; you can redistribute it and/or   */
/* modify it under the terms of the GNU Library General Public     */
/* License as published by the Free Software Foundation; either    */
/* version 2 of the License, or (at your option) any later         */
/* version.                                                        */
/* This library is distributed in the hope that it will be useful, */
/* but WITHOUT ANY WARRANTY; without even the implied warranty of  */
/* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.            */
/* See the GNU Library General Public License for more details.    */
/* You should have received a copy of the GNU Library General      */
/* Public License along with this library; if not, write to the    */
/* Free Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA   */
/* 02111-1307  USA                                                 */

/* Copyright (C) 1997 Makoto Matsumoto and Takuji Nishimura.       */
/* When you use this, send an email to: matumoto@math.keio.ac.jp   */
/* with an appropriate reference to your work.                     */

/* REFERENCE                                                       */
/* M. Matsumoto and T. Nishimura,                                  */
/* "Mersenne Twister: A 623-Dimensionally Equidistributed Uniform  */
/* Pseudo-Random Number Generator",                                */
/* ACM Txns on Modeling and Computer Simulation,                   */
/* Vol. 8, No. 1, January 1998, pp 3--30.                          */

/* Modified for iAnywhere TPCC kit to pass in seed_info            */
/* instead of using static variables mt and mti                    */

#include <stdio.h>
#include <stdlib.h>
#include "random.hpp"


/* Period parameters */
#define N GENRAND_SEED_N
#define M 397
#define MATRIX_A 0x9908b0df   /* constant vector a */
#define UPPER_MASK 0x80000000 /* most significant w-r bits */
#define LOWER_MASK 0x7fffffff /* least significant r bits */

/* Tempering parameters */
#define TEMPERING_MASK_B 0x9d2c5680
#define TEMPERING_MASK_C 0xefc60000
#define TEMPERING_SHIFT_U(y)  (y >> 11)
#define TEMPERING_SHIFT_S(y)  (y << 7)
#define TEMPERING_SHIFT_T(y)  (y << 15)
#define TEMPERING_SHIFT_L(y)  (y >> 18)

/* initializing the array with a NONZERO seed */
void sgenrand( unsigned long seed, a_seed_info *seed_info )
/*****************************************************/
{
    /* setting initial seeds to mt[N] using         */
    /* the generator Line 25 of Table 1 in          */
    /* [KNUTH 1981, The Art of Computer Programming */
    /*    Vol. 2 (2nd Ed.), pp102]                  */
    unsigned long * mt  = seed_info->mt;
    int                     & mti = seed_info->mti;

    mt[0]= seed & 0xffffffff;
    for (mti=1; mti<N; mti++)
        mt[mti] = (69069 * mt[mti-1]) & 0xffffffff;
}

unsigned long genrand( a_seed_info *seed_info )
/*****************************************/
{
```

```
    unsigned long y;
    static unsigned long mag01[2]={0x0, MATRIX_A};
    /* mag01[x] = x * MATRIX_A  for x=0,1 */

    unsigned long * mt  = seed_info->mt;
    int                     & mti = seed_info->mti;

    if( mti == 0 || mti > N ) {
            printf( "internal error: genrand called before sgenrand\n" );
            exit( 1 );
    }

    if (mti >= N) { /* generate N words at one time */
        int kk;

        for (kk=0;kk<N-M;kk++) {
            y = (mt[kk]&UPPER_MASK)|(mt[kk+1]&LOWER_MASK);
            mt[kk] = mt[kk+M] ^ (y >> 1) ^ mag01[y & 0x1];
        }
        for (;kk<N-1;kk++) {
            y = (mt[kk]&UPPER_MASK)|(mt[kk+1]&LOWER_MASK);
            mt[kk] = mt[kk+(M-N)] ^ (y >> 1) ^ mag01[y & 0x1];
        }
        y = (mt[N-1]&UPPER_MASK)|(mt[0]&LOWER_MASK);
        mt[N-1] = mt[M-1] ^ (y >> 1) ^ mag01[y & 0x1];

        mti = 0;
    }

    y = mt[mti++];
    y ^= TEMPERING_SHIFT_U(y);
    y ^= TEMPERING_SHIFT_S(y) & TEMPERING_MASK_B;
    y ^= TEMPERING_SHIFT_T(y) & TEMPERING_MASK_C;
    y ^= TEMPERING_SHIFT_L(y);

    return y;
}



/*c:\original\kit\random.hpp*/
// random number function declarations

#ifndef RANDOM_HPP
#define RANDOM_HPP

#define GENRAND_SEED_N 624

typedef struct a_seed_info {
    unsigned long   mt[GENRAND_SEED_N];
    int                     mti;
} a_seed_info;

// sgenrand _must_ be called with a seed > 0 before genrand is called
extern void sgenrand( unsigned long seed, a_seed_info *seed_info );

extern unsigned long genrand( a_seed_info *seed_info );

#endif




/*c:\original\kit\stub.cpp*/
```

```
// ***************************************************************************
// Copyright (c) 2001-2014 SAP SE or an SAP affiliate company. All rights reserved.
// ***************************************************************************

// Define stub transaction class, inherited from ITxn.
// Only used for testing.


#include "txn.hpp"

#define   TRUE      1

class Stub: public ITxn {
public:
   void new_order() { };
   void payment() { };
   void order_status() { _params.order_status.o_ol_cnt = 5; };
   void delivery() { };
   void stock_level() { };
   void get_maximum() { _params.max_w_id = 2000; };
};

ITxn *
new_Txn()
/*******/
{
   Stub *  stub = new Stub;
   if( stub != NULL ) {
           memset( &stub->_params, 0, sizeof( stub->_params ) );
   }
   return stub;
}

void free_Txn( ITxn * txn )
/*************************/
{
   Stub *  stub = (Stub *) txn;
   delete stub;
}


a_bool
init_Txn()
/********/
{
   return( TRUE );
}

void
fini_Txn()
/********/
{
}




/*c:\original\kit\tpcc.cpp*/
// ***************************************************************************
// Copyright (c) 2001-2014 SAP SE or an SAP affiliate company. All rights reserved.
// ***************************************************************************

// This module implements a combined SUT/RTE which is easier to run than
// the full ISAPI/COM+/RTE components for testing.
// The results produced by this utility are not valid for disclosure.
```

```
#if defined( UNIX )
   #include "compiler.h"
#else
   #include <crtdbg.h>
   #include <direct.h>
   #include <process.h>
   #include <windows.h>
#endif
#include <math.h>

#include <stdio.h>

#include "random.hpp"
#include "txn.hpp"
#include "utils.hpp"
#include "rteutils.hpp"
#include <signal.h>

#if defined( UNIX )
   #define _Sleep( x ) delay( x )
   #define _stricmp( s1, s2 )          strcasecmp( s1, s2 )
#elif defined( WINNT )
   #define _Sleep( x ) Sleep( x )
   #define _stricmp( s1, s2 )          stricmp( s1, s2 )
#else
   #error Need to define _Sleep for this platform.
#endif

// define this so that each terminal uses its own connection
// (default is each worker thread uses its own connection)
//#define CONN_PER_TERM

int Debugging = 0;
int NumThreads = 1;
int NumDeliveryThreads = 1;
int NWarehouses = 1;
int WarehouseLimit = 0;
int WarehouseFolding = 0;
int ExecuteOnlyTran = -1;
int ExcludeTran = -1;
int ShowStatistics = 0;
int DisplayRate = 0;
a_display_type DisplayType = UNSPECIFIED_DISPLAY_TYPE;
int StopWorkers = FALSE;
a_time InitTime = 0;
int OutputLogs = TRUE;
double ThinkMultiplier = 1.0;
int DumpServerProperties = FALSE;
int DumpDatabaseProperties = FALSE;
int DumpCmdLineOptions = TRUE;
char * RunComment = NULL;
int SavedArgc;
char **SavedArgv;

extern char * DSN;

inline double
Infinity()
/********/
{
#if defined( SUN )
   return (double)HUGE_VAL;
#elif defined( UNIX )
   return (double)HUGE_VALF;
#else
   double d; * (__int64 *) &d = 0x7FF0000000000000i64; return( d );
#endif
```

```
}


#if 1
typedef Timer TimerType;
#else
// Timer object with out real times and no waits.
class StubTimer {
public:
   Timer()
           : _now( 0 )
   {
   }
   a_time query()
   {
           return _now;
   }
   void wait( a_time til )
   {
           _ASSERT( til >= _now );
           _now = til;
   }
   a_time infinity()
   {
           return ~0;
   }
private
   a_time              _now;
};

typedef StubTimer TimerType;
#endif

TimerType Clock;

class CondVar {
public:
   CondVar()
           : _semaphore( 0 )
           , _waiting( 0 )
   {
   }
   void wait( Mutex& mutex )
   {
           _waiting++;
           mutex.give();
           _semaphore.wait();
           mutex.get();
   }
   void signal()
   {
           if( _waiting ) {
              _semaphore.post();
              --_waiting;
           }
   }
private:
   Semaphore                   _semaphore;
   unsigned                    _waiting;
};

// Constraints that need to be verified:
//
// 90% Menu RT < 2s (emulated, so trivially satisfied)
//
// Transaction      Minimum %       Minimum          90th %ile Minimum Mean
// Type             of mix          Keying time      RT constriant     of Think Time
```

```
//
// New-Order         n/a                  18s                  5s                  12s
// Payment           43                   3s                   5s                  12s
// Order-Status      4                    2s                   5s                  10s
// Delivery          4                    2s                   5s                  5s
// Stock-Level       4                    2s                   20s                 5s
//
// Response time for Delivery transaction is terminal response (acknowledging
// that the transaction has been queued), not for the execution of the
// transaction itself.  At least 90% of the transactions must complete within
// 80s of being queued.

typedef unsigned short a_term_id;

class Terminal {
public:
   Terminal( a_term_id id, unsigned long seed )
           : _t_id( id )
           , _txn( NULL )
           , _alarm( 0 )
   {
           sgenrand( seed, &_seed_info );
#ifdef CONN_PER_TERM
           _conn = new_Txn();
#endif
   }
#ifdef CONN_PER_TERM
   ~Terminal() {
           free_Txn( _conn );
   }
#endif

   Terminal * service( class Env   & env,
                                   ITxn            * conn,
                                   TimerType   & clock,
                                   unsigned    thread_id );
   int operator>( const Terminal & t2 ) const
   {
           return _alarm > t2._alarm;
   }
private:
   a_w_id remoteWarehouse( a_w_id local_w_id );

   a_time newOrder   ( Txn * txn, ITxn * conn );
   a_time payment    ( Txn * txn, ITxn * conn );
   a_time orderStatus( Txn * txn, ITxn * conn );
   a_time delivery   ( Txn * txn, ITxn * conn, class Env & env );
   a_time stockLevel ( Txn * txn, ITxn * conn );

   a_term_id                   _t_id;
   Txn *               _txn;
#ifdef CONN_PER_TERM
   ITxn *              _conn;
#endif
   a_seed_info                 _seed_info;
public:
   a_time              _alarm;
};


class DeliveryService {
public:
   DeliveryService( unsigned nDelivery )
           : _queue( new a_delivery_info[nDelivery] )
           , _head( nDelivery )
           , _tail( nDelivery )
           , _remaining_threads( 0 )
```

```
        , _capacity( nDelivery )
    {
        memset( _queue, 0, sizeof( a_delivery_info ) * nDelivery );
    }
    void defer( a_w_id w_id, a_carrier_id o_carrier_id );
    void worker();
    void waitForDeliveries();
private:
    a_delivery_info * _queue;
    unsigned                    _head;
    unsigned                    _tail;
    int                         _remaining_threads;
    unsigned                    _capacity;
    Mutex           _mutex;
    CondVar                     _available;
    Semaphore                   _deliveriesDone;
};

void
DeliveryService::defer( a_w_id w_id, a_carrier_id o_carrier_id )
/*****************************************************************/
{
    _mutex.get();
    if( _head == 0 ) {
        _mutex.give();
        ThrowError( "Out of deliverable objects" );
    }
    a_delivery_info & d = _queue[--_head];
    d.w_id      = w_id;
    d.carrier_id = o_carrier_id;
    d.queued_time = Clock.query();
    _available.signal();
    _mutex.give();
}

void
DeliveryService::worker()
/***********************/
{
    ITxn *  conn = NULL;
    int     i;

    _mutex.get();
    _remaining_threads++;
    _mutex.give();
#ifdef SERVER_COM
    CoInitializeEx( NULL, COINIT_MULTITHREADED );
#endif
    try {
        conn = new_Txn();
        _mutex.get();
        while( _tail > 0 ) {
            while( _head == _tail ) {
                if( _tail == 0 ) goto DONE_WORKER;
                _available.wait( _mutex );
            }
            a_delivery_info * d = &_queue[--_tail];
            _mutex.give();
            if( d->w_id != 0 ) {
                a_delivery & param = conn->_params.delivery;
                param.w_id = d->w_id;
                param.o_carrier_id = d->carrier_id;
                conn->delivery();
                for( i = 0; i < 10; i++ ) {
                    d->o_id[i] = param.o_id[i];
                }
                d->completed_time = Clock.query();
```

```
                }
                _mutex.get();
            }
    } catch ( char * err ) {
            NumErrors++;
            DisplayMutex.get();
            fprintf( stderr, "Delivery Error: %s\n", err );
            DisplayMutex.give();
            // err is always strdup'ed to ensure it is valid even if the object is
            // desctructed
            free( err );
            _mutex.get();
    }
DONE_WORKER:
    if( conn != NULL ) {
            free_Txn( conn );
    }
#ifdef SERVER_COM
    CoUninitialize();
#endif
    if( --_remaining_threads == 0 ) {
            _deliveriesDone.post();
    }
    // ensure all threads stop
    _available.signal();
    _mutex.give();
}

void
DeliveryService::waitForDeliveries()
/*********************************/
{
    unsigned dump_head = 0;
    if( ExecuteOnlyTran != -1 ) return;
    // OK to refer to head outside the mutex since only modified when
    // we call defer
    if( _head > 0 ) {
            // don't dump unused deliveries
            dump_head = _head;
            // not all deliveries where queued.
            // "defer" empty deliveries so that workers finish
            while( _head > 0 ) {
                defer( 0, 0 );
            }
    }
    _mutex.get();
    if( _tail > dump_head ) {
            DisplayMutex.get();
            fprintf( stderr, "%d deliveries remaining\n", _tail - dump_head );
            DisplayMutex.give();
    }
    _mutex.give();
    _deliveriesDone.wait();

    // dump deliveries to .dmp file
    if( OutputLogs ) {
            DumpDeliveryStats( "tpcc_deliveries",
                               &_queue[dump_head],
                               _capacity - dump_head,
                               StartTime,
                               EndTime,
                               NumErrors,
                               NWarehouses );
    }
}

void
```

```
ut_set_debugger_thread_name( char *name )
/*************************************/
// Sets the name of the thread as seen inside the debugger
{
#if defined( UNIX )
    _unused( name );
#else
    struct {
            DWORD   dwType;  // must be 0x1000
            LPCSTR  szName;             // pointer to name (in user addr space)
            DWORD   dwThreadID;         // thread ID (-1=caller thread)
            DWORD   dwFlags;  // reserved for future use, must be zero
    } info;

    info.dwType = 0x1000;
    info.szName = name;
    info.dwThreadID = (DWORD)-1;
    info.dwFlags = 0;

    __try
    {
            RaiseException( 0x406D1388, 0, sizeof(info)/sizeof(DWORD), (ULONG_PTR *)&info );
    }
    __except(EXCEPTION_CONTINUE_EXECUTION)
    {
    }
#endif
}

#if defined( UNIX )
void *
_cdecl DeliveryWorker( void * svc )
#else
void
_cdecl DeliveryWorker( void * svc )
#endif
/*********************************/
{
    ut_set_debugger_thread_name( "delivery" );
    ((DeliveryService *) svc)->worker();
#if defined( UNIX )
    return NULL;
#endif
}

// one global transaction list for all terminals.
// Initialized by Env constructor.
TxnList * Transactions = NULL;

class Env {
public:
    Env( unsigned nTxn, unsigned nTerminal );
    ~Env();
    void deliver( Txn *, a_w_id w_id, a_carrier_id o_carrier_id )
    {
            _delivery.defer( w_id, o_carrier_id );
    }
    void waitForDeliveries()
    {
            _delivery.waitForDeliveries();
    }
    void worker( unsigned thread_id );
    void run();
private:
    Mutex               _mutex;
    DeliveryService     _delivery;
    PriorityQueue<Terminal> _queue;
```

```
    unsigned                    _active_workers;
    CondVar         _inactive;
};

// abort execution cleanly
static void __cdecl stopWorkers( int )
/************************************/
{
    // used by Env::worker
    StopWorkers = TRUE;
}

Env::Env( unsigned nTxn, unsigned nTerminal )
/*******************************************/
    : _delivery( (unsigned)(.06*nTxn) )
    , _queue( nTerminal )
    , _active_workers( 0 )
{
    unsigned            i;
    a_seed_info         seed_info;
    unsigned long   seed;

    // set break handlers
    signal( SIGINT, stopWorkers );
#if !defined( UNIX )
    signal( SIGBREAK, stopWorkers );
#endif

    Transactions = new TxnList( nTxn, DisplayType, DisplayRate );

    for( i = 0; i < (unsigned)NumDeliveryThreads; ++ i ) {
            #if defined( UNIX )
            pthread_t id;
            if( pthread_create(&id, NULL, DeliveryWorker, &_delivery ) != 0 ) {
                printf( "Creation of DeliveryWorker thread %d failed\n", i );
                fflush( stdout );
            }
            #else
            _beginthread( DeliveryWorker, 0, &_delivery );
            #endif
    }

    // Initialize terminals.
    sgenrand( 4513, &seed_info );
    for( i = 0; i < nTerminal; ++i ) {
            seed = genrand( &seed_info );
            if( seed == 0 ) {
                seed = 0xffffffff;
            }
            _queue.insert( new Terminal( i, seed ) );
    }

    // stagger start times of all terminals (do this once all terminals
    // have been created, since creating terminals is slow).
    // 21 seconds is approximately the average transaction time
    a_time start_inc = ToTime( 21.0 ) / nTerminal;
    a_time now = Clock.query();
    Terminal *term;
    for( i = 0; i < nTerminal; i++ ) {
            // terminals are created with _alarm 0, so this gets a
            // terminal which does not have it's service time set yet.
            term = _queue.deleteMin();
            term->_alarm = now + i * start_inc;
            _queue.insert( term );
    }
}
```

```
Env::~Env()
/*********/
{
   delete Transactions;
   Transactions = NULL;
}

// service time of previous terminal
a_time Prev = 0;
// abort timers now
a_bool CalledAbort = FALSE;

void
Env::worker( unsigned thread_id )
/*******************************/
{
   ITxn * conn = NULL;

#ifdef SERVER_COM
   CoInitializeEx( NULL, COINIT_MULTITHREADED );
#endif
   try {
            // need a Timer object per thread which calls Timer::wait since wait is
            // not thread safe
            TimerType           worker_clock;

#ifndef CONN_PER_TERM
   #if      defined( SERVER_MSSQL )
            // serialize connections
            _mutex.get();
            try {
               conn = new_Txn();
            } catch( char * err ) {
               _mutex.give();
               throw( err );
            }
            _mutex.give();
   #else
            conn = new_Txn();
   #endif
#endif
            Terminal * t;
            _mutex.get();
            for( ;; ) {
               t = _queue.deleteMin();
               if( t == NULL ) break;
#ifdef DEBUG
               _ASSERT( t->_alarm >= Prev );
               Prev = t->_alarm;
#endif
               _mutex.give();
               if( StopWorkers ) {
                        // stop all workers and terminals cleanly and quickly
                        if( EndTime == 0 ) {
                           EndTime = worker_clock.query() - ToTime( 0.1 );
                        }
                        if( !CalledAbort ) {
                           CalledAbort = TRUE;
                           Timer::abort();
                           DisplayMutex.get();
                           printf( "Stopping...\n" );
                           DisplayMutex.give();
                        }
                        delete( t );
                        _mutex.get();
                        continue;
               }
```

```
                t = t->service( *this, conn, worker_clock, thread_id );
                _mutex.get();
                if( t == NULL ) {
                        if( EndTime == 0 ) {
                            EndTime = worker_clock.query();
                            StopWorkers = TRUE;
                        }
                        if( thread_id != 1 ) break;
                } else {
                        _queue.insert( t );
#ifdef DEBUG
                        if( t->_alarm < Prev ) {
                            // next request for the terminal just serviced needs to
                            // be serviced before what was going to be the next
                            // serviced request.
                            Prev = t->_alarm;
                        }
#endif
                }
            }
    } catch( char * err ) {
            NumErrors++;
            DisplayMutex.get();
            fprintf( stderr, "[%d] Worker error: %s\n", thread_id, err );
            DisplayMutex.give();
            _mutex.get();
            // err is always strdup'ed to ensure it is valid even if the object is
            // destructed
            free( err );
    }
    if( conn != NULL ) {
            free_Txn( conn );
    }
#ifdef SERVER_COM
    CoUninitialize();
#endif
    if( --_active_workers == 0 ) {
            _inactive.signal();
    }
    _mutex.give();
}

typedef struct a_worker_parm {
    Env *       env;
    unsigned            thread_id;
} a_worker_parm;


#if defined( UNIX )
void *
Worker( void * prm )
#else
void
__cdecl Worker( void * prm )
#endif
/*************************/
{
    char    thread_name[20];
    a_worker_parm *             p = (a_worker_parm *) prm;

    sprintf( thread_name, "worker %d", p->thread_id );
    ut_set_debugger_thread_name( thread_name );
    p->env->worker( p->thread_id );
#if defined( UNIX )
    return NULL;
#endif
}
```

```
static
void dump_server_properties( ITxn * conn, FILE * file )
/*****************************************************/
{
    if( conn != NULL ) {
            fprintf( file, "Server properties:\n" );
            conn->dump_server_properties( file );
            fprintf( file, "\n" );
            fflush( file );
    }
}

static
void dump_database_properties( ITxn * conn, FILE * file )
/*******************************************************/
{
    if( conn != NULL ) {
            fprintf( file, "Server properties:\n" );
            conn->dump_database_properties( file );
            fprintf( file, "\n" );
            fflush( file );
    }
}

static
void dump_cmd_line_options( FILE * file )
/***************************************/
{
    int i;
    fprintf( file, "Command line options: [" );
    for( i = 0; i < SavedArgc; i++ ) {
            fprintf( file, " %s", SavedArgv[i] );
    }
    fprintf( file, " ]\n" );
    if( RunComment != NULL ) {
            fprintf( file, "Comment: [%s]\n", RunComment );
    }
    fflush( file );
}


void
Env::run()
/********/
{
    a_worker_parm *             parms = new a_worker_parm[NumThreads];
    a_worker_parm *             prm;
    char                summary_file_name[100];
    ITxn *              conn;

    InitTime = Clock.query();
    // so the first 2 seconds is ramp up time.
    StartTime = InitTime + ToTime( 2.0 );
    for( int i = 0; i < NumThreads; ++i ) {
            _mutex.get();
            ++_active_workers;
            _mutex.give();
            prm = &parms[i];
            prm->env = this;
            prm->thread_id = i + 1;
#if defined( UNIX )
            pthread_t id;
            if( pthread_create( &id, NULL, Worker, prm ) != 0 ) {
#else
            if( _beginthread( Worker, 0, prm ) == -1 ) {
#endif
```

```
                DisplayMutex.get();
                printf( "Error creating worker thread\n" );
                DisplayMutex.give();
                _mutex.get();
                --_active_workers;
                _mutex.give();
                StopWorkers = TRUE;
                break;
            }
    }
    _mutex.get();
    while( _active_workers != 0 ) {
            _inactive.wait( _mutex );
    }
    _mutex.give();
    if( ThinkMultiplier != 1.0 ) {
            printf( "Non-default think multiplier %f; results file would be incorrect.\n",
                    ThinkMultiplier );
    }

    if( DumpServerProperties || DumpDatabaseProperties ) {
            try {
                conn = new_Txn();
            } catch( char * msg ) {
                printf( "Could not connect to server (%s)! to dump properties. Properties not dumpped!", msg );
                DumpServerProperties = FALSE;
                DumpDatabaseProperties = FALSE;
            }
    }

    if( OutputLogs ) {
#if defined( UNIX )
            mkdir( TPCC_OUTPUT_DIRECTORY, 0766 );
#else
            _mkdir( TPCC_OUTPUT_DIRECTORY );
#endif
            Transactions->dumpTxnStats( "tpcc_txn", 1, NWarehouses );
            SetStatFileName( summary_file_name, "tpcc_summary", StartTime, "txt" );
            FILE * runstatfile = fopen( summary_file_name, "w" );
            if( runstatfile != NULL ) {
                if( DumpCmdLineOptions ) {
                        dump_cmd_line_options( runstatfile );
                }
                Transactions->printRunStats( runstatfile, NWarehouses * 10 );
                if( DumpServerProperties ) {
                        dump_server_properties( conn, runstatfile );
                }
                if( DumpDatabaseProperties ) {
                        dump_database_properties( conn, runstatfile );
                }
                fclose( runstatfile );
            }
    } // don't write output logs
    if( DumpServerProperties ) {
            dump_server_properties( conn, stdout );
    }
    if( DumpServerProperties ) {
            dump_database_properties( conn, stdout );
    }
    if( DumpCmdLineOptions ) {
            dump_cmd_line_options( stdout );
    }
    Transactions->printRunStats( stdout, NWarehouses * 10 );

    if( DumpServerProperties || DumpDatabaseProperties ) {
            free_Txn( conn );
    }
```

```
}

#define N_DPW      10          // Districts per Warehouse

inline a_w_id
TidToWid( a_term_id t_id )
/************************/
{
   if( WarehouseFolding == 0 ) {
           return t_id/N_DPW + 1;
   } else {
           return (t_id/N_DPW)/WarehouseFolding + 1;
   }
}

inline a_d_id
TidToDid( a_term_id t_id )
/************************/
{
   return( (a_d_id)(t_id%N_DPW + 1) );
}

a_w_id
Terminal::remoteWarehouse( a_w_id local_w_id )
/*******************************************/
{
   a_w_id w_id = Random( &_seed_info, 1, WarehouseLimit - 1 );
   if( w_id >= local_w_id ) {
           ++w_id;
   }
   return( w_id );
}

a_time
Terminal::newOrder( Txn * txn, ITxn * conn )
/******************************************/
{
   a_new_order & param = conn->_params.new_order;
   param.w_id = TidToWid( _t_id );
   param.d_id = Random( &_seed_info, 1, 10 );
   param.c_id = NURand( &_seed_info, 1023, 1, 3000, NU_C_C_ID );
   param.o_ol_cnt = Random( &_seed_info, 5, 15 );
   txn->new_order_lines = param.o_ol_cnt;
   for( unsigned i = 0; i < param.o_ol_cnt; ++i ) {
           param.ol[i].ol_i_id = NURand( &_seed_info, 8191, 1, 100000, NU_C_OL_ID );
           if( WarehouseLimit == 1 || Random( &_seed_info, 1, 100 ) > 1 ) {
              param.ol[i].ol_supply_w_id = param.w_id;
           } else {
              param.ol[i].ol_supply_w_id = remoteWarehouse( param.w_id );
              txn->new_order_remote_lines++;
           }
           param.ol[i].ol_quantity = Random( &_seed_info, 1, 10 );
   }
   if( Random( &_seed_info, 1, 100 ) == 1 ) {
           param.ol[param.o_ol_cnt-1].ol_i_id = ~0; // Invalid item to force rollback.
           txn->new_order_rollback = TRUE;
   }

   a_time sut_start = Clock.query();
   conn->new_order();
   Transactions->incCompleteNewOrder();
   return sut_start;
}

a_time
Terminal::payment( Txn * txn, ITxn * conn )
/*****************************************/
```

```
{
   a_payment & param = conn->_params.payment;
   param.w_id = TidToWid( _t_id );
   param.d_id = Random( &_seed_info, 1, 10 );
   if( WarehouseLimit == 1 || Random( &_seed_info, 1, 100 ) <= 85 ) {
            param.c_w_id = param.w_id;
            param.c_d_id = param.d_id;
   } else {
            txn->payment_remote = TRUE;
            param.c_w_id = remoteWarehouse( param.w_id );
            param.c_d_id = Random( &_seed_info, 1, 10 );
   }
   param.c_id = RandUser( &_seed_info, param.c_last );
   if( param.c_id == 0 ) {
            txn->payment_c_last = TRUE;
   }
   param.h_amount = Random( &_seed_info, 100, 500000 )/100.0;

   a_time sut_start = Clock.query();
   conn->payment();
   return sut_start;
}

a_time
Terminal::orderStatus( Txn * txn, ITxn * conn )
/*********************************************/
{
   an_order_status & param = conn->_params.order_status;
   param.w_id = TidToWid( _t_id );
   param.c_id = RandUser( &_seed_info, param.c_last );
   if( param.c_id == 0 ) {
            txn->order_status_c_last = TRUE;
   }
   param.d_id = Random( &_seed_info, 1, 10 );

   a_time sut_start = Clock.query();
   conn->order_status();
   return sut_start;
}

a_time
Terminal::delivery( Txn * txn, ITxn * conn, Env & env )
/*****************************************************/
{
   a_delivery & param = conn->_params.delivery;
   param.w_id = TidToWid( _t_id );
   param.o_carrier_id = Random( &_seed_info, 1, 10 );

   a_time sut_start = Clock.query();
   env.deliver( txn, param.w_id, param.o_carrier_id );
   return sut_start;
}

a_time
Terminal::stockLevel( Txn *, ITxn * conn )
/****************************************/
{
   a_stock_level & param = conn->_params.stock_level;
   param.w_id = TidToWid( _t_id );
   param.d_id = TidToDid( _t_id );
   param.threshold = Random( &_seed_info, 10, 20 );

   a_time sut_start = Clock.query();
   conn->stock_level();
   return sut_start;
}
```

```
#define   RESPONSE_TIME_DELAY       0.1

// 1. Select transaction type    (1 - 4 == think time)
// 2. Display screen             (2 - 1 == menu time)
// 3. Submit request             (3 - 2 == keying time)
// 4. Display data               (4 - 3 == response time)

Terminal *
Terminal::service( class Env    & env,
                   ITxn         * conn,
                   TimerType    & worker_clock,
                   unsigned       )
/*****************************************/
{
#ifdef CONN_PER_TERM
   conn = _conn;
#endif
   if( _txn == NULL ) {
          _txn = Transactions->getNewTxn( &_seed_info );
          if( _txn == NULL ) return( NULL );
          worker_clock.wait( _alarm );
          _alarm = worker_clock.query();
          _txn->keying_time = ToTime( TxnKeyingTime[_txn->type] );
   } else {
          // use worker_clock instead of global Clock since wait is not thread
          // safe.
          worker_clock.wait( _alarm );
          if( StopWorkers ) {
             // don't include transactions after we are stopping since
             // timer waits are aborted and not accurate
             return( NULL );
          }
   }
   a_time now = worker_clock.query();

   a_time sut_start = _alarm;
   if( ( ExecuteOnlyTran == -1 || ExecuteOnlyTran == _txn->type )
     &&( ExcludeTran != _txn->type ) ) {
          switch( _txn->type ) {
          case NEW_ORDER:    sut_start = newOrder   ( _txn, conn ); break;
          case PAYMENT:      sut_start = payment    ( _txn, conn ); break;
          case ORDER_STATUS: sut_start = orderStatus( _txn, conn ); break;
          case DELIVERY:     sut_start = delivery   ( _txn, conn, env ); break;
          case STOCK_LEVEL:  sut_start = stockLevel ( _txn, conn ); break;
          }
   }
   _txn->txn_complete_time = worker_clock.query() + ToTime( RESPONSE_TIME_DELAY );
   _txn->state = TXN_COMPLETED;
   _txn->successful = TRUE;
   _txn->term_w_id = TidToWid( _t_id );
   _txn->term_d_id = TidToDid( _t_id );
   _txn->keying_time += sut_start - _alarm;        // add late service time to keying time
   _txn->txn_start_time = sut_start;
   _txn->response_time = _txn->txn_complete_time - sut_start;
   _txn->think_time = ToTime( NegExp( &_seed_info,
                                      TxnMeanThinkTime[ _txn->type ] * ThinkMultiplier ) );
   _alarm = _txn->txn_complete_time + _txn->think_time;

   // Begin new transaction
   _txn = Transactions->getNewTxn( &_seed_info );
   if( _txn == NULL ) return( NULL );
   _txn->keying_time = ToTime( TxnKeyingTime[_txn->type] * ThinkMultiplier );
   _txn->menu_response_time = ToTime( RESPONSE_TIME_DELAY );
   _alarm += _txn->keying_time + ToTime( RESPONSE_TIME_DELAY );
   return this;
}
```

```
static int StrToTran( char *str )
/*******************************/
{
    int tran = -1;

    if( _stricmp( str, "new_order" ) == 0 ) {
            tran = NEW_ORDER;
    } else if( _stricmp( str, "payment" ) == 0 ) {
            tran = PAYMENT;
    } else if( _stricmp( str, "order_status" ) == 0 ) {
            tran = ORDER_STATUS;
    } else if( _stricmp( str, "delivery" ) == 0 ) {
            tran = DELIVERY;
    } else if( _stricmp( str, "stock_level" ) == 0 ) {
            tran = STOCK_LEVEL;
    }
    return( tran );
}

int
main( int argc, char *argv[] )
/*****************************/
{
    unsigned nTxn = 1000;
    SavedArgc = argc;
    SavedArgv = argv;
    while( --argc > 0 ) {
            char * arg = *++argv;
            if( strcmp( arg, "-thinkmul" ) == 0 && argc > 1 ) {
                ThinkMultiplier = strtod( *++argv, NULL ); --argc;
            } else if( strcmp( arg, "-dsn" ) == 0 && argc > 1 ) {
                DSN = strdup( *++argv ); --argc;
            } else if( strcmp( arg, "-srv_prop" ) == 0 ) {
                DumpServerProperties = TRUE;
            } else if( strcmp( arg, "-db_prop" ) == 0 ) {
                DumpDatabaseProperties = TRUE;
            } else if( strcmp( arg, "-dump_args" ) == 0 ) {
                DumpCmdLineOptions = TRUE;
            } else if( strcmp( arg, "-comment" ) == 0 ) {
                ++argv;
                --argc;
                RunComment = *argv;
            } else if( arg[0] == '-' && arg[1] == 'd' ) {
                Debugging = 1;
            } else if( arg[0] == '-' && arg[1] == 'e' && argc > 1 ) {
                ++argv;
                --argc;
                ExcludeTran = StrToTran( *argv );
                if( ExcludeTran < 0 ) {
                        fprintf( stderr, "Invalid transaction name\n" );
                        exit( 1 );
                }
            } else if( arg[0] == '-' && arg[1] == 'n' && argc > 1 ) {
                nTxn = atoi( *++argv ); --argc;
            } else if( arg[0] == '-' && arg[1] == 'o' ) {
                OutputLogs = FALSE;
            } else if( arg[0] == '-' && arg[1] == 'r' && argc > 1 ) {
                DisplayRate = atoi( *++argv ); --argc;
            } else if( arg[0] == '-' && arg[1] == 's' ) {
                ShowStatistics = 1;
            } else if( arg[0] == '-' && arg[1] == 't' && argc > 1 ) {
                NumThreads  = atoi( *++argv ); --argc;
            } else if( arg[0] == '-' && arg[1] == 'v' && argc > 1 ) {
                NumDeliveryThreads  = atoi( *++argv ); --argc;
            } else if( arg[0] == '-' && arg[1] == 'w' && argc > 1 ) {
                if( arg[2] == 'l' ) {
                        WarehouseLimit = atoi( *++argv );
```

```
                --argc;
            } else {
                NWarehouses  = atoi( *++argv ); --argc;
            }
        } else if( arg[0] == '-' && arg[1] == 'x' && argc > 1 ) {
            ++argv;
            --argc;
            ExecuteOnlyTran = StrToTran( *argv );
            if( ExecuteOnlyTran < 0 ) {
                fprintf( stderr, "Invalid transaction name\n" );
                exit( 1 );
            }
        } else if( arg[0] == '-' && arg[1] == 'z' && argc > 1 ) {
            ++argv;
            --argc;
            if( _stricmp( *argv, "dot" ) == 0 ) {
                DisplayType = DOT_PER_TRANS;
            } else if( _stricmp( *argv, "thread" ) == 0 ) {
                DisplayType = SHOW_EXECUTING_THREAD;
            } else if( _stricmp( *argv, "count" ) == 0 ) {
                DisplayType = COUNTS_PER_TXN_TYPE;
            } else if( _stricmp( *argv, "detail" ) == 0 ) {
                DisplayType = DISPLAY_DETAIL;
            } else if( _stricmp( *argv, "none" ) == 0 ) {
                DisplayType = DISPLAY_NONE;
            } else {
                fprintf( stderr, "Invalid display type\n" );
                exit( 1 );
            }
        } else {
            fprintf( stderr, "Options:\n" );
            fprintf( stderr, "    -thinkmul <num>   (think time multiplier)\n" );
            fprintf( stderr, "    -dsn <name>       (ODBC dsn name (default:tpcc))\n" ) ;
            fprintf( stderr, "    -srv_prop         (dump server properties at end of run)\n" );
            fprintf( stderr, "    -db_prop          (dump database properties at end of run)\n" );
            fprintf( stderr, "    -dump_args        (dump command line arguments to output at end of run)\n" );
            fprintf( stderr, "    -comment <string> (optional comment to document the test run)\n" );
            fprintf( stderr, "    -d                (debugging)\n" );
            fprintf( stderr, "    -e  <trans_name>  (exclude transaction)\n" );
            fprintf( stderr, "    -n  <txns>\n" );
            fprintf( stderr, "    -o                (do not write output logs)\n" );
            fprintf( stderr, "    -r                <display rate>\n" );
            fprintf( stderr, "    -s                (show terminal counts)\n" );
            fprintf( stderr, "    -t  <num>         <number of threads)\n" );
            fprintf( stderr, "    -v  <num>         (number of deliver threads)\n" );
            fprintf( stderr, "    -x  <trans_name>  (execute only transaction name specified)\n" );
            fprintf( stderr, "    -w  <warehouses>\n" );
            fprintf( stderr, "    -wl <warehouse limit>\n" );
            fprintf( stderr, "    -z  <display_type>\n" );
            fprintf( stderr, "\n" );
            fprintf( stderr, "  trans_name:   [new_order,payment,order_status,delivery,stock_level]\n" );
            fprintf( stderr, "  display_type: [dot, thread, count, detail, none]\n" );
            exit( 1 );
        }
    }
    if( WarehouseLimit != 0 ) {
        WarehouseFolding = NWarehouses / WarehouseLimit;
        if( (WarehouseLimit * WarehouseFolding) != NWarehouses ) {
            fprintf( stderr, "Invalid warehouse limit\n" );
            exit( 1 );
        }
    } else {
        WarehouseLimit = NWarehouses;
    }
    unsigned nTerminal = NWarehouses*N_DPW;
    if( DisplayType == UNSPECIFIED_DISPLAY_TYPE ) {
        if( NumThreads == 1 ) {
```

```
            DisplayType = DOT_PER_TRANS;
        } else {
            DisplayType = COUNTS_PER_TXN_TYPE;
        }
    }
    if( DisplayRate == 0 ) {
        if( NWarehouses < 100 ) {
            DisplayRate = 100;
        } else {
            DisplayRate = 1000;
        }
    }

    if( init_Txn() ) {
        try {
            // Initialize txns.
            if( nTxn < nTerminal ) nTxn = nTerminal * 4;
            printf( "Terminals: %d, Threads: %d, Transactions: %d\n",
                    nTerminal, NumThreads, nTxn );
            Env env( nTxn, nTerminal );

            // Run test.
            env.run();

            if( Debugging ) {
                DisplayMutex.get();
                fprintf( stderr, "waiting for deliveries to complete\n" );
                DisplayMutex.give();
            }
            env.waitForDeliveries();
        } catch( char *msg ) {
            printf( "main error: %s\n", msg );
            free( msg );
        }

        fini_Txn();
    }

    return( NumErrors );
}



/*c:\original\kit\tpcccom.cpp*/
// ****************************************************************************
// Copyright (c) 2001-2014 SAP SE or an SAP affiliate company. All rights reserved.
// ****************************************************************************

// Implement COM+ component ITPCCTran interface.
// comclient.cpp is the client which uses the ITPCCTran interface.

// tpcccom.cpp : Implementation of DLL Exports and ITPCCTran interface
// using ATL (Microsoft Active Template Library).

#define _ATL_APARTMENT_THREADED

#include <atlbase.h>
extern CComModule _Module;
#include <atlcom.h>
#include <initguid.h>
#include <mtx.h>
#include <stdio.h>
#include "tpcccom.h"
#include "tpcccom_i.c"
#include "txn.hpp"
#include "utils.hpp"
```

```
// write errors to this file
#define ERROR_FILE                      "com_error.txt"

// must match definition in tpcccom.rc
#define IDR_TPCCTRAN                    101

CComModule _Module;

// Windows 2000 function not available in VisualStudio 6 header files
// or libraries.
typedef HRESULT (STDAPICALLTYPE * T_CoGetObjectContext)
                                (IN REFIID riid, OUT LPVOID FAR* ppv);
static T_CoGetObjectContext P_CoGetObjectContext;


/////////////////////////////////////////////////////////////////////////////
// CTPCCTran
class ATL_NO_VTABLE CTPCCTran :
        public CComObjectRootEx<CComMultiThreadModel>,
        public CComCoClass<CTPCCTran, &CLSID_TPCCTran>,
        public IObjectControl,
        public ITPCCTran
{
public:
        CTPCCTran();
        ~CTPCCTran();

DECLARE_REGISTRY_RESOURCEID(IDR_TPCCTRAN)

DECLARE_PROTECT_FINAL_CONSTRUCT()

BEGIN_COM_MAP(CTPCCTran)
        COM_INTERFACE_ENTRY(ITPCCTran)
        COM_INTERFACE_ENTRY(IObjectControl)
END_COM_MAP()

// IObjectControl
public:
    STDMETHODIMP_( BOOL) CanBePooled() { return( _conn != NULL ); }
    STDMETHODIMP Activate() { return S_OK; }
    STDMETHODIMP_( void ) Deactivate() { /* nothing to do */ }

// ITPCCTran
public:
    STDMETHOD(initialization_complete)();

    STDMETHOD(new_order)( int data_len, unsigned char *param );
    STDMETHOD(payment)( int data_len, unsigned char *param );
    STDMETHOD(order_status)( int data_len, unsigned char *param );
    STDMETHOD(delivery)( int data_len, unsigned char *param );
    STDMETHOD(stock_level)( int data_len, unsigned char *param );
    STDMETHOD(get_maximum)( int data_len, unsigned char *param );

private:
    void set_complete( void );

    // common transaction processing upto calling the _conn method
    inline HRESULT pre_tran( char               * name,
                             int                  expected_len,
                             int                  data_len,
                             unsigned char   * param );
    // handle caught exception msg
    inline HRESULT transaction_exception( char * name, char * msg );

private:
    ITxn *  _conn;
```

```
};


// object contructor
CTPCCTran::CTPCCTran() : _conn( NULL ) //, _last_error_text( NULL )
/***********************************/
{
   if( P_CoGetObjectContext == NULL ) {
          WriteError( "CTPCCTran Constructor Error",
                        "Could not find CoGetObjectContext symbol" );
          return;
   }
   try {
          _conn = new_Txn();
   } catch( char * msg ) {
          WriteError( "CTPCCTran Constructor Error",
                        msg );
          free( msg );
   }
}

// object desctructor
CTPCCTran::~CTPCCTran()
/**********************/
{
   if( _conn != NULL ) {
          free_Txn( _conn );
   }
}

// helper for all transactions: check parameters and initialize _conn's _param
HRESULT CTPCCTran::pre_tran( char         * tran_name,
                             int          expected_len,
                             int          data_len,
                             unsigned char  * param )
/*****************************************************/
{
   if( data_len != expected_len ) {
          WriteError( tran_name,
                        "invalid parameter length" );
          return( E_INVALIDARG );
   }
   memcpy( &_conn->_params, param, data_len );
   return( S_OK );
}

// helper for all transactions: handle the caught exception, msg
HRESULT CTPCCTran::transaction_exception( char * tran_name, char * msg )
/************************************************************************/
{
   WriteError( tran_name, msg );
   // free _conn and set to NULL so we stop pooling this connection
   free_Txn( _conn );
   _conn = NULL;
   return( E_FAIL );
}

// new_order transaction
STDMETHODIMP CTPCCTran::new_order( int data_len, unsigned char *param )
/**********************************************************************/
{
   HRESULT              ret = S_OK;

   ret = pre_tran( "new_order", sizeof( a_new_order ), data_len, param );
   if( ret == S_OK ) {
          try {
              _conn->new_order();
```

```
            memcpy( param, &_conn->_params, data_len );
        } catch( char * msg ) {
            ret = transaction_exception( "new_order", msg );
        }
    }
    set_complete();
    return( ret );
}

// payment transaction
STDMETHODIMP CTPCCTran::payment( int data_len, unsigned char *param )
/*********************************************************************/
{
    HRESULT             ret = S_OK;

    ret = pre_tran( "payment", sizeof( a_payment ), data_len, param );
    if( ret == S_OK ) {
        try {
            _conn->payment();
            memcpy( param, &_conn->_params, data_len );
        } catch( char * msg ) {
            ret = transaction_exception( "payment", msg );
        }
    }
    set_complete();
    return( ret );
}

// order_status transaction
STDMETHODIMP CTPCCTran::order_status( int data_len, unsigned char *param )
/*************************************************************************/
{
    HRESULT             ret = S_OK;

    ret = pre_tran( "order_status", sizeof( an_order_status ), data_len, param );
    if( ret == S_OK ) {
        try {
            _conn->order_status();
            memcpy( param, &_conn->_params, data_len );
        } catch( char * msg ) {
            ret = transaction_exception( "order_status", msg );
        }
    }
    set_complete();
    return( ret );
}

// delivery transaction
STDMETHODIMP CTPCCTran::delivery( int data_len, unsigned char *param )
/*********************************************************************/
{
    HRESULT             ret = S_OK;

    ret = pre_tran( "delivery", sizeof( a_delivery ), data_len, param );
    if( ret == S_OK ) {
        try {
            _conn->delivery();
            memcpy( param, &_conn->_params, data_len );
        } catch( char * msg ) {
            ret = transaction_exception( "delivery", msg );
        }
    }
    set_complete();
    return( ret );
}

// stock_level transaction
```

```
STDMETHODIMP CTPCCTran::stock_level( int data_len, unsigned char *param )
/************************************************************************/
{
   HRESULT              ret = S_OK;

   ret = pre_tran( "stock_level", sizeof( a_stock_level ), data_len, param );
   if( ret == S_OK ) {
           try {
              _conn->stock_level();
              memcpy( param, &_conn->_params, data_len );
           } catch( char * msg ) {
              ret = transaction_exception( "stock_level", msg );
           }
   }
   set_complete();
   return( ret );
}

// get_maximum
STDMETHODIMP CTPCCTran::get_maximum( int data_len, unsigned char *param )
/************************************************************************/
{
   HRESULT              ret = S_OK;

   ret = pre_tran( "get_maximum",
                           sizeof( _conn->_params.max_w_id ),
                           data_len,
                           param );
   if( ret == S_OK ) {
           try {
              _conn->get_maximum();
              memcpy( param, &_conn->_params, data_len );
           } catch( char * msg ) {
              ret = transaction_exception( "get_maximum", msg );
           }
   }
   set_complete();
   return( ret );
}

// releases the object back to the object pool
void CTPCCTran::set_complete()
/****************************/
{
   char err_buf[100];

   // this code is necessary for object pooling, but causes a new
   // connection per request when not using object pooling.
   IObjectContext* pObjectContext = NULL;

   // get our object context
   HRESULT hr = P_CoGetObjectContext( IID_IObjectContext,
                                                   (void **)& pObjectContext );
   if( pObjectContext == NULL ) {
           sprintf( err_buf, "GetObjectContext failed with 0x%x", hr );
           WriteError( "set_complete Error", err_buf );
   } else {
           pObjectContext->SetComplete();
           pObjectContext->Release();
   }
}

// This method must be called after the client creates a ITPCCTran object
// for connection pooling to work correctly.
// Returns E_FAIL if object construction failed.
STDMETHODIMP CTPCCTran::initialization_complete()
/***********************************************/
```

```
{
   HRESULT hr;

   if( _conn == NULL ) {
           hr = E_FAIL;
   } else {
           hr = S_OK;
   }
   set_complete();
   return( hr );
}


BEGIN_OBJECT_MAP(ObjectMap)
OBJECT_ENTRY(CLSID_TPCCTran, CTPCCTran)
END_OBJECT_MAP()

////////////////////////////////////////////////////////////////////////////
// DLL Entry Point

extern "C"
BOOL WINAPI DllMain(HINSTANCE hInstance, DWORD dwReason, LPVOID /*lpReserved*/)
{
   if( dwReason == DLL_PROCESS_ATTACH ) {
           SetErrorFileName( ERROR_FILE );
           WriteError( "DllMain", "TPCC COM Component Loaded" );
           // Get Windows 2000 function pointer not in
           // Visual Studio 6 header files or libaries.
           HMODULE hDLL = GetModuleHandle( "OLE32.DLL" );
           if( hDLL != NULL ) {
               P_CoGetObjectContext = (T_CoGetObjectContext)
                       GetProcAddress( hDLL, "CoGetObjectContext" );
           }
           // Initialize COM component
      _Module.Init( ObjectMap, hInstance, &LIBID_TPCCCOMLib );
      DisableThreadLibraryCalls( hInstance );
   } else if( dwReason == DLL_PROCESS_DETACH ) {
      _Module.Term();
           WriteError( "DllMain", "TPCC COM Component Unloaded" );
   }
   return TRUE;
}

////////////////////////////////////////////////////////////////////////////
// Used to determine whether the DLL can be unloaded by OLE

STDAPI DllCanUnloadNow(void)
{
   return (_Module.GetLockCount()==0) ? S_OK : S_FALSE;
}

////////////////////////////////////////////////////////////////////////////
// Returns a class factory to create an object of the requested type

STDAPI DllGetClassObject(REFCLSID rclsid, REFIID riid, LPVOID* ppv)
{
   return _Module.GetClassObject(rclsid, riid, ppv);
}

////////////////////////////////////////////////////////////////////////////
// DllRegisterServer - Adds entries to the system registry

STDAPI DllRegisterServer(void)
{
   // registers object, typelib and all interfaces in typelib
   return _Module.RegisterServer(TRUE);
}
```

```
////////////////////////////////////////////////////////////////////////////
// DllUnregisterServer - Removes entries from the system registry

STDAPI DllUnregisterServer(void)
{
   return _Module.UnregisterServer(TRUE);
}




/*c:\original\kit\tpccisapi.cpp*/
// ********************************************************************************
// Copyright (c) 2001-2014 SAP SE or an SAP affiliate company. All rights reserved.
// ********************************************************************************

// This module implements the ISAPI TPC-C web application


// REQUIRED SETUP:
// create tpcc IIS virtual directory in the Default Web Site folder
// - configure it so this DLL is run for * extensions
// create TPCC_OUTPUT_DIRECTORY (see utils.hpp)
// - give IUSR_<MACHINE_NAME> and IWAM_<MACHINE_NAME> accounts full control
//   this this directory
// may want to set HKEY_LOCAL_MACHINE\Software\SAP\TPCC registry entries
// - (see setParameters function)

#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <windows.h>
#include <httpext.h>
#include <limits.h>
#include <float.h>
#include <process.h>
#include "txn.hpp"
#include "utils.hpp"

// write errors to this file
#define ERROR_FILE                      "isapi_error.txt"
// deferred delivery result file
#define DELIVERY_RESULT_FILE        TPCC_OUTPUT_DIRECTORY "\\delivery_results.txt"

// Define USE_WORKER_THREADS for worker threads, don't define it for no
// worker threads.  Note the deferred execution delivery threads are different
// from worker threads (worker threads are used for all requests which interact
// with the database other than the delivery transaction).

#ifdef SERVER_COM
   // When built for COM+, worker threads are required to avoid
   // RPC_E_WRONG_THREAD errors
   #define USE_WORKER_THREADS 1
#endif

#ifdef DEBUG
   // uncomment to dump every request begin & end
   //#define DUMP_REQ_IN_OUT 1
   // uncomment to dump delivery begin & end on the delivery thread
   //#define DUMP_DELIVERY_IN_OUT 1
#endif

#define HTML_PRE_BODY "<HTML><HEAD><TITLE>"                          \
```

```
                        "iAnywhere Solutions TPC-C Application"         \
                        "</TITLE></HEAD>\r\n"                           \
                        "<BODY>"

#define HTML_POST_BODY "</BODY></HTML>\r\n\r\n"

// multiple of 10 spaces
#define SP10 "          "
#define SP20 SP10 SP10
#define SP30 SP20 SP10
#define SP40 SP30 SP10

#define NO_INT_VALUE        INT_MIN

// Maximum parameter values
#define MAX_D_ID      10          // district ID
#define MAX_C_ID      3000    // customer ID
#define MAX_S_I_ID              999999  // stock item ID (above max id for rollback)
#define MAX_ORDER_QTY         50          // order item quantity
#define MAX_PAY_AMOUNT      9999.991// payment amount
#define MAX_CARRIER_ID        10          // delivery carrier ID
#define MAX_SL_THRESHOLD    99          // stock level threshold

// Maximum warehouse ID.  Set on first valid connection.
static int maxW_ID = 0;


typedef struct {
    a_bool active;              // request currently being processed
    int             w_id;               // warehouse id for terminal
    int             d_id;               // district id for stock-level for terminal
    DWORD           last_tick_count;// time of last request in tick counts
    ITxn * conn;                // transaction connection object
} a_terminal;

// maximum number of concurrent terminals
static int maxTerminals;
// an array of terminals (size maxTerminals)
static a_terminal *terminal;

// used when adding and deleting terminals.
static Mutex termAddDeleteMutex;

// Next availible terminal in terminals array.
// When this reaches maxTerminals, the terminal which has been deleted or
// the least recently accessed is reused.
static int nextAvailTerm = 0;

// types of web requests
typedef enum {
    REQ_NEW_ORDER_SUBMIT,
    REQ_PAYMENT_SUBMIT,
    REQ_INPUT_FORM,
    REQ_ORDER_STATUS_SUBMIT,
    REQ_DELIVERY_SUBMIT,
    REQ_STOCK_LEVEL_SUBMIT,
    REQ_WELCOME,
    REQ_MENU,
    NUM_REQ_TYPES
} a_req_type;

static char *requestNames[NUM_REQ_TYPES] = {
    "Process+New-Order",
    "Process+Payment",
    "Input",
    "Process+Order-Status",
    "Process+Delivery",
```

```
    "Process+Stock-Level",
    "_Welcome",
    "Menu",
};

static int sumRequests[NUM_REQ_TYPES] = { 0, 0, 0, 0, 0, 0, 0, 0 };

// Number of errors during run
static int numErrors = 0;

// an array of deliveries allocated when this DLL is loaded
static a_delivery_info *deliveries;

// number of elements allocated in deliveries array
static int maxDeliveries = 250000;

// protect nextDelivery
static Mutex nextDeliveryMutex;

// next unused element in deliveries array.  Guarded by nextDeliveryMutex
static int nextDelivery = 0;

static int numDeliveryThreads;
static long numDeliveryThreadsRunning = 0;
static HANDLE deliveryCompletionPort;

static int numWorkerThreads;
#ifdef USE_WORKER_THREADS
    static long numWorkerThreadsRunning = 0;
    static HANDLE workerCompletionPort;
    #define USE_WORKER_THREADS_STR "Yes"
#else
    #define USE_WORKER_THREADS_STR "No"
#endif

// start and stop worker and delivery threads
static void stopThreads( void );
static void startThreads( void );

// write Delivery deferred execution results to result file
static void writeDeliveryResults( void );



// take left substring of full_str upto length
// characters (length does not include null char), and copy into fill_buffer
static char *leftStr( char *fill_buffer, char *full_str, int length )
/*********************************************************************/
{
    strncpy( fill_buffer, full_str, length );
    fill_buffer[ length ] = '\0';
    return( fill_buffer );
}

// return the value data for the registry entry
// HKEY_LOCAL_MACHINE\Software\SAP\TPCC\<value_name>
// Returns default_value if an error occurs or the registry entry doesn't exist
static int readRegistryDWord( char *value_name, int default_value )
/*********************************************************************/
{
    HKEY            hkey;
    LONG            rc;
    DWORD                   type;
    DWORD                   value_data;
    DWORD                   data_len = sizeof( value_data );

    rc = RegOpenKeyEx( HKEY_LOCAL_MACHINE,
```

```
                        "Software\\SAP\\TPCC",
                        0L,
                        KEY_READ, &hkey );
   if( rc == ERROR_SUCCESS ) {
        rc = RegQueryValueEx( hkey,
                              value_name,
                              NULL,
                              &type,
                              (LPBYTE)&value_data,
                              &data_len );
        RegCloseKey( hkey );
        if( rc == ERROR_SUCCESS && type == REG_DWORD ) {
           return( value_data );
        }
   }
   return( default_value );
}

// get one parameter value based on registry setting or default.  Ensure within
// min_value & max_value
static int getParamValue( char   *value_name,
                          int    default_value,
                          int    min_value,
                          int    max_value )
/*********************************************/
{
   int value = readRegistryDWord( value_name, default_value );

   return( max( min( value, max_value ), min_value ) );
}

// set parameters based on registry settings or defaults,
// and ensure settings are reasonable
static void setParameters( void )
/*******************************/
{
   numWorkerThreads = getParamValue( "ISAPIWorkerThreads", 100, 5, 1000 );
   maxDeliveries = getParamValue( "ISAPIMaxDeliveries", 1000000, 50000, 5000000 );
   numDeliveryThreads = getParamValue( "ISAPIDeliveryThreads", 50, 1, 200 );
   maxTerminals = getParamValue( "ISAPIMaxTerminals", 20000, 1000, 500000 );
}

BOOL WINAPI DllMain( HINSTANCE       hinstDll,
                     DWORD           dwReason,
                     LPVOID          lpvContext )
/***************************************/
{
   int i;
   char buffer[500];
   BOOL ret = TRUE;

   switch( dwReason ) {
   case DLL_PROCESS_ATTACH:
        // DLL Initialization
        try {
           SetErrorFileName( ERROR_FILE );
           // set parameters based on registry settings or defaults
           setParameters();
           sprintf( buffer, "Starting TPCC Application\n"
                            "\tPID: %d, hinstDLL: 0x%Ix\n"
                            "\tUsing Worker Threads: %s, # Worker Threads: %d\n"
                            "\tMax Deliveries: %d, # Delivery Threads: %d\n"
                            "\tMax Terminals: %d\n",
                            GetCurrentProcessId(),
                            hinstDll,
                            USE_WORKER_THREADS_STR,
                            numWorkerThreads,
```

```
                                maxDeliveries,
                                numDeliveryThreads,
                                maxTerminals );
            WriteError( "DllMain", buffer );
            terminal = (a_terminal *)malloc( sizeof( a_terminal )
                                                    * maxTerminals );
            if( terminal == NULL ) {
                    ret = FALSE;
                    WriteError( "DllMain", "out of memory" );
            }
            if( ret ) {
                    memset( terminal, 0, sizeof( a_terminal ) * maxTerminals );
                    ret = init_Txn();
            }
            if( ret ) {
                    deliveries = (a_delivery_info *)malloc( sizeof( a_delivery_info )
                                                                    * maxDeliveries );
                    if( deliveries == NULL ) {
                       WriteError( "DllMain", "out of memory" );
                       ret = FALSE;
                    } else {
                       memset( deliveries,
                                    0,
                                    sizeof( a_delivery_info ) * maxDeliveries );
                       startThreads();
                    }
            }
         } catch( char *msg ) {
            WriteError( "DLLMain Initialize Error", msg );
         }
      break;

   case DLL_PROCESS_DETACH:
         // see TerminateExtension
         sprintf( buffer, "Stopping TPCC Application\n"
                          "\tPID: %d, hinstDLL: 0x%Ix\n"
                          "\tNum Terminals: %d, Num Deliveries: %d\n",
                          GetCurrentProcessId(),
                          hinstDll,
                          nextAvailTerm,
                          nextDelivery );
         WriteError( "DllMain", buffer );
      break;
   }

   return( ret );
}

// called once after the DLL is loaded
BOOL WINAPI GetExtensionVersion( HSE_VERSION_INFO *pVer )
/*******************************************************/
{
   pVer->dwExtensionVersion = MAKELONG( HSE_VERSION_MINOR,
                           HSE_VERSION_MAJOR );

   strcpy( pVer->lpszExtensionDesc,
         "iAnywhere Solutions TPC-C ISAPI Extension" );

   // allow threads to start
   Sleep( 100 );

   return( TRUE );
}

// delete the given terminal
static void deleteTerm( int term_id )
/***********************************/
```

```
{
   if( terminal[term_id].conn != NULL ) {
            free_Txn( terminal[term_id].conn );
            terminal[term_id].conn = NULL;
   }
   terminal[term_id].active = FALSE;
   terminal[term_id].last_tick_count = 0;
}

// called just before DLL is unloaded
BOOL WINAPI TerminateExtension( DWORD dwFlags )
/*********************************************/
{
   char buffer[1024];

   if( terminal == NULL || deliveries == NULL ) {
            WriteError( "Terminate Error", "TerminateExtension called multiple times\n" );
            return TRUE;
   }

   try {
            // DLL is about to be unloaded, do finalization
#if !defined( USE_WORKER_THREADS )
            for( int i = 0; i < nextAvailTerm; i++ ) {
               deleteTerm( i );
            }
#endif
            stopThreads();
            fini_Txn();
            writeDeliveryResults();
            free( terminal );
            terminal = NULL;
            free( deliveries );
            deliveries = NULL;
            sprintf( buffer, "Done TPCC Application\n"
                                    "\tNum Terminals: %d, Num Deliveries: %d\n"
                                    "\tTotal requests by type: (Input is approximate)\n",
                                    nextAvailTerm,
                                    nextDelivery );
            for( int i = 0; i < NUM_REQ_TYPES; i++ ) {
               sprintf( buffer + strlen( buffer ),
                           "\t\t%-20s: %d\n", requestNames[i], sumRequests[i] );
            }
            WriteError( "TerminateExtension", buffer );
   } catch( char *msg ) {
            WriteError( "Terminate Error", msg );
            free( msg );
   }

   return( TRUE );
}

#define STATUS_OK_STR "200 OK"

// send HTTP header and HTML response back to browser
static void writeHTML( EXTENSION_CONTROL_BLOCK *pECB, char *html )
/****************************************************************/
{
   HSE_SEND_HEADER_EX_INFO         header_ex_info;
   DWORD                                    bytes_to_write;
   char                                header_and_html[10000];

   // length of html
   bytes_to_write = (DWORD)strlen( html );

   sprintf( header_and_html,
               "Content-Length: %d\r\nContent-Type: text/html\r\n\r\n%s",
```

```
                bytes_to_write,
                html );

    _ASSERT_LEN_VALID( header_and_html );

    // lenght of http header + html
    bytes_to_write += strlen( header_and_html + bytes_to_write );

    header_ex_info.pszStatus = STATUS_OK_STR;
    header_ex_info.pszHeader = header_and_html;
    header_ex_info.cchStatus = (DWORD)( sizeof( STATUS_OK_STR ) - 1 );
    header_ex_info.cchHeader = bytes_to_write;
    header_ex_info.fKeepConn = TRUE;

    // send HTTP header + HTML body
    pECB->ServerSupportFunction( pECB->ConnID,
                                 HSE_REQ_SEND_RESPONSE_HEADER_EX,
                                 &header_ex_info,
                                 NULL,
                                 NULL );
}

// Fatal error for this request.
// The request could not be completed so send back a response with error text
static void sendFatalError( EXTENSION_CONTROL_BLOCK *pECB,
                            char                    *details,
                            int                      term_id = NO_INT_VALUE )
/*****************************************************************************/
{
    char buffer[2048];

    sprintf( buffer,
                HTML_ERROR_PREFIX
                HTML_PRE_BODY
                "<H2><FONT color=RED>ERROR:</FONT></H2><B>%s</B>\r\n<P>",
                details );
    if( term_id == NO_INT_VALUE ) {
            strcat( buffer, "<A href=\"/tpcc\">Click here to go to welcome page</A>" );
    } else {
            sprintf( buffer + strlen( buffer ),
                        "<form method=\"get\" action=\"/tpcc\">\r\n"
                        "<input type=hidden name=\"term_id\" value=\"%d\">"
                        "<PRE>Use the browser's back button or "
                        "<input type=submit name=\"req\" value=\"Menu\"></PRE>",
                        term_id );
    }
    strcat( buffer, HTML_POST_BODY );
    writeHTML( pECB, buffer );

    sprintf( buffer,
                "Parameters: \"%s\"\n%s",
                pECB->lpszQueryString,
                details );
    numErrors++;
    WriteError( "HTTP request Error", buffer );
}

// fill value with value of name param, upto len value_len
// return FALSE if name is not found
static a_bool getStringParam( EXTENSION_CONTROL_BLOCK  *pECB,
                              char                     *name,
                              char                     *value,
                              int                       value_len )
/*****************************************************************/
{
    char *  query_str = pECB->lpszQueryString;
    char *  found_str;
```

```
    int         i;

    found_str = strstr( query_str, name );
    while( found_str != NULL ) {
            found_str += strlen( name );
            if( *found_str != '=' ) {
                found_str = strstr( query_str, name );
                continue;
            }
            // we have found param_name=, found_str is on '=' char
            found_str++;
            for( i = 0; i < value_len - 1; i++ ) {
                if( *found_str == '\0' || *found_str == '&' ) {
                        // end of parameter value
                        break;
                }
                *value++ = *found_str++;
            }
            *value = '\0';
            return( TRUE );
    }
    return( FALSE );
}

// Return the integer value of the name param.
// Returns NO_INT_VALUE if the parameter does not exist or is empty.
// If allow_empty is FALSE and the parameter exists but has an empty string,
// an error is thrown.
// If allow_missing is FALSE and the parameter does not exist,
// an error is thrown.
// If there is a conversion error, an error is thrown.
static int getIntParam( EXTENSION_CONTROL_BLOCK         *pECB,
                        char                            *name,
                        a_bool                          allow_empty = FALSE,
                        a_bool                          allow_missing = FALSE )
/*****************************************************************************/
{
    char conv_buf[20];
    char err_buf[100] = "";
    char *conv_end;
    int value;

    if( getStringParam( pECB, name, conv_buf, sizeof( conv_buf ) ) ) {
            if( conv_buf[0] == '\0' ) {
                if( allow_empty ) {
                        return( NO_INT_VALUE );
                } else {
                        sprintf( err_buf, "empty value for parameter \"%s\"", name );
                        ThrowError( err_buf );
                }
            }
            value = strtol( conv_buf, &conv_end, 10 );
            if( *conv_end == '\0' ) {
                if( value < 0 ) {
                        sprintf( err_buf, "\"%s\" parameter cannot be negative", name );
                        ThrowError( err_buf );
                }
                return( value );
            } else {
                sprintf( err_buf,
                        "cannot convert parameter \"%s\" value \"%s\" to an integer",
                        name,
                        conv_buf );
                ThrowError( err_buf );
            }
    } else if( !allow_missing ) {
            sprintf( err_buf, "\"%s\" parameter is missing", name );
```

```
            ThrowError( err_buf );
    }
    return( NO_INT_VALUE );
}

// Return the double value of the name param.
// Throws an error on a conversion error, if the parameter does not exist or
// if the parameter is empty
static double getDoubleParam( EXTENSION_CONTROL_BLOCK            *pECB,
                                    char                         *name )
/*************************************************************/
{
    char conv_buf[30];
    char err_buf[100] = "";
    char *conv_end;
    double value;

    if( getStringParam( pECB, name, conv_buf, sizeof( conv_buf ) ) ) {
            if( conv_buf[0] == '\0' ) {
                sprintf( err_buf, "empty value for parameter \"%s\"", name );
                ThrowError( err_buf );
            }
            value = strtod( conv_buf, &conv_end );
            if( *conv_end == '\0' ) {
                return( value );
            } else {
                sprintf( err_buf,
                        "cannot convert parameter \"%s\" value \"%s\" to a decimal",
                        name,
                        conv_buf );
                ThrowError( err_buf );
            }
    } else {
            sprintf( err_buf, "\"%s\" parameter is missing", name );
            ThrowError( err_buf );
    }
    // should never get here
    return( -1 );
}

// return the type of request based on the path in the URL.
static inline a_req_type getReqType( EXTENSION_CONTROL_BLOCK *pECB )
/*****************************************************************/
{
    int i;
    char buffer[25] = "uninit";

    getStringParam( pECB, "req", buffer, sizeof( buffer ) );

    for( i = 0; i < NUM_REQ_TYPES; i++ ) {
            if( strcmp( buffer, requestNames[i] ) == 0 ) {
                return( (a_req_type)i );
            }
    }
    return( REQ_WELCOME );
}

// helper for getAndInitTermForReq
// do terminal initialization required for each request
inline static void activateTerm( int term_id )
/*********************************************/
{
    terminal[term_id].active = TRUE;
    terminal[term_id].last_tick_count = GetTickCount();
}

// set maxW_ID if it is not already set.
```

```
// It will get set on the first terminal connection.
static void setMaxW_ID( ITxn * conn )
/***********************************/
{
   if( maxW_ID == 0 ) {
          conn->get_maximum();
          // Don't get the mutex until after get_maximum, since get_maximum
          // can throw an exception.  This ensures we always do the mutex.give.
          termAddDeleteMutex.get();
          maxW_ID = conn->_params.max_w_id;
          termAddDeleteMutex.give();
   }
}


// Call at the beginning of each web request (except welcome) to determine the
// term_id and initialize terminal[term_id] for the request.
// Fill term_id with the terminal id for the request.
// If create is TRUE, create a new terminal if necessary.
static void getAndInitTermForReq( EXTENSION_CONTROL_BLOCK   *pECB,
                                  int                       *term_id,
                                  a_bool                    create )
/*********************************************************************/
{
   char    buffer[100];

   *term_id = getIntParam( pECB, "term_id", FALSE, create );
   if( *term_id >= 0 && *term_id < nextAvailTerm ) {
          // valid term_id passed as a parameter
          if( terminal[*term_id].active ) {
             sprintf( buffer, "term_id %d already active", *term_id );
             *term_id = NO_INT_VALUE;
             ThrowError( buffer );
          } else if( terminal[*term_id].conn == NULL ) {
             sprintf( buffer, "term_id %d connection is NULL", *term_id );
             *term_id = NO_INT_VALUE;
             ThrowError( buffer );
          }
          activateTerm( *term_id );
   } else if( create && *term_id == NO_INT_VALUE ) {
          termAddDeleteMutex.get();
          // don't do anything between the mutex.get and mutex.give that
          // can throw an exception (otherwise the mutex.give won't be done).
          if( nextAvailTerm < maxTerminals ) {
             *term_id = nextAvailTerm;
             nextAvailTerm++;
          } else {
             // loop though all terminals looking for a terminal which
             // hasn't been used in the last five minutes.
             unsigned int min_time = UINT_MAX;
             int min_time_term = 0;

             for( int i = 0; i < maxTerminals; i++ ) {
                    if( terminal[i].last_tick_count < min_time ) {
                       min_time = terminal[i].last_tick_count;
                       min_time_term = i;
                    }
             }
             // GetTickCount is in ms, compare with 5 minutes in ms.
             if( GetTickCount() - min_time < 1000 * 60 * 5 ) {
                    termAddDeleteMutex.give();
                    sprintf( buffer,
                           "Out of term_ids "
                                "(all %d term_ids used in last %d seconds)",
                                maxTerminals,
                                ( GetTickCount() - min_time ) / 1000 );
```

```
                ThrowError( buffer );
            }
            // terminal has not been used in 5 minutes.  Reuse it.
            deleteTerm( min_time_term );
            *term_id = min_time_term;
        }
        activateTerm( *term_id );
        termAddDeleteMutex.give();
        terminal[*term_id].w_id = 1; // set again below
        terminal[*term_id].d_id = 1; // set again below
        // new_Txn will throw an exception if it fails.
        terminal[*term_id].conn = new_Txn();
        setMaxW_ID( terminal[*term_id].conn );
        // maxW_ID is only valid after setMaxW_ID for the first connection
        terminal[*term_id].w_id = getIntParam( pECB, "W_ID" );
        terminal[*term_id].d_id = getIntParam( pECB, "D_ID" );
    } else {
        sprintf( buffer, "invalid term_id %d", *term_id );
        *term_id = NO_INT_VALUE;
        ThrowError( buffer );
    }
}

// Call at the end of each web request (except welcome) to
// finalize terminal[term_id] for the request.
inline static void finiTermForReq( int term_id )
/*********************************************/
{
    if( term_id >= 0 && term_id < maxTerminals ) {
        terminal[term_id].active = FALSE;
    }
}

// the first HTML form of the application
static void welcomeForm( EXTENSION_CONTROL_BLOCK *pECB )
/*****************************************************/
{
    char buffer[1024];

    sprintf( buffer,
            HTML_PRE_BODY
            "<H2>Welcome to the iAnywhere Solutions TPC-C Application</H2>\r\n"
            "<form method=\"get\" action=\"/tpcc\">\r\n"
            "<input type=hidden name=\"req\" value=\"%s\">\r\n"
            "<PRE>Home Warehouse: <input type=text name=W_ID size=5 maxlength=4>\r\n"
            "District for Stock-Level: <input type=text name=D_ID size=2 maxlength=2>\r\n"
            "<input type=submit value=\"Continue\">\r\n"
            "</PRE></form>"
            HTML_POST_BODY,
            requestNames[REQ_MENU] );
    _ASSERT_LEN_VALID( buffer );
    writeHTML( pECB, buffer );
}

static void menuInputAndPostHTML( char *buffer, int term_id )
/***********************************************************/
{
    sprintf( buffer,
            "\r\n<form method=\"get\" action=\"/tpcc\">"
            "<input type=hidden name=term_id value='%d'>"
            "<input type=hidden name=req value='Input'>"
            "Select a transaction: "
            "<input type=submit name=type value=\"New-Order\">"
            "<input type=submit name=type value=\"Payment\">"
            "<input type=submit name=type value=\"Order-Status\">"
            "<input type=submit name=type value=\"Delivery\">"
            "<input type=submit name=type value=\"Stock-Level\">"
```

```
                "</PRE></form>",
                term_id );
}

// the next HTML form after the welcome form
static void menuForm( EXTENSION_CONTROL_BLOCK *pECB, int term_id )
/*****************************************************************/
{
    char buffer[5000];

    strcpy( buffer, HTML_PRE_BODY "<PRE>" );
    menuInputAndPostHTML( buffer + strlen( buffer ), term_id );
    _ASSERT_LEN_VALID( buffer );
    writeHTML( pECB, buffer );
}

char *inputBody[NUM_TXN_TYPES] = {
// New Order
"                              New Order\r\n\
Warehouse: %4d   District: <input type=text name=D_ID size=1 maxlength=2>                    Date:\r\n\
Customer:  <input type=text name=C_ID size=1 maxlength=4>  Name:                  Credit:     %%Disc:\r\n\
Order Number:          Number of Lines:          W_tax:         D_tax:\r\n\
\r\n\
 Supp_W  Item_Id  Item Name                Qty  Stock  B/G  Price    Amount\r\n\
  <input type=text name=OL_SWID1 size=1 maxlength=4>  <input type=text name=OL_IID1 size=3 maxlength=6>
<input type=text name=OL_Q1 size=1 maxlength=2>\r\n\
  <input type=text name=OL_SWID2 size=1 maxlength=4>  <input type=text name=OL_IID2 size=3 maxlength=6>
<input type=text name=OL_Q2 size=1 maxlength=2>\r\n\
  <input type=text name=OL_SWID3 size=1 maxlength=4>  <input type=text name=OL_IID3 size=3 maxlength=6>
<input type=text name=OL_Q3 size=1 maxlength=2>\r\n\
  <input type=text name=OL_SWID4 size=1 maxlength=4>  <input type=text name=OL_IID4 size=3 maxlength=6>
<input type=text name=OL_Q4 size=1 maxlength=2>\r\n\
  <input type=text name=OL_SWID5 size=1 maxlength=4>  <input type=text name=OL_IID5 size=3 maxlength=6>
<input type=text name=OL_Q5 size=1 maxlength=2>\r\n\
  <input type=text name=OL_SWID6 size=1 maxlength=4>  <input type=text name=OL_IID6 size=3 maxlength=6>
<input type=text name=OL_Q6 size=1 maxlength=2>\r\n\
  <input type=text name=OL_SWID7 size=1 maxlength=4>  <input type=text name=OL_IID7 size=3 maxlength=6>
<input type=text name=OL_Q7 size=1 maxlength=2>\r\n\
  <input type=text name=OL_SWID8 size=1 maxlength=4>  <input type=text name=OL_IID8 size=3 maxlength=6>
<input type=text name=OL_Q8 size=1 maxlength=2>\r\n\
  <input type=text name=OL_SWID9 size=1 maxlength=4>  <input type=text name=OL_IID9 size=3 maxlength=6>
<input type=text name=OL_Q9 size=1 maxlength=2>\r\n\
  <input type=text name=OL_SWID10 size=1 maxlength=4>  <input type=text name=OL_IID10 size=3 maxlength=6>
<input type=text name=OL_Q10 size=1 maxlength=2>\r\n\
  <input type=text name=OL_SWID11 size=1 maxlength=4>  <input type=text name=OL_IID11 size=3 maxlength=6>
<input type=text name=OL_Q11 size=1 maxlength=2>\r\n\
  <input type=text name=OL_SWID12 size=1 maxlength=4>  <input type=text name=OL_IID12 size=3 maxlength=6>
<input type=text name=OL_Q12 size=1 maxlength=2>\r\n\
  <input type=text name=OL_SWID13 size=1 maxlength=4>  <input type=text name=OL_IID13 size=3 maxlength=6>
<input type=text name=OL_Q13 size=1 maxlength=2>\r\n\
  <input type=text name=OL_SWID14 size=1 maxlength=4>  <input type=text name=OL_IID14 size=3 maxlength=6>
<input type=text name=OL_Q14 size=1 maxlength=2>\r\n\
  <input type=text name=OL_SWID15 size=1 maxlength=4>  <input type=text name=OL_IID15 size=3 maxlength=6>
<input type=text name=OL_Q15 size=1 maxlength=2>\r\n\
Execution Status:                                  Total:\r\n\
\r\n\r\n",

// Payment
"                              Payment\r\n\
Date:\r\n\
\r\n\
Warehouse: %4d                  District: <input type=text name=D_ID size=1 maxlength=2>\r\n\
\r\n\
\r\n\
\r\n\
\r\n\
```

```
Customer: <input type=text name=C_ID size=1 maxlength=4> Cust-Warehouse: <input type=text name=C_W_ID size=1
maxlength=4> Cust-District: <input type=text name=C_D_ID size=1 maxlength=2>\r\n\
Name:                    <input type=text name=C_LAST size=16 maxlength=16>     Since:\r\n\
                                        Credit:\r\n\
                                        %%Disc:\r\n\
                                        Phone:\r\n\
\r\n\
Amount Paid:          $<input type=text name=H_AMOUNT size=4 maxlength=7>      New Cust-Balance:\r\n\
Credit Limit:\r\n\
\r\n\
Cust-Data:\r\n\
\r\n\
\r\n\
\r\n\
\r\n\r\n\r\n\r\n",

// Order-Status
"                          Order-Status\r\n\
Warehouse: %4d   District: <input type=text name=D_ID size=1 maxlength=2>\r\n\
Customer: <input type=text name=C_ID size=1 maxlength=4>  Name:                  <input type=text name=C_LAST size=16
maxlength=16>\r\n\
Cust-Balance:\r\n\
\r\n\
Order-Number:          Entry-Date:                  Carrier-Number:\r\n\
Supply-W     Item-Id    Qty     Amount      Delivery-Date\r\n\
\r\n\
\r\n\
\r\n\
\r\n\
\r\n\
\r\n\
\r\n\
\r\n\
\r\n\
\r\n\
\r\n\
\r\n\
\r\n\
\r\n\
\r\n\
\r\n\
\r\n\
\r\n",

// Delivery
"                           Delivery\r\n\
Warehouse: %4d\r\n\
\r\n\
Carrier Number: <input type=text name=O_CARRIER_ID size=1 maxlength=2>\r\n\
\r\n\
Execution Status:\r\n\
\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n",

// Stock-Level
"                          Stock-Level\r\n\
Warehouse: %4d   District: %2d\r\n\
\r\n\
Stock Level Threshold: <input type=text name=THRESHOLD size=1 maxlength=2>\r\n\
\r\n\
low stock:\r\n\
\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n"
};

static void inputForm( EXTENSION_CONTROL_BLOCK *pECB, int term_id )
/*****************************************************************/
{
   char       buffer[5000];
   int                buffer_len;
   char       txn_type_buf[20];
```

```
    int                     txn_type;

    if( !getStringParam( pECB, "type", txn_type_buf, sizeof( txn_type_buf ) ) ) {
            ThrowError( "missing type on Input request" );
    }
    for( txn_type = 0; txn_type < NUM_TXN_TYPES; txn_type++ ) {
            if( strcmp( txn_type_buf, TxnName[txn_type] ) == 0 ) break;
    }
    if( txn_type == NUM_TXN_TYPES ) {
            ThrowError( "invalid type on Input request" );
    }

    sprintf( buffer,
                HTML_PRE_BODY
                "<form method=get action='/tpcc'>"
                "<input type=hidden name=term_id value='%d'>"
                "<PRE>",
//              "<PRE>12345678901234567890123456789012345678901234567890123456789012345678901234567890\r\n",
                term_id );
    buffer_len = strlen( buffer );
    sprintf( buffer + buffer_len,
                inputBody[txn_type],
                terminal[term_id].w_id,     // optional parameters
                terminal[term_id].d_id );
    buffer_len += strlen( buffer + buffer_len );
    sprintf( buffer + buffer_len,
                "<input type=submit name=req value=\"Process %s\"> "
                "<input type=submit name=req value=Menu>"
                "</PRE></form></BODY></HTML>",
                TxnName[txn_type] );

    _ASSERT_LEN_VALID( buffer );
    writeHTML( pECB, buffer );
}

// helper for newOrderSubmit.  fill new_order structure with parameters
// which user entered on New Order input screen.
static void getNewOrderParameters( EXTENSION_CONTROL_BLOCK  *pECB,
                                   a_new_order              *new_order,
                                   int                      term_id )
/*********************************************************************/
{
    int         w_id = terminal[term_id].w_id;
    int         ol_swid;
    int         ol_iid;
    int         ol_q;
    int         i;
    int         num_lines = 0;
    a_bool  all_local = TRUE;
    char    param_name[20];
    char    err_buf[100];

    for( i = 1; i <= 15; i++ ) {
            sprintf( param_name, "OL_SWID%d", i );
            ol_swid = getIntParam( pECB, param_name, TRUE );
            sprintf( param_name, "OL_IID%d", i );
            ol_iid = getIntParam( pECB, param_name, TRUE );
            sprintf( param_name, "OL_Q%d", i );
            ol_q = getIntParam( pECB, param_name, TRUE );
            if( ol_swid == NO_INT_VALUE ) {
                if( ol_iid != NO_INT_VALUE || ol_q != NO_INT_VALUE ) {
                        sprintf( err_buf,
                                "row %d line item has empty Supp_w but value for Item_ID or Qty",
                                i );
                        ThrowError( err_buf );
                }
                // line has no data set, skip it
```

```
            } else {
                if( ol_iid == NO_INT_VALUE || ol_q == NO_INT_VALUE ) {
                        sprintf( err_buf,
                                "row %d line item has value for Supp_w but empty Item_ID or Qty",
                                i );
                        ThrowError( err_buf );
                }
                // line has valid data
                if( ol_swid != w_id ) {
                        all_local = FALSE;
                }
                new_order->ol[num_lines].ol_supply_w_id = ol_swid;
                new_order->ol[num_lines].ol_i_id        = ol_iid;
                new_order->ol[num_lines].ol_quantity    = ol_q;
                num_lines++;
            }
    }
    if( num_lines == 0 ) {
            ThrowError( "no order items entered" );
    }
    new_order->w_id                 = w_id;
    new_order->d_id    = getIntParam( pECB, "D_ID" );
    new_order->c_id    = getIntParam( pECB, "C_ID" );
    new_order->o_ol_cnt             = num_lines;
    new_order->o_all_local  = all_local;
}

// submit data entered in New Order input screen,
// and display the New Order output screen.
static void newOrderSubmit( EXTENSION_CONTROL_BLOCK *pECB, int term_id )
/*********************************************************************/
{
    ITxn        *conn = terminal[term_id].conn;
    a_new_order     *new_order = &conn->_params.new_order;
    char        buffer[8000];
    int                     buffer_len;
    int                     i;

    memset( new_order, 0, sizeof( a_new_order ) );

    // set new_order input fields from web request parameters
    getNewOrderParameters( pECB, new_order, term_id );

    // execute new_order transaction
    conn->new_order();

    sprintf( buffer,
                HTML_PRE_BODY
                "<PRE>" SP30 "  New Order\r\n"
//              "12345678901234567890123456789012345678901234567890123456789012345678901234567890\r\n"
                "Warehouse: %4d   District: %2d" SP20 "    Date: ",
                new_order->w_id,
                new_order->d_id );

    buffer_len = strlen( buffer );
    if( new_order->o_commit_flag ) {
            // transaction committed
            sprintf( buffer + buffer_len,
                        "%02d-%02d-%04d %02d:%02d:%02d\r\n"
                        "Customer:  %4d   "
                          "Name: %-16s   Credit: %-2s   %%Disc: %5.2f\r\n"
                        "Order Number: %8d  Number of Lines: %2d        "
                          "W_tax: %5.2f   D_tax: %5.2f\r\n\r\n",
                new_order->o_entry_d.day,
                new_order->o_entry_d.month,
                new_order->o_entry_d.year,
                new_order->o_entry_d.hour,
```

```
                new_order->o_entry_d.minute,
                new_order->o_entry_d.second,
                new_order->c_id,
                new_order->c_last,
                new_order->c_credit,
                new_order->c_discount,
                new_order->o_id,
                new_order->o_ol_cnt,
                new_order->w_tax,
                new_order->d_tax );
    } else {
            // transaction rolled back
            new_order->o_ol_cnt = 0;    // don't display any order items
            sprintf( buffer + buffer_len,
                        "\r\n"
                        "Customer:  %4d   Name: %-16s  Credit: %-2s    %%Disc:\r\n"
                        "Order Number: %8d  Number of Lines:           "
                          "W_tax:         D_tax:\r\n\r\n",
                        new_order->c_id,
                        new_order->c_last,
                        new_order->c_credit,
                        new_order->o_id );
    }
    buffer_len += strlen( buffer + buffer_len );
    sprintf( buffer + buffer_len,
                " Supp_W  Item_Id  Item Name" SP10 "       Qty  Stock  B/G  Price"
                        "    Amount\r\n" );
    buffer_len += strlen( buffer + buffer_len );

    for( i = 0; i < new_order->o_ol_cnt; i++ ) {
            sprintf( buffer + buffer_len,
                        " %4d   %6d   %-24s  %2d    %3d    %1s   $%6.2f  $%7.2f\r\n",
                        new_order->ol[i].ol_supply_w_id,
                        new_order->ol[i].ol_i_id,
                        new_order->ol[i].ol_i_name,
                        new_order->ol[i].ol_quantity,
                        new_order->ol[i].ol_stock,
                        new_order->ol[i].ol_brand_generic,
                        new_order->ol[i].ol_i_price,
                        new_order->ol[i].ol_amount );
            buffer_len += strlen( buffer + buffer_len );
    }
    for( ; i < 15; i++ ) {
            strcpy( buffer + buffer_len, "\r\n" );
            buffer_len += strlen( buffer + buffer_len );
    }
    if( new_order->o_commit_flag ) {
            sprintf( buffer + buffer_len,
                        "Execution Status:" SP40 "    Total:  $%8.2f\r\n\r\n",
                        new_order->total_amount );
    } else {
            strcat( buffer + buffer_len,
                       "Execution Status: Item number is not valid         " SP10
                       "Total:\r\n\r\n" );
    }
    buffer_len += strlen( buffer + buffer_len );
    menuInputAndPostHTML( buffer + buffer_len, term_id );
    _ASSERT_LEN_VALID( buffer );
    writeHTML( pECB, buffer );
}

// helper for paymentSubmit.  fill payment structure with parameters
// which user entered on Payment input screen.
static void getPaymentParameters( EXTENSION_CONTROL_BLOCK   *pECB,
                                            a_payment                       *payment,
                                            int                             term_id )
/*****************************************************************/
```

```
{
   a_c_id c_id;

   payment->w_id  = terminal[term_id].w_id;
   payment->d_id  = getIntParam( pECB, "D_ID" );
   c_id           = getIntParam( pECB, "C_ID", TRUE );
   payment->c_d_id        = getIntParam( pECB, "C_D_ID" );
   payment->c_w_id        = getIntParam( pECB, "C_W_ID" );
   payment->h_amount  = getDoubleParam( pECB, "H_AMOUNT" );
   getStringParam( pECB, "C_LAST", payment->c_last, LAST_LEN + 1 );

   if( c_id == NO_INT_VALUE ) {
         c_id = 0;
   }
   if( c_id == 0 && payment->c_last[0] == '\0' ) {
         ThrowError( "One of Customer ID or Customer Last Name"
                     " must be entered" );
   }
   payment->c_id = c_id;
}

// submit data entered in Payment input screen,
// and display the Payment output screen.
static void paymentSubmit( EXTENSION_CONTROL_BLOCK *pECB, int term_id )
/*********************************************************************/
{
   ITxn       *conn = terminal[term_id].conn;
   a_payment          *payment = &conn->_params.payment;
   char       buffer[8000];
   int                buffer_len;
   char       w_zip_buf[6];
   char       d_zip_buf[6];
   char       c_zip_buf[6];
   char       c_phone_buf1[7];
   char       c_phone_buf2[4];
   char       c_phone_buf3[4];
   char       c_data_buf1[51];
   char       c_data_buf2[51];
   char       c_data_buf3[51];

   memset( payment, 0, sizeof( a_payment ) );

   // set new_order input fields from web request parameters
   getPaymentParameters( pECB, payment, term_id );

   // execute payment transaction
   conn->payment();

   sprintf( buffer,
            HTML_PRE_BODY
            "<PRE>" SP30 "    Payment\r\n"
            "Date: %02d-%02d-%04d %02d:%02d:%02d\r\n"
            "\r\n"
            "Warehouse: %4d" SP20 "      District: %4d\r\n"
//          "12345678901234567890123456789012345678901234567890123456789012345678901234567890\r\n"
            "%-20s " SP20 "%-20s\r\n"
            "%-20s " SP20 "%-20s\r\n"
            "%-20s %-2s %-5s-%-4s       %-20s %-2s %-5s-%-4s\r\n"
            "\r\n"
            "Customer: %4d  Cust-Warehouse: %4d  Cust-District: %2d\r\n"
            "Name:   %-16s %-2s %-16s     Since:  %02d-%02d-%04d\r\n"
            "        %-20s " SP20 "Credit: %-2s\r\n"
            "        %-20s " SP20 "%%Disc:  %5.2f\r\n"
            "        %-20s %-2s %-5s-%-4s       Phone:  %6s-%3s-%3s-%4s\r\n"
            "\r\n"
            "Amount Paid:" SP10 "$%7.2f      New Cust-Balance: $%14.2f\r\n"
            "Credit Limit:   $%13.2f\r\n"
```

```
                "\r\n",
                payment->h_date.day,
                payment->h_date.month,
                payment->h_date.year,
                payment->h_date.hour,
                payment->h_date.minute,
                payment->h_date.second,
                payment->w_id,
                payment->d_id,
                payment->w_street_1,
                payment->d_street_1,
                payment->w_street_2,
                payment->d_street_2,
                payment->w_city,
                payment->w_state,
                leftStr( w_zip_buf, payment->w_zip, 5 ),
                payment->w_zip + 5,
                payment->d_city,
                payment->d_state,
                leftStr( d_zip_buf, payment->d_zip, 5 ),
                payment->d_zip + 5,
                payment->c_id,
                payment->c_w_id,
                payment->c_d_id,
                payment->c_first,
                payment->c_middle,
                payment->c_last,
                payment->c_since.day,
                payment->c_since.month,
                payment->c_since.year,
                payment->c_street_1,
                payment->c_credit,
                payment->c_street_2,
                payment->c_discount,
                payment->c_city,
                payment->c_state,
                leftStr( c_zip_buf, payment->c_zip, 5 ),
                payment->c_zip + 5,
                leftStr( c_phone_buf1, payment->c_phone, 6 ),
                leftStr( c_phone_buf2, payment->c_phone + 6, 3 ),
                leftStr( c_phone_buf3, payment->c_phone + 9, 3 ),
                payment->c_phone + 12,
                payment->h_amount,
                payment->c_balance,
                payment->c_credit_lim );
    buffer_len = strlen( buffer );
    if( strcmp( payment->c_credit, "BC" ) == 0 ) {
            sprintf( buffer + buffer_len,
                        "Cust-Data: %-50s\r\n"
                        "           %-50s\r\n"
                        "           %-50s\r\n"
                        "           %-50s\r\n\r\n\r\n",
                        leftStr( c_data_buf1, payment->c_data, 50 ),
                        leftStr( c_data_buf2, payment->c_data + 50, 50 ),
                        leftStr( c_data_buf3, payment->c_data + 100, 50 ),
                        payment->c_data + 150 );
    } else {
            strcpy( buffer + buffer_len, "Cust-Data:\r\n\r\n\r\n\r\n\r\n\r\n" );
    }
    buffer_len += strlen( buffer + buffer_len );
    menuInputAndPostHTML( buffer + buffer_len, term_id );
    _ASSERT_LEN_VALID( buffer );
    writeHTML( pECB, buffer );
}

// helper for OrderStatusSubmit.  fill payment structure with parameters
// which user entered on Order-Status input screen.
```

```
static void getOrderStatusParameters( EXTENSION_CONTROL_BLOCK  *pECB,
                                      an_order_status          *order_status,
                                      int                      term_id )
/*****************************************************************************/
{
    a_c_id c_id;

    order_status->w_id          = terminal[term_id].w_id;
    order_status->d_id          = getIntParam( pECB, "D_ID" );
    c_id                = getIntParam( pECB, "C_ID", TRUE );
    getStringParam( pECB, "C_LAST", order_status->c_last, LAST_LEN + 1 );

    if( c_id == NO_INT_VALUE ) {
            c_id = 0;
    }
    if( c_id == 0 && order_status->c_last[0] == '\0' ) {
            ThrowError( "One of Customer ID and Customer Last Name"
                            " must be entered" );
    }
    order_status->c_id = c_id;
}

// submit data entered in Order Status input screen,
// and display the Order Status output screen.
static void orderStatusSubmit( EXTENSION_CONTROL_BLOCK *pECB, int term_id )
/*****************************************************************************/
{
    ITxn        *conn = terminal[term_id].conn;
    an_order_status *order_status = &conn->_params.order_status;
    char        buffer[8000];
    char        carrier_id_buf[20];
    int                 buffer_len;
    int                 i;

    memset( order_status, 0, sizeof( an_order_status ) );

    // set order_status input fields from web request parameters
    getOrderStatusParameters( pECB, order_status, term_id );

    // execute new_order transaction
    conn->order_status();

    if( order_status->ind_carrier_id < 0 ) {
            // display a blank carrier id if NULL
            carrier_id_buf[0] = '\0';
    } else {
            sprintf( carrier_id_buf, "%2d", order_status->o_carrier_id );
    }
    sprintf( buffer,
                HTML_PRE_BODY
                "<PRE>" SP30 "  Order-Status\r\n"
                "Warehouse: %4d   District: %2d\r\n"
                "Customer: %4d   Name: %-16s %-2s %-16s\r\n"
//              "12345678901234567890123456789012345678901234567890123456789012345678901234567890\r\n"
                "Cust-Balance: $%9.2f\r\n"
                "\r\n"
                "Order-Number: %8d   Entry-Date: %02d-%02d-%04d %02d:%02d:%02d"
                        "   Carrier-Number: %2s\r\n"
                "Supply-W     Item-Id    Qty     Amount      Delivery-Date",
                order_status->w_id,
                order_status->d_id,
                order_status->c_id,
                order_status->c_first,
                order_status->c_middle,
                order_status->c_last,
                order_status->c_balance,
                order_status->o_id,
```

```
                order_status->o_entry_d.day,
                order_status->o_entry_d.month,
                order_status->o_entry_d.year,
                order_status->o_entry_d.hour,
                order_status->o_entry_d.minute,
                order_status->o_entry_d.second,
                carrier_id_buf );

    buffer_len = strlen( buffer );
    for( i = 0; i < order_status->o_ol_cnt; i++ ) {
            if( order_status->ol[i].ind_delivery_d < 0 ) {
                // NULL delivery date, use blanks for date
                sprintf( buffer + buffer_len,
                            "\r\n  %4d       %6d     %2d     $%8.2f",
                            order_status->ol[i].ol_supply_w_id,
                            order_status->ol[i].ol_i_id,
                            order_status->ol[i].ol_quantity,
                            order_status->ol[i].ol_amount );
            } else {
                // non-NULL delivery date
                sprintf( buffer + buffer_len,
                            "\r\n  %4d       %6d     %2d     $%8.2f      "
                                "%02d-%02d-%04d",
                            order_status->ol[i].ol_supply_w_id,
                            order_status->ol[i].ol_i_id,
                            order_status->ol[i].ol_quantity,
                            order_status->ol[i].ol_amount,
                            order_status->ol[i].ol_delivery_d.day,
                            order_status->ol[i].ol_delivery_d.month,
                            order_status->ol[i].ol_delivery_d.year );
            }
            buffer_len += strlen( buffer + buffer_len );
    }
    for( ; i < 15; i++ ) {
            strcpy( buffer + buffer_len, "\r\n" );
            buffer_len += strlen( buffer + buffer_len );
    }
    strcpy( buffer + buffer_len, "\r\n\r\n" );
    buffer_len += strlen( buffer + buffer_len );
    menuInputAndPostHTML( buffer + buffer_len, term_id );
    _ASSERT_LEN_VALID( buffer );
    writeHTML( pECB, buffer );
}

#ifdef DUMP_DELIVERY_IN_OUT
    #define _DUMP_DELIVERY_IN_OUT( context ) {                                          \
            char err_buff[100];                                                 \
            sprintf( err_buff, "delivery thread %d", GetCurrentThreadId() );\
            WriteError( context, err_buff ); }
#else
    #define _DUMP_DELIVERY_IN_OUT( context )
#endif

// Data passed between PostQueuedCompletionStatus & GetQueueCompletionStatus
// for delivery threads
// stop_if_zero - if 0 stop thread
// index - index of deliveries array element to perform delivery on
// LPOVERLAPPED - unused
static void __cdecl deliveryThread( void * )
/*****************************************/
{
    LPOVERLAPPED            o;
    DWORD                       stop_if_zero;
    ULONG_PTR                   index;
    ITxn                    *conn = NULL;
    a_delivery                  *delivery_txn;      // structure in conn
    a_delivery_info         *delivery_info;     // element in deliveries array
```

```
    int                                 i;
    FILETIME                            file_time;

    InterlockedIncrement( &numDeliveryThreadsRunning );
    try {
            conn = new_Txn();
            delivery_txn = &conn->_params.delivery;
            _DUMP_DELIVERY_IN_OUT( "Thread starting" );
            for( ; ; ) {
               GetQueuedCompletionStatus( deliveryCompletionPort,
                                          &stop_if_zero,
                                          &index,
                                          &o,
                                          INFINITE );
               if( stop_if_zero == 0 ) break;
               _DUMP_DELIVERY_IN_OUT( "starting delivery" );
               delivery_info = &deliveries[index];
               _ASSERT( delivery_info->queued_time != 0 );
               _ASSERT( delivery_info->completed_time == 0 );

               // execute delivery transaction
               memset( delivery_txn, 0, sizeof( a_delivery ) );
               delivery_txn->w_id = delivery_info->w_id;
               delivery_txn->o_carrier_id = delivery_info->carrier_id;

               try {
                       conn->delivery();

                       if( delivery_txn->exec_status_code == eOK ) {
                          for( i = 0; i < 10; i++ ) {
                                  delivery_info->o_id[i] = delivery_txn->o_id[i];
                          }
                          GetSystemTimeAsFileTime( &file_time );
                          delivery_info->completed_time = ToTime( &file_time );
                       } else {
                          numErrors++;
                          WriteError( "Delivery Thread Error",
                                          "exec_stats_code != e_OK" );
                       }
               } catch( char *msg ) {
                       numErrors++;
                       WriteError( "Delivery Thread Error", msg );
                       free( msg );
               }
               _DUMP_DELIVERY_IN_OUT( "finished delivery" );
            }
    } catch( char *msg ) {
            numErrors++;
            WriteError( "Delivery Thread Error (thread aborted)", msg );
            free( msg );
    }
    if( conn != NULL ) {
            free_Txn( conn );
    }
    InterlockedDecrement( &numDeliveryThreadsRunning );
    _DUMP_DELIVERY_IN_OUT( "Thread stopped" );
}

static void writeDeliveryResults( void )
/***************************************/
{
    int                         i;
    int                         district;
    char                file_name[100];
    FILE                *file;
    a_delivery_info     *delivery_info;
    char                queued_time_buf[20];
```

```
char            completed_time_buf[20];
int                     num_did_not_complete = 0;
int                     num_skipped = 0;
int                     num_completed;
a_time          process_time;
a_time          min_time = ToTime( 1000 );
a_time          max_time = 0;
a_time          sum_time = 0;
FILETIME                file_time;
int                     num_under_90p_max = 0;
a_time          start_time = 0;
a_time          end_time = 0;

if( nextDelivery > 0 ) {
        start_time = deliveries[0].completed_time;
        end_time = deliveries[nextDelivery-1].completed_time;
}
if( start_time == 0 || end_time == 0 ) {
        GetSystemTimeAsFileTime( &file_time );
        start_time = ToTime( &file_time );
        end_time = start_time;
}
SetStatFileName( file_name, "isapi_deliveries", start_time, "txt" );
file = fopen( file_name, "w" );
if( file == NULL ) {
        numErrors++;
        WriteError( "writeDeliveryResults Error",
                "Unable to open isapi_deliveries file" );
        return;
}
for( i = 0; i < nextDelivery; i++ ) {
        delivery_info = &deliveries[i];

        TimeToFileTime( delivery_info->queued_time, &file_time );
        FileTimeToTimeStr( &file_time, queued_time_buf );
        if( delivery_info->completed_time == 0 ) {
           // didn't complete
           fprintf( file,
                "Delivery queued at %s DID NOT COMPLETE\n"
                "  warehouse number: %d, carrier number: %d\n\n",
                queued_time_buf,
                delivery_info->w_id,
                delivery_info->carrier_id );
           num_did_not_complete++;
        } else {
           // did complete
           process_time = delivery_info->completed_time
                        - delivery_info->queued_time;
           TimeToFileTime( delivery_info->completed_time, &file_time );
           FileTimeToTimeStr( &file_time, completed_time_buf );
           fprintf( file,
                "Delivery queued at %s completed at %s (%.3fsec)\n"
                "\twarehouse number: %4d, carrier number: %2d\n",
                queued_time_buf,
                completed_time_buf,
                ToSec( process_time ),
                delivery_info->w_id,
                delivery_info->carrier_id );
           for( district = 0; district < 10; district++ ) {
                if( delivery_info->o_id[district] == 0 ) {
                   fprintf( file,
                        "\t\tdistrict: %d  SKIPPED\n",
                        district + 1 );
                   num_skipped++;
                } else {
                   fprintf( file,
                        "\t\tdistrict: %2d - order %8d delivered\n",
```

```
                                district + 1,
                                delivery_info->o_id[district] );
                        }
                }
                fprintf( file, "\n" );
                if( process_time < min_time ) {
                        min_time = process_time;
                }
                if( process_time > max_time ) {
                        max_time = process_time;
                }
                sum_time += process_time;
                if( process_time < ToTime( 80.0 ) ) {
                        num_under_90p_max++;
                }
        }
    }

    fprintf( file,
             "\n\nTotal Number of Delivery Transactions: %d\n"
             "Number which did not complete: %d\n"
             "Number of districts which were skipped: %d\n\n",
             nextDelivery,
             num_did_not_complete,
             num_skipped );
    num_completed = nextDelivery - num_did_not_complete;
    if( num_completed > 0 ) {
        fprintf( file, "%-30s %7s %7s %7s   %-20s\n",
                 "DEFERRED EXECUTION TIMES (sec)",
                 "Min", "Average", "Max", "Under 90Percentile Max" );
        fprintf( file,
                 "- %-28s %7.3f %7.3f %7.3f   %6.3f%%\n",
                 "Delivery (deferred portion)",
                 ToSec( min_time ),
                 ToSec( sum_time ) / num_completed,
                 ToSec( max_time ),
                 _PERCENT( num_under_90p_max, num_completed ) );
    }
    fclose( file );
    // Write .dat file
    DumpDeliveryStats( "isapi_deliveries",
                       deliveries,
                       nextDelivery,
                       start_time,
                       end_time,
                       numErrors,
                       maxW_ID );
}

// queue data entered in Delivery input screen,
// and display the Delivery output screen.
static void deliverySubmit( EXTENSION_CONTROL_BLOCK *pECB, int term_id )
/*********************************************************************/
{
    char        buffer[7000];
    a_carrier_id    carrier_id;
    unsigned long   delivery_index;
    a_delivery_info *delivery_info;
    FILETIME            file_time;

    // get carrier ID
    carrier_id = getIntParam( pECB, "O_CARRIER_ID" );

    // get index in deliveries array to queue this transaction
    nextDeliveryMutex.get();
    if( nextDelivery == maxDeliveries ) {
        nextDeliveryMutex.give();
```

```
            ThrowError( "Out of elements in delivery array.  "
                        "No more deliveries can be queued" );
    }
    delivery_index = nextDelivery;
    nextDelivery++;
    // workerThread is not used for REQ_DELIVERY_SUBMIT, so need to update
    // the number of requests here
    sumRequests[REQ_DELIVERY_SUBMIT]++;
    nextDeliveryMutex.give();

    // initialize deliveries element and queue the delivery
    delivery_info = &deliveries[delivery_index];
    delivery_info->w_id = terminal[term_id].w_id;
    delivery_info->carrier_id = carrier_id;

    GetSystemTimeAsFileTime( &file_time );
    delivery_info->queued_time = ToTime( &file_time );

    if( !PostQueuedCompletionStatus( deliveryCompletionPort,
                                     1,  /* don't stop thread */
                                     delivery_index,
                                     NULL ) ) {
            ThrowError( "PostQueueCompletionStatus failed for delivery" );
    }

    // display the output screen
    sprintf( buffer,
                HTML_PRE_BODY
                "<PRE>" SP30 "    Delivery\r\n"
                "Warehouse: %4d\r\n"
//              "12345678901234567890123456789012345678901234567890123456789012345678901234567890\r\n"
                "\r\n"
                "Carrier Number: %2d\r\n"
                "\r\n"
                "Execution Status: Delivery has been queued"
                "\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n",
                delivery_info->w_id,
                carrier_id );
    menuInputAndPostHTML( buffer + strlen( buffer ), term_id );
    _ASSERT_LEN_VALID( buffer );
    writeHTML( pECB, buffer );
}

// submit data entered in Stock-Level input screen,
// and display the Stock-Level output screen.
static void stockLevelSubmit( EXTENSION_CONTROL_BLOCK *pECB, int term_id )
/*********************************************************************/
{
    ITxn        *conn = terminal[term_id].conn;
    a_stock_level   *stock_level = &conn->_params.stock_level;
    char        buffer[7000];
    a_threshold             threshold;

    // get threshold
    threshold = getIntParam( pECB, "THRESHOLD" );

    // execute transaction
    memset( stock_level, 0, sizeof( a_stock_level ) );
    stock_level->w_id = terminal[term_id].w_id;
    stock_level->d_id = terminal[term_id].d_id;
    stock_level->threshold = threshold;
    conn->stock_level();

    // display the output screen
    sprintf( buffer,
                HTML_PRE_BODY
                "<PRE>" SP30 " Stock-Level\r\n"
```

```
                "Warehouse: %4d   District: %2d\r\n"
                "\r\n"
//              "12345678901234567890123456789012345678901234567890123456789012345678901234567890\r\n"
                "Stock Level Threshold: %2d\r\n"
                "\r\n"
                "low stock: %3d"
                "\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n",
                stock_level->w_id,
                stock_level->d_id,
                threshold,
                stock_level->low_stock );
    menuInputAndPostHTML( buffer + strlen( buffer ), term_id );
    _ASSERT_LEN_VALID( buffer );
    writeHTML( pECB, buffer );
}

// debug dumping gear for the doRequest function
#ifdef DUMP_REQ_IN_OUT
    #define _DUMP_REQ_IN_OUT( context )                                                 \
            sprintf( err_buff, "thread %d, term_id %d, req_type %d",    \
                        GetCurrentThreadId(), term_id, req_type );              \
            WriteError( context, err_buff );
#else
    #define _DUMP_REQ_IN_OUT( context )
#endif

#ifdef USE_WORKER_THREADS
// called by HttpExtensionProc or workerThread to process each web page request
static DWORD doRequest( EXTENSION_CONTROL_BLOCK *pECB, a_req_type req_type )
#else
// main entry point called for each web page request
DWORD WINAPI HttpExtensionProc( EXTENSION_CONTROL_BLOCK *pECB )
#endif
/*************************************************************/
{
#if !defined( USE_WORKER_THREADS )
    a_req_type        req_type    = getReqType( pECB );
#endif
    int               term_id       = NO_INT_VALUE;
    char    err_buff[100];

    try {
            if( req_type == REQ_WELCOME ) {
                // no term_id needed.
                welcomeForm( pECB );
            } else {
                // initialize term_id
                getAndInitTermForReq( pECB, &term_id, req_type == REQ_MENU );
                _DUMP_REQ_IN_OUT( "REQUEST IN" );
                switch( req_type ) {
                        case REQ_MENU:
                            menuForm( pECB, term_id );
                            break;
                        case REQ_NEW_ORDER_SUBMIT:
                            newOrderSubmit( pECB, term_id );
                            break;
                        case REQ_PAYMENT_SUBMIT:
                            paymentSubmit( pECB, term_id );
                            break;
                        case REQ_INPUT_FORM:
                            inputForm( pECB, term_id );
                            break;
                        case REQ_ORDER_STATUS_SUBMIT:
                            orderStatusSubmit( pECB, term_id );
                            break;
                        case REQ_DELIVERY_SUBMIT:
                            deliverySubmit( pECB, term_id );
```

```
                        break;
                    case REQ_STOCK_LEVEL_SUBMIT:
                        stockLevelSubmit( pECB, term_id );
                        break;
                    default:
                        sprintf( err_buff, "Invalid request type: %d", req_type );
                        sendFatalError( pECB, err_buff, term_id );
                }
                _DUMP_REQ_IN_OUT( "REQUEST OUT" );
            }
    } catch( char *msg ) {
            // an unexpected error occurred
            sendFatalError( pECB, msg, term_id );
            // msg is always strdup'ed so this is safe
            free( msg );
            _DUMP_REQ_IN_OUT( "REQUEST OUT EXCEPTION" );
    }
    if( term_id != NO_INT_VALUE ) {
            finiTermForReq( term_id );
    }

    return( HSE_STATUS_SUCCESS );
}

#ifdef USE_WORKER_THREADS
// Data passed between PostQueuedCompletionStatus & GetQueueCompletionStatus
// for worker threads
// req_type - request type
// key - unused
// LPOVERLAPPED - cast to EXTENSION_CONTROL_BLOCK *
// if LPOVERLAPPED is NULL stop worker thread, if stopping and req_type is
// 1, then delete all terminals also

static void __cdecl workerThread( void * )
/****************************************/
{
    EXTENSION_CONTROL_BLOCK         *pECB;
    DWORD                           req_type;
    ULONG_PTR                       key;
    int                             workerRequests[NUM_REQ_TYPES] = { 0, 0, 0, 0, 0, 0, 0, 0 };

    InterlockedIncrement( &numWorkerThreadsRunning );
#ifdef SERVER_COM
    CoInitializeEx( NULL, COINIT_MULTITHREADED );
#endif
    for( ; ; ) {
            GetQueuedCompletionStatus( workerCompletionPort,
                                       &req_type,
                                       &key,
                                       (LPOVERLAPPED *)&pECB,
                                       INFINITE );
            if( pECB == NULL ) break;
            if( req_type < NUM_REQ_TYPES ) {
                workerRequests[req_type]++;
            }
            doRequest( pECB, (a_req_type)req_type );
            // tell webserver request is done now
            pECB->ServerSupportFunction( pECB->ConnID,
                                         HSE_REQ_DONE_WITH_SESSION,
                                         NULL,
                                         NULL,
                                         NULL );
    }
    termAddDeleteMutex.get(); // use mutex (any mutex) to ensure accurate sums
    for( int i = 0; i < NUM_REQ_TYPES; i++ ) {
            sumRequests[i] += workerRequests[i];
    }
```

```
    termAddDeleteMutex.give();
    if( req_type == 1 ) {
            // wait for all other worker threads to stop
            while( numWorkerThreadsRunning > 1 ) {
                Sleep( 100 );
            }
            // delete all terminals (disconnect)
            for( int i = 0; i < nextAvailTerm; i++ ) {
                deleteTerm( i );
            }
    }
#ifdef SERVER_COM
    CoUninitialize();
#endif
    InterlockedDecrement( &numWorkerThreadsRunning );
}

DWORD WINAPI HttpExtensionProc( EXTENSION_CONTROL_BLOCK *pECB )
/*************************************************************/
{
    a_req_type        req_type    = getReqType( pECB );

    if( req_type == REQ_DELIVERY_SUBMIT || req_type == REQ_INPUT_FORM ) {
            // The Delivery request does not do any database processing (deferred
            // execution) nor do the menu to input form requests,
            // so avoid the thread switch by executing them in this thread
            if( req_type == REQ_INPUT_FORM ) {
                // this is approximate since no mutex is held
                sumRequests[REQ_INPUT_FORM]++;
            }
            return( doRequest( pECB, req_type ) );
    } else {
            // These requests do database processing, use the worker threads
            if( PostQueuedCompletionStatus( workerCompletionPort,
                                            req_type,
                                            0,
                                            (LPOVERLAPPED)pECB ) ) {
                return( HSE_STATUS_PENDING );
            } else {
                sendFatalError( pECB,
                                "PostQueuedCompletionStatus to worker thread failed",
                                NO_INT_VALUE );
                return( HSE_STATUS_SUCCESS );
            }
    }
}
#endif

static void startThreads( void )
/******************************/
{
    int i;

    // start delivery threads
    deliveryCompletionPort = CreateIoCompletionPort( INVALID_HANDLE_VALUE,
                                                     NULL,
                                                     0,
                                                     numDeliveryThreads );
    _ASSERT( deliveryCompletionPort != NULL );
    for( i = 0; i < numDeliveryThreads; i++ ) {
            if( _beginthread( &deliveryThread, 0, NULL ) == -1 ) {
                _ASSERT( FALSE );
            }
    }

    // start worker threads
#ifdef USE_WORKER_THREADS
```

```
    workerCompletionPort = CreateIoCompletionPort( INVALID_HANDLE_VALUE,
                                                   NULL,
                                                   0,
                                                   numWorkerThreads );
    _ASSERT( workerCompletionPort != NULL );
    for( i = 0; i < numWorkerThreads; i++ ) {
            if( _beginthread( &workerThread, 0, NULL ) == -1 ) {
                _ASSERT( FALSE );
            }
    }
#endif
}

static void stopThreads( void )
/*****************************/
{
    int i;

    // stop delivery threads
    for( i = 0; i < numDeliveryThreads; i++ ) {
            PostQueuedCompletionStatus( deliveryCompletionPort,
                                        0,
                                        0,
                                        NULL );
    }
    while( numDeliveryThreadsRunning > 0 ) {
            Sleep( 100 );
    }
    CloseHandle( deliveryCompletionPort );

    // stop worker threads
#ifdef USE_WORKER_THREADS
    // one (and only one) worker thread will disconnect the terminals
    int disconnect = 1;

    for( i = 0; i < numWorkerThreads; i++ ) {
            PostQueuedCompletionStatus( workerCompletionPort,
                                        disconnect,
                                        0,
                                        NULL );
            disconnect = 0;
    }
    while( numWorkerThreadsRunning > 0 ) {
            Sleep( 100 );
    }
    CloseHandle( workerCompletionPort );
#endif
}




/*c:\original\kit\tpccld.sqc*/
// ********************************************************************************
// Copyright (c) 2001-2014 SAP SE or an SAP affiliate company. All rights reserved.
// ********************************************************************************

// based on sample code supplied in the TPC-C Standard Specification
// Revision 5 appendix A.6


/*+=================================================================+
 | Load TPCC tables
 +=================================================================*/
#if defined( UNIX )
```

```
    #define _FILE_OFFSET_BITS 64
    #if !defined( _LARGEFILE_SOURCE )
            #define _LARGEFILE_SOURCE
    #endif
    #if !defined( _LARGEFILE64_SOURCE )
            #define _LARGEFILE64_SOURCE
    #endif
    #define _MAX_PATH PATH_MAX
    #include <errno.h>
    #include "xitoa.h"
#endif

#if defined( _MSC_VER )
    // attempt to avoid crash in LoadWare (was incorrectly calling
    // fprintf with a NULL fp when inserting is 1)
    #pragma optimize("", off)
#endif

#include <time.h>
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include "random.hpp"
#include "utils.hpp"
#include "sqldef.h"
//#include <itoa.h>

#ifndef _unused
#define _unused( x ) ( x = x )
#endif
#define MAXITEMS      100000
#define CUST_PER_DIST 3000
#define DIST_PER_WARE 10
#define ORD_PER_DIST  3000

EXEC SQL INCLUDE SQLCA;

extern long count_ware;

// Include space for null terminator plus quotes
#define _def_null_term( str )  char _nt_ ## str[sizeof(str)+1+2]
#define _nt( str )                  MakeNullTerm( str, _nt_ ## str, sizeof( str ) )

/* Functions */

long            RandomNumber(long lower, long upper);
void        LoadItems();
void        LoadWare();
void        LoadCust();
void        LoadOrd();
void        LoadNewOrd();
void            Stock( long w_id );
void            District( long w_id );
void            Customer( FILE * fpc, FILE * fph, long  d_id, long  w_id );
void            Orders( FILE * fpo, FILE * fpno, FILE * fpol, long d_id, long w_id );
void        New_Orders();
void            MakeAddress( char *str1, char *str2, char *city, char *state, char *zip );
void        Error();
void            Lastname(int num, char *name);
void            InitPermutation();
int             MakeAlphaString( int x, int y, char *str);
int             MakeNumberString(int x, int y, char *str);
int             MakeZipNumberString(int x, int y, char *str);
int             GetPermutation();
void            gettimestamp ( char* timestamp );
static FILE * MakeFile( char * tabname );
```

```
/* Global SQL Variables */
EXEC SQL BEGIN DECLARE SECTION;
   char        timestamp[31];
   long        count_ware;
EXEC SQL END DECLARE SECTION;
   _def_null_term(timestamp);

/* Global Variables */
   int         i;
   int         option_debug = 0;      /* 1 if generating debug output    */
   char *  data_dir = NULL;    // directory for generated data files;
                                                        // NULL implies INSERT directly
   int                 inserting = 1;
   int                 split_data = 0;

   FILE * StockFile = NULL;
   FILE * DistrictFile = NULL;
   a_seed_info SeedInfo;

/*=================================================================+
 |      main()
 +=================================================================*/
int main(
   int             argc,
   char *          argv[] )
{
   char        arg[2];
   char *  usagestr =
            "Usage:\n"
            "\tWarehouses n\n"
            "\t[Debug]\n"
            "\t[Path directory]\n"
            "\t[SplitData]\n"
            "\t[Help]\n";

EXEC SQL WHENEVER SQLERROR GOTO Error_SqlCall;

  count_ware=0;

  for (i=1; i<argc; i++)
  {
  strncpy(arg,argv[i],2);
  arg[0] = toupper(arg[0]);

  switch (arg[0]) {
   case 'W': /* Warehouses */
     if (count_ware)
     {
       printf("Error - Warehouses specified more than once.\n");
       exit(-1);
     }
     if (argc-1>i)
     {
       i++;
       count_ware=atoi(argv[i]);
       if (count_ware<=0)
       {
         printf("Invalid Warehouse Count.\n");
         exit(-1);
       }
     }
     else
     {
       printf("Error - Warehouse count must follow Warehouse keyword\n");
       exit(-1);
     }
```

```
    break;

/******* Generic Args *********************/
  case 'D': /* Debug Option */
    if (option_debug)
    {
      printf("Error - Debug option specified more than once\n");
      exit(-1);
    }
    option_debug=1;
    break;

  case 'P': /* Path to data files */
    if( argc - 1 > i ) {
            data_dir = argv[++i];
            inserting = 0;
    } else {
            printf( "Error - directory must follow Path keyword\n" );
            exit( -1 );
    }
    break;

  case 'H': /* List Args */
    printf(usagestr);
    exit(0);
    break;

  case 'S':
    split_data = 1;
    break;

  default : printf("Error - Unknown Argument (%s)\n",arg);
    printf(usagestr);
    exit(-1);
   }
 }

 if (!(count_ware)) {
   printf("Not enough arguments.\n");
    printf(usagestr);
   exit(-1);
   }

   if( inserting ) {
      db_init( &sqlca );
            EXEC SQL CONNECT "DBA" IDENTIFIED BY "sql";
   }
   sgenrand( 123, &SeedInfo ); // any nonzero value can be used
   /* Initialize timestamp (for date columns) */
   gettimestamp(timestamp);
   printf( "TPCC Data Load Started...\n" );
   if( !inserting ) {
            StockFile = MakeFile( "stock" );
            DistrictFile = MakeFile( "district" );
   }
   LoadItems();
   LoadWare();
   LoadCust();
   LoadOrd();

   if( inserting ) {
            EXEC SQL COMMIT WORK;
            db_fini( &sqlca );
   } else {
            if( StockFile != NULL ) {
               fclose( StockFile );
            }
```

```
            if( DistrictFile != NULL ) {
                fclose( DistrictFile );
            }
    }
    printf( "\n...DATA LOADING COMPLETED SUCCESSFULLY.\n" );
    exit( 0 );
Error_SqlCall:
    Error();
    return 0;
}

static FILE * MakeFile( char * tabname )
/**************************************/
{
    FILE *      fp;
    char        fname[_MAX_PATH+1];

    if( inserting ) {
            return( NULL );
    }
#if defined( UNIX )
    sprintf( fname, "%s/%s.dat", data_dir, tabname );
#else
    sprintf( fname, "%s\\%s.dat", data_dir, tabname );
#endif
    fp = fopen( fname, "wt" );
    if( fp == NULL ) {
            printf( "Unable to open %s\n", fname );
            exit( -1 );
    }
    setvbuf( fp, NULL, _IOFBF, 128*1024L );
    return( fp );
}

static void CloseFile( FILE * fp )
/********************************/
{
    if( fp != NULL ) {
            fclose( fp );
    }
}

static void CheckWrite( char * tabname )
/**************************************/
{
    if( errno != 0 ) {
            int err;
            err = errno;
            printf( "Error %d writing file for %s\n", err, tabname );
            exit( -1 );
    }
}

static char * MakeNullTerm(
    char *      str,
    char *      tmp,
    unsigned            len )
/*************************/
{
    memcpy( tmp, str, len );
    tmp[len] = '\0';
    return( tmp );
}

/*====================================================================+
 | ROUTINE NAME
 |      LoadItems
```

```
 | DESCRIPTION
 |      Loads the Item table
 | ARGUMENTS
 |      none
 +================================================================*/
void LoadItems()
{
    EXEC SQL BEGIN DECLARE SECTION;
        long    i_id;
             long        i_im_id;
        char    i_name[25];
        float   i_price;
        char    i_data[51];
    EXEC SQL END DECLARE SECTION;
        int     idatasiz;
        int     orig[MAXITEMS];
        long    pos;
        int     i;
             FILE *     fp;
             _def_null_term(i_name);
             _def_null_term(i_data);


    EXEC SQL WHENEVER SQLERROR GOTO sqlerr;
    printf("Loading Item \n");

    fp = MakeFile( "item" );

    for (i=0; i<MAXITEMS/10; i++) orig[i]=0;
    for (i=0; i<MAXITEMS/10; i++)
    {
      do
      {
        pos = RandomNumber(0L,MAXITEMS/10);
      } while (orig[pos]);
      orig[pos] = 1;
    }
    for (i_id=1; i_id<=MAXITEMS; i_id++) {

      /* Generate Item Data */
      i_im_id = RandomNumber(1,10000);
      MakeAlphaString( 14, 24, i_name);
      i_price=((float) RandomNumber(100L,10000L))/(float)100.0;
      idatasiz=MakeAlphaString(26,50,i_data);
      if (orig[i_id])
      {
        pos = RandomNumber(0L,idatasiz-8);
        i_data[pos]='o';
        i_data[pos+1]='r';
        i_data[pos+2]='i';
        i_data[pos+3]='g';
        i_data[pos+4]='i';
        i_data[pos+5]='n';
        i_data[pos+6]='a';
        i_data[pos+7]='l';
      }

      if ( option_debug )
         printf( "IID = %ld, Name= %16s, Price = %5.2f\n",
                 i_id, i_name, i_price );
      if( !inserting ) {
             fprintf( fp, "%d,%d,%s,%5.2f,%s\n",
                 i_id, i_im_id, _nt(i_name), i_price, _nt(i_data) );
             CheckWrite( "item" );
      } else {
      EXEC SQL INSERT INTO
        item (i_id, i_im_id, i_name, i_price, i_data)
```

```
      values (:i_id, :i_im_id, :i_name, :i_price, :i_data);
    }
    if ( !(i_id % 100) ) {
            if( inserting ) {
               printf(".");
               EXEC SQL COMMIT WORK;
            }
      if ( !(i_id % 5000) ) printf(" %ld\r",i_id);
    }
  }
  if( inserting ) {
          EXEC SQL COMMIT WORK;
  } else {
          CloseFile( fp );
  }
  printf("Item Done. \n");
  return;
sqlerr:
  Error();
}
/*================================================================+
 | ROUTINE NAME
 |      LoadWare
 | DESCRIPTION
 |      Loads the Warehouse table
 |      Loads Stock, District as Warehouses are created
 | ARGUMENTS
 |      none
 +================================================================*/
void LoadWare()
{
  EXEC SQL BEGIN DECLARE SECTION;
     long    w_id;
     char    w_name[11];
     char    w_street_1[21];
     char    w_street_2[21];
     char    w_city[21];
     char    w_state[3];
     char    w_zip[10];
     float   w_tax;
     float   w_ytd;
  EXEC SQL END DECLARE SECTION;
          FILE *     fp;
     _def_null_term(w_name);
     _def_null_term(w_street_1);
     _def_null_term(w_street_2);
     _def_null_term(w_city);
     _def_null_term(w_state);
     _def_null_term(w_zip);

  EXEC SQL WHENEVER SQLERROR GOTO sqlerr;
  printf("Loading Warehouse \n");

  fp = MakeFile( "warehouse" );

  for (w_id=1L; w_id<=count_ware; w_id++) {

  /* Generate Warehouse Data */
    MakeAlphaString( 6, 10, w_name);
    MakeAddress( w_street_1, w_street_2, w_city, w_state, w_zip );
    w_tax=((float)RandomNumber(10L,20L))/(float)100.0;
    w_ytd=300000.00;

    if ( option_debug )
      printf( "WID = %ld, Name= %16s, Tax = %5.2f\n",
              w_id, w_name, w_tax );
    if( !inserting ) {
```

```
            fprintf( fp, "%d,%s,%s,%s,%s,%s,%5.2f,%f\n",
                        w_id, _nt(w_name),
              _nt(w_street_1), _nt(w_street_2), _nt(w_city), _nt(w_state),
              _nt(w_zip), w_tax, w_ytd);
            CheckWrite( "warehouse" );
      } else {
      EXEC SQL INSERT INTO
        warehouse (w_id, w_name,
                w_street_1, w_street_2, w_city, w_state, w_zip,
                w_tax, w_ytd)
        values (:w_id, :w_name,
              :w_street_1, :w_street_2, :w_city, :w_state,
              :w_zip, :w_tax, :w_ytd);
      }
      /** Make Rows associated with Warehouse **/
      Stock(w_id);
      District(w_id);
    }
    if( inserting ) {
            EXEC SQL COMMIT WORK;
    } else {
            printf( "\n" );
            CloseFile( fp );
    }
    return;
sqlerr:
    Error();
}
/*================================================================+
 | ROUTINE NAME
 |      LoadCust
 | DESCRIPTION
 |      Loads the Customer Table
 | ARGUMENTS
 |      none
 +================================================================*/
void LoadCust()
{
   EXEC SQL BEGIN DECLARE SECTION;
   EXEC SQL END DECLARE SECTION;
   long    w_id;
   long    d_id;
   FILE *   fpc;
   FILE *   fph;
   EXEC SQL WHENEVER SQLERROR GOTO sqlerr;

   fpc = MakeFile( "customer" );
   fph = MakeFile( "history" );

   for (w_id=1L; w_id<=count_ware; w_id++) {
     printf("Loading Customer for WID=%ld\n",w_id);
     for (d_id=1L; d_id<=DIST_PER_WARE; d_id++) {
        Customer(fpc,fph,d_id,w_id);
     }
   }

    if( inserting ) {
            EXEC SQL COMMIT WORK;  /* Just in case */
    } else {
            CloseFile( fpc );
            CloseFile( fph );
    }
   return;
sqlerr:
   Error();
}
/*================================================================+
```

```
 | ROUTINE NAME
 |      LoadOrd
 | DESCRIPTION
 |      Loads the Orders and Order_Line Tables
 | ARGUMENTS
 |      none
 +=====================================================================*/
void LoadOrd()
{
   EXEC SQL BEGIN DECLARE SECTION;
     long  w_id;
     long  d_id;
   EXEC SQL END DECLARE SECTION;
   FILE *   fpo;
   FILE *   fpno;
   FILE *   fpol;

   EXEC SQL WHENEVER SQLERROR GOTO sqlerr;

   fpo = MakeFile( "orders" );
   fpno = MakeFile( "new_order" );
   fpol = MakeFile( "order_line" );

   for (w_id=1L; w_id<=count_ware; w_id++) {
     printf("Loading Orders for W= %ld\n", w_id);
     for (d_id=1L; d_id<=DIST_PER_WARE; d_id++) {
       Orders(fpo,fpno,fpol,d_id, w_id);
     }
   }

    if( inserting ) {
            EXEC SQL COMMIT WORK;  /* Just in case */
    } else {
            CloseFile( fpo );
            CloseFile( fpno );
            CloseFile( fpol );
    }
   return;
sqlerr:
   Error();
}
/*=====================================================================+
 | ROUTINE NAME
 |      Stock
 | DESCRIPTION
 |      Loads the Stock table
 | ARGUMENTS
 |      w_id - warehouse id
 +=====================================================================*/
void Stock( long w_id )
{
    EXEC SQL BEGIN DECLARE SECTION;
       long    s_i_id;
       long    s_w_id;
       long    s_quantity;
       char    s_dist_01[25];
       char    s_dist_02[25];
       char    s_dist_03[25];
       char    s_dist_04[25];
       char    s_dist_05[25];
       char    s_dist_06[25];
       char    s_dist_07[25];
       char    s_dist_08[25];
       char    s_dist_09[25];
       char    s_dist_10[25];
       char    s_data[51];
    EXEC SQL END DECLARE SECTION;
```

```
    int     sdatasiz;
    long    orig[MAXITEMS];
    long    pos;
    int     i;
          FILE *  fp;
    _def_null_term(s_dist_01);
    _def_null_term(s_dist_02);
    _def_null_term(s_dist_03);
    _def_null_term(s_dist_04);
    _def_null_term(s_dist_05);
    _def_null_term(s_dist_06);
    _def_null_term(s_dist_07);
    _def_null_term(s_dist_08);
    _def_null_term(s_dist_09);
    _def_null_term(s_dist_10);
    _def_null_term(s_data);

 EXEC SQL WHENEVER SQLERROR GOTO sqlerr;
 if( inserting ) {
          printf("Loading Stock Wid=%ld\n",w_id);
 } else {
          printf("Loading Stock Wid=%ld\r",w_id);
 }

 fp = StockFile;

 s_w_id=w_id;

 for (i=0; i<MAXITEMS/10; i++) orig[i]=0;
 for (i=0; i<MAXITEMS/10; i++)
 {
   do
   {
     pos=RandomNumber(0L,MAXITEMS);
   } while (orig[pos]);
   orig[pos] = 1;
 }

for (s_i_id=1; s_i_id<=MAXITEMS; s_i_id++) {

   /* Generate Stock Data */
   s_quantity=RandomNumber(10L,100L);
   MakeAlphaString(24,24,s_dist_01);
   MakeAlphaString(24,24,s_dist_02);
   MakeAlphaString(24,24,s_dist_03);
   MakeAlphaString(24,24,s_dist_04);
   MakeAlphaString(24,24,s_dist_05);
   MakeAlphaString(24,24,s_dist_06);
   MakeAlphaString(24,24,s_dist_07);
   MakeAlphaString(24,24,s_dist_08);
   MakeAlphaString(24,24,s_dist_09);
   MakeAlphaString(24,24,s_dist_10);
   sdatasiz=MakeAlphaString(26,50,s_data);
   if (orig[s_i_id])
   {
     pos=RandomNumber(0L,sdatasiz-8);
     s_data[pos]='o';
     s_data[pos+1]='r';
     s_data[pos+2]='i';
     s_data[pos+3]='g';
     s_data[pos+4]='i';
     s_data[pos+5]='n';
     s_data[pos+6]='a';
     s_data[pos+7]='l';
   }

    if( !inserting ) {
```

```
      fprintf( fp, "%d,%d,%d,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%d,%d,%d,%s\n",
                        s_i_id, s_w_id, s_quantity,
                        _nt(s_dist_01), _nt(s_dist_02), _nt(s_dist_03), _nt(s_dist_04), _nt(s_dist_05),
                        _nt(s_dist_06), _nt(s_dist_07), _nt(s_dist_08), _nt(s_dist_09), _nt(s_dist_10),
                        0, 0, 0, _nt(s_data) );
            CheckWrite( "stock" );
     } else {
     EXEC SQL INSERT INTO
       stock (s_i_id, s_w_id, s_quantity,
            s_dist_01, s_dist_02, s_dist_03, s_dist_04, s_dist_05,
            s_dist_06, s_dist_07, s_dist_08, s_dist_09, s_dist_10,
            s_ytd, s_order_cnt, s_remote_cnt, s_data)
       values (:s_i_id, :s_w_id, :s_quantity,
            :s_dist_01, :s_dist_02, :s_dist_03, :s_dist_04, :s_dist_05,
            :s_dist_06, :s_dist_07, :s_dist_08, :s_dist_09, :s_dist_10,
            0, 0, 0, :s_data);
     }
     if ( option_debug )
        printf( "SID = %ld, WID = %ld, Quan = %ld\n",
             s_i_id, s_w_id, s_quantity );
     if ( !(s_i_id % 100) ) {
             if( inserting ) {
                printf(".");
                if ( !(s_i_id % 5000) ) printf(" %ld\r",s_i_id);
                EXEC SQL COMMIT WORK;
             }
     }
   }
   if( inserting ) {
            EXEC SQL COMMIT WORK;  /* Just in case */
            printf(" Stock Done.\n");
   }
   return;
sqlerr:
   Error();
}

/*================================================================+
 | ROUTINE NAME
 |      District
 | DESCRIPTION
 |      Loads the District table
 | ARGUMENTS
 |      w_id - warehouse id
 +================================================================*/
void District( long w_id )
{

   EXEC SQL BEGIN DECLARE SECTION;
      long    d_id;
      long    d_w_id;
      char    d_name[11];
      char    d_street_1[21];
      char    d_street_2[21];
      char    d_city[21];
      char    d_state[3];
      char    d_zip[10];
      float   d_tax;
      float   d_ytd;
      long    d_next_o_id;
   EXEC SQL END DECLARE SECTION;
            FILE *      fp;
      _def_null_term(d_name);
      _def_null_term(d_street_1);
      _def_null_term(d_street_2);
      _def_null_term(d_city);
      _def_null_term(d_state);
```

```
        _def_null_term(d_zip);

    EXEC SQL WHENEVER SQLERROR GOTO sqlerr;

    if( inserting ) {
            printf("Loading District\n");
    }

    fp = DistrictFile;

    d_w_id=w_id;
    d_ytd=30000.0;
    d_next_o_id=3001L;
    for (d_id=1; d_id<=DIST_PER_WARE; d_id++) {

      /* Generate District Data */
      MakeAlphaString(6L,10L,d_name);
      MakeAddress( d_street_1, d_street_2, d_city, d_state, d_zip );
      d_tax=((float)RandomNumber(10L,20L))/(float)100.0;

      if ( option_debug )
         printf( "DID = %ld, WID = %ld, Name = %10s, Tax = %5.2f\n",
               d_id, d_w_id, d_name, d_tax );

      if( !inserting ) {
        fprintf( fp, "%d,%d,%s,%s,%s,%s,%s,%s,%5.2f,%f,%d\n",
                        d_id, d_w_id, _nt(d_name),
              _nt(d_street_1), _nt(d_street_2), _nt(d_city), _nt(d_state), _nt(d_zip),
              d_tax, d_ytd, d_next_o_id );
             CheckWrite( "district" );
      } else {
      EXEC SQL INSERT INTO
        district (d_id, d_w_id, d_name,
                d_street_1, d_street_2, d_city, d_state, d_zip,
                d_tax, d_ytd, d_next_o_id)
        values (:d_id, :d_w_id, :d_name,
               :d_street_1, :d_street_2, :d_city, :d_state, :d_zip,
               :d_tax, :d_ytd, :d_next_o_id);
      }

    }
    if( inserting ) {
            EXEC SQL COMMIT WORK;
    }

    return;
sqlerr:
    Error();
}

/*================================================================+
 | ROUTINE NAME
 |      Customer
 | DESCRIPTION
 |      Loads Customer Table
 |      Also inserts corresponding history record
 | ARGUMENTS
 |      id   - customer id
 |      d_id - district id
 |      w_id - warehouse id
 +================================================================*/
void Customer(
    FILE * fpc,
    FILE * fph,
    long  d_id,
    long  w_id )
{
```

```
EXEC SQL BEGIN DECLARE SECTION;
    long    c_id;
    long    c_d_id;
    long    c_w_id;
    char    c_first[17];
    char    c_middle[3];
    char    c_last[17];
    char    c_street_1[21];
    char    c_street_2[21];
    char    c_city[21];
    char    c_state[3];
    char    c_zip[10];
    char    c_phone[17];
    char    c_credit[3];
    long    c_credit_lim;
    float   c_discount;
    float   c_balance;
    char    c_data[501];
    char    c_data1[251];
    char    c_data2[251];
    float   h_amount;
    char    h_data[25];
EXEC SQL END DECLARE SECTION;
    _def_null_term(c_first);
    _def_null_term(c_middle);
    _def_null_term(c_last);
    _def_null_term(c_street_1);
    _def_null_term(c_street_2);
    _def_null_term(c_city);
    _def_null_term(c_state);
    _def_null_term(c_zip);
    _def_null_term(c_phone);
    _def_null_term(c_credit);
    _def_null_term(c_data);
    _def_null_term(c_data1);
    _def_null_term(c_data2);
    _def_null_term(h_data);

char * sep;
char * d1;
char * d2;

EXEC SQL WHENEVER SQLERROR GOTO sqlerr;

// printf("Loading Customer for DID=%ld, WID=%ld\n",d_id,w_id);

for (c_id=1; c_id<=CUST_PER_DIST; c_id++) {

  /* Generate Customer Data */
  c_d_id=d_id;
  c_w_id=w_id;
  MakeAlphaString( 8, 16, c_first );
  strcpy( c_middle, "OE" );
  if (c_id <= 1000)
    Lastname(c_id-1,c_last);
  else
    Lastname(NURand(&SeedInfo,255,0,999,NU_C_C_LAST_LOAD),c_last);
  MakeAddress( c_street_1, c_street_2, c_city, c_state, c_zip );
  MakeNumberString( 16, 16, c_phone );
  if (RandomNumber(0L,10L) < 10 ) {
          // 90% are GC
    strcpy( c_credit, "GC" );
  } else {
          // 10% are BC
    strcpy( c_credit, "BC" );
  }
```

```
c_credit_lim=50000;
c_discount=((float)RandomNumber(0L,50L))/(float)100.0;
c_balance= -10.0;
if( split_data ) {
        MakeAlphaString(250,250,c_data1);
        MakeAlphaString(50,250,c_data2);
        d1 = _nt(c_data1);
        sep =",";
        d2 = _nt(c_data2);
} else {
        MakeAlphaString(300,500,c_data);
        d1 = _nt(c_data);
        sep = "";
        d2 = "";
}

if( !inserting ) {
  // c_data output at end to avoid listing columns on LOAD TABLE
        fprintf( fpc, "%d,%d,%d,"
                        "%s,%s,%s,"
                        "%s,%s,%s,%s,%s,"
                        "%s,%s,%s,"
                        "%d,%5.2f,%5.2f,"
                        "%f,%d,%d,"
                        "%s%s%s\n",
                        c_id, c_d_id, c_w_id,
            _nt(c_first), _nt(c_middle), _nt(c_last),
            _nt(c_street_1), _nt(c_street_2), _nt(c_city), _nt(c_state), _nt(c_zip),
            _nt(c_phone), _nt(timestamp), _nt(c_credit),
            c_credit_lim, c_discount, c_balance,
            10.0, 1, 0,
                        d1,sep,d2 );
         CheckWrite( "customer" );
} else {
          if( split_data ) {
            EXEC SQL INSERT INTO
            customer (c_id, c_d_id, c_w_id,
                                c_first, c_middle, c_last,
                                c_street_1, c_street_2, c_city, c_state, c_zip,
                                c_phone, c_since, c_credit,
                                c_credit_lim, c_discount, c_balance, c_data1, c_data2,
                                c_ytd_payment, c_payment_cnt, c_delivery_cnt)
            values (:c_id, :c_d_id, :c_w_id,
                                :c_first, :c_middle, :c_last,
                                :c_street_1, :c_street_2, :c_city, :c_state, :c_zip,
                                :c_phone, :timestamp, :c_credit,
                                :c_credit_lim, :c_discount, :c_balance, :c_data1, :c_data2,
                                10.0, 1, 0) ;
          } else {
            EXEC SQL INSERT INTO
            customer (c_id, c_d_id, c_w_id,
                                c_first, c_middle, c_last,
                                c_street_1, c_street_2, c_city, c_state, c_zip,
                                c_phone, c_since, c_credit,
                                c_credit_lim, c_discount, c_balance, c_data,
                                c_ytd_payment, c_payment_cnt, c_delivery_cnt)
            values (:c_id, :c_d_id, :c_w_id,
                                :c_first, :c_middle, :c_last,
                                :c_street_1, :c_street_2, :c_city, :c_state, :c_zip,
                                :c_phone, :timestamp, :c_credit,
                                :c_credit_lim, :c_discount, :c_balance, :c_data,
                                10.0, 1, 0) ;
          }
}

h_amount=10.0;
MakeAlphaString(12,24,h_data);
```

```
      if( !inserting ) {
              fprintf( fph, "%d,%d,%d,%d,%d,%s,%f,%s\n",
                          c_id, c_d_id, c_w_id,
               c_d_id, c_w_id, _nt(timestamp), h_amount, _nt(h_data));
              CheckWrite( "history" );
      } else {
      EXEC SQL INSERT INTO
          history (h_c_id, h_c_d_id, h_c_w_id,
                   h_d_id, h_w_id, h_date, h_amount, h_data)
          values (:c_id, :c_d_id, :c_w_id,
                  :c_d_id, :c_w_id, :timestamp, :h_amount, :h_data);
      }

      if ( option_debug )
          printf( "CID = %ld, LST = %s, P# = %s\n",
                  c_id, c_last, c_phone );
      if ( !(c_id % 100) ) {
         if( inserting ) {
                  EXEC SQL COMMIT WORK;
                  printf(".");
                  if ( !(c_id % 1000) ) printf(" %ld\r",c_id);
                }
      }
    }
    // printf("Customer Done.\n");

  return;
sqlerr:
  Error();
}

/*+====================================================================+
 | ROUTINE NAME
 |      Orders
 | DESCRIPTION
 |      Loads the Orders table
 |      Also loads the Order_Line table on the fly
 | ARGUMENTS
 |      w_id - warehouse id
 +====================================================================*/
void Orders(
   FILE * fpo,
   FILE * fpno,
   FILE * fpol,
   long d_id,
   long w_id )
{

    EXEC SQL BEGIN DECLARE SECTION;
        long    o_id;
        long    o_c_id;
        long    o_d_id;
        long    o_w_id;
        long    o_carrier_id;
        long    o_ol_cnt;
        long    ol;
        long    ol_i_id;
        long    ol_supply_w_id;
        long    ol_quantity;
        long    ol_amount;
        char    ol_dist_info[25];
    EXEC SQL END DECLARE SECTION;
        _def_null_term(ol_dist_info);

    EXEC SQL WHENEVER SQLERROR GOTO sqlerr;

    // printf("Loading Orders for D=%ld, W= %ld\n", d_id, w_id);
```

```
o_d_id=d_id;
o_w_id=w_id;
InitPermutation();          /* initialize permutation of customer numbers */
for (o_id=1; o_id<=ORD_PER_DIST; o_id++) {

  /* Generate Order Data */
  o_c_id=GetPermutation();
  o_carrier_id=RandomNumber(1L,10L);
  o_ol_cnt=RandomNumber(5L,15L);

  if (o_id > 2100)        /* the last 900 orders have not been delivered) */
  {
        if( !inserting ) {
          fprintf( fpo, "%d,%d,%d,%d,%s,,%d,%d\n",
                    o_id, o_d_id, o_w_id, o_c_id,
            _nt(timestamp), o_ol_cnt, 1);
          CheckWrite( "orders" );
          fprintf( fpno, "%d,%d,%d\n",
                    o_id, o_d_id, o_w_id);
          CheckWrite( "new_order" );
        } else {
    EXEC SQL INSERT INTO
      orders (o_id, o_d_id, o_w_id, o_c_id,
            o_entry_d, o_carrier_id, o_ol_cnt, o_all_local)
      values (:o_id, :o_d_id, :o_w_id, :o_c_id,
          :timestamp, NULL, :o_ol_cnt, 1);
    EXEC SQL INSERT INTO
      new_order (no_o_id, no_d_id, no_w_id)
      values (:o_id, :o_d_id, :o_w_id);
        }
  } else {
    if( !inserting ) {
          fprintf( fpo, "%d,%d,%d,%d,%s,%d,%d,%d\n",
                    o_id, o_d_id, o_w_id, o_c_id,
            _nt(timestamp), o_carrier_id, o_ol_cnt, 1);
          CheckWrite( "orders" );
        } else {
    EXEC SQL INSERT INTO
      orders (o_id, o_d_id, o_w_id, o_c_id,
            o_entry_d, o_carrier_id, o_ol_cnt, o_all_local)
      values (:o_id, :o_d_id, :o_w_id, :o_c_id,
            :timestamp, :o_carrier_id, :o_ol_cnt, 1);
        }
  }


  if ( option_debug )
     printf( "OID = %ld, CID = %ld, DID = %ld, WID = %ld\n",
          o_id, o_c_id, o_d_id, o_w_id);

  for (ol=1; ol<=o_ol_cnt; ol++) {
  /* Generate Order Line Data */
    ol_i_id=RandomNumber(1L,MAXITEMS);
    ol_supply_w_id=o_w_id;
    ol_quantity=5;
    ol_amount=0;

    MakeAlphaString(24,24,ol_dist_info);

    if (o_id > 2100) {
          if( !inserting ) {
            ol_amount=(long)(((float)RandomNumber(10L, 10000L))/(float)100.0);
            // ol_delivery_d moved to avoid listing columns on LOAD TABLE
            fprintf( fpol, "%d,%d,%d,%d,%d,%d,,"
                      "%d,%d,%s\n",
                      o_id, o_d_id, o_w_id, ol,
              ol_i_id, ol_supply_w_id, /* ol_delivery_d, */
```

```
                        ol_quantity, ol_amount, _nt(ol_dist_info) );
                CheckWrite( "order_line" );
              } else {
          EXEC SQL INSERT INTO
            order_line (ol_o_id, ol_d_id, ol_w_id, ol_number,
                  ol_i_id, ol_supply_w_id, ol_quantity, ol_amount,
                  ol_dist_info, ol_delivery_d)
            values (:o_id, :o_d_id, :o_w_id, :ol,
                 :ol_i_id, :ol_supply_w_id, :ol_quantity, :ol_amount,
                 :ol_dist_info, NULL);
              }
        } else {
              // FIXME:  use better date/time
              if( !inserting ) {
                fprintf( fpol, "%d,%d,%d,%d,%d,%d,%s,"
                        "%d,%d,%s\n",
                        o_id, o_d_id, o_w_id, ol,
                  ol_i_id, ol_supply_w_id, _nt(timestamp),
                        ol_quantity, ol_amount, _nt(ol_dist_info) );
                CheckWrite( "order_line" );
              } else {
          EXEC SQL INSERT INTO
            order_line (ol_o_id, ol_d_id, ol_w_id, ol_number,
                  ol_i_id, ol_supply_w_id, ol_quantity, ol_amount,
                  ol_dist_info, ol_delivery_d)
            values (:o_id, :o_d_id, :o_w_id, :ol,
                 :ol_i_id, :ol_supply_w_id, :ol_quantity, :ol_amount,
                 :ol_dist_info, :timestamp);
              }
            }

        if ( option_debug )
          printf( "OL = %ld, IID = %ld, QUAN = %ld, AMT = %8.2f\n",
                  ol, ol_i_id, ol_quantity, ol_amount);

      }
      if ( !(o_id % 100) ) {
        if( inserting ) {
                printf(".");
                EXEC SQL COMMIT WORK;
                if ( !(o_id % 1000) ) printf(" %ld\r",o_id);
              }
      }
    }
    if( inserting ) {
            EXEC SQL COMMIT WORK;
    }

    // printf("Orders Done.\n");
    return;
sqlerr:
    Error();
}

/*+=================================================================+
 | ROUTINE NAME
 |      MakeAddress()
 | DESCRIPTION
 |      Build an Address
 | ARGUMENTS
 +=================================================================*/
void MakeAddress(
     char *str1,
     char *str2,
     char *city,
     char *state,
     char *zip )
```

```
{
  MakeAlphaString(10,20,str1); /* Street 1*/
  MakeAlphaString(10,20,str2); /* Street 2*/
  MakeAlphaString(10,20,city); /* City */
  MakeAlphaString(2,2,state);  /* State */
  MakeZipNumberString(9,9,zip);   /* Zip */
}

/*=================================================================+
 | ROUTINE NAME
 |      Error()
 | DESCRIPTION
 |      Handles an error from a SQL call.
 | ARGUMENTS
 +=================================================================*/
void Error()
{
   printf( "SQL Error %d\n", sqlca.sqlcode);

   EXEC SQL WHENEVER SQLERROR CONTINUE;
   EXEC SQL ROLLBACK WORK;

   exit( -1 );
}

/*=================================================================+
 | ROUTINE NAME
 |      Lastname
 | DESCRIPTION
 |      TPC-C Lastname Function.
 | ARGUMENTS
 |      num  - non-uniform random number
 |      name - last name string
 +=================================================================*/
void Lastname(
  int num,
  char *name )
{
  static char *n[] =
    {"BAR", "OUGHT", "ABLE", "PRI", "PRES",
     "ESE", "ANTI", "CALLY", "ATION", "EING"};

  strcpy(name,n[num/100]);
  strcat(name,n[(num/10)%10]);
  strcat(name,n[num%10]);

 return;
}

static int Permutation[ORD_PER_DIST+1];
static int LastUsed;

//============================================================================
//
// Function : InitPermutation
//
//============================================================================
void InitPermutation()
{
   int i, r, t;
   for (i=1;i<=ORD_PER_DIST;i++)
      Permutation[i] = i;
   for (i=1;i<=ORD_PER_DIST;i++)
   {
      r = RandomNumber(i,ORD_PER_DIST);
      t = Permutation[i];
      Permutation[i] = Permutation[r];
```

```
        Permutation[r] = t;
    }
    LastUsed = 0;
}

//========================================================================
//
// Function : GetPermutation
//
//========================================================================
int GetPermutation()
{
    return( Permutation[++LastUsed] );
}

void gettimestamp ( char* timestamp )
{
    struct tm when;
    time_t now;
    time( &now );
    when = *localtime( &now );
    mktime( &when );
// odbc datetime format
    strftime( timestamp , 30 , "%Y-%m-%d %H:%M:%S.000", &when );
    return;
}

//========================================================================
//
// Function name: MakeAlphaString
//
//========================================================================
//philipdu 08/13/96 Changed MakeAlphaString to use A-Z, a-z, and 0-9 in
//accordance with spec see below:
//The spec says:
//4.3.2.2 The notation random a-string [x .. y]
//(respectively, n-string [x .. y]) represents a string of random alphanumeric
//(respectively, numeric) characters of a random length of minimum x, maximum y,
//and mean (y+x)/2. Alphanumerics are A..Z, a..z, and 0..9. The only other
//requirement is that the character set used "must be able to represent a minimum
//of 128 different characters". We are using 8-bit chars, so this is a non issue.
//It is completely unreasonable to stuff non-printing chars into the text fields.
//-CLevine 08/13/96
//str must have size y+1 for room for null char (Ian McHardy, March 22, 2006)
//
// Do not blank-pad the string (Ivan Bowman, Mar 2008)
int MakeAlphaString( int x, int y, char *str)
{
    int len;
    int i;
    static char chArray[] =
        "0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz";
    static int chArrayMax = 61;
#ifdef DEBUG
    printf("[%ld]DBG: Entering MakeAlphaString()\n", (int) GetCurrentThreadId());
#endif
    len= RandomNumber(x, y);
    for (i=0; i<len; i++)
        str[i] = chArray[RandomNumber(0, chArrayMax)];
    if ( len < y )
        memset(str+len, '\0', y - len);
    str[y] = '\0';
    return len;
}


//========================================================================
```

```
//
// Function name: MakeNumberString
//
//=========================================================================
// sizeof( str ) must be at least 17
int MakeNumberString(int x, int y, char *str)
{
    char tmp[16];
    _unused( x );
    _unused( y );
//MakeNumberString is always called MakeNumberString(16, 16, 16, string)
    memset(str, '0', 16);
    str[16] = '\0';
#if defined( UNIX )
    i32toa(RandomNumber(0, 99999999), tmp, 10);
#else
    itoa(RandomNumber(0, 99999999), tmp, 10);
#endif
    memcpy(str, tmp, strlen(tmp));
#if defined( UNIX )
    i32toa(RandomNumber(0, 99999999), tmp, 10);
#else
    itoa(RandomNumber(0, 99999999), tmp, 10);
#endif
    memcpy(str+8, tmp, strlen(tmp));
    return 16;
}


//=========================================================================
//
// Function name: MakeZipNumberString
//
//=========================================================================
int MakeZipNumberString(int x, int y, char *str)
{
    char tmp[16];
    _unused( x );
    _unused( y );
//MakeZipNumberString is always called MakeZipNumberString(9, 9, 9, string)
    strcpy(str, "000011111");
#if defined( UNIX )
    i32toa(RandomNumber(0, 9999), tmp, 10);
#else
    itoa(RandomNumber(0, 9999), tmp, 10);
#endif
    memcpy(str, tmp, strlen(tmp));
    return 9;
}

/*************************************************************************
 * *
 * drand - returns a double pseudo random number between 0.0 and 1.0. *
 * See irand. *
 *************************************************************************/
double drand()
{
    return( (double)genrand( &SeedInfo ) / 2147483647.0);
}


//=========================================================================
// Function : RandomNumber
//
// Description:
//=========================================================================
long RandomNumber(long lower, long upper)
```

```
{
   return( (long)Random( &SeedInfo, lower, upper ) );
}



/*c:\original\kit\txn.hpp*/
// ********************************************************************************
// Copyright (c) 2001-2014 SAP SE or an SAP affiliate company. All rights reserved.
// ********************************************************************************

// Define base transaction class.
// Both odbc.cpp and comclient.cpp implement ITxn.

#ifndef TXN_HPP
#define TXN_HPP

#include "utils.hpp"
#if defined( UNIX )
#include "unixodbc.h"  // for SQLLEN
#else
#include "ntodbc.h"  // for SQLLEN
#endif

// String length constants
#define SERVER_NAME_LEN                 20
#define DATABASE_NAME_LEN               20
#define USER_NAME_LEN                   20
#define PASSWORD_LEN                    20
#define TABLE_NAME_LEN                  20
#define I_DATA_LEN                      50
#define I_NAME_LEN                      24
#define BRAND_LEN                       1
#define LAST_LEN                16
#define W_NAME_LEN                      10
#define ADDRESS_LEN                     20
#define STATE_LEN               2
#define ZIP_LEN                         9
#define S_DIST_LEN                      24
#define S_DATA_LEN                      50
#define D_NAME_LEN                      10
#define FIRST_LEN               16
#define MIDDLE_LEN                      2
#define PHONE_LEN                       16
#define DATETIME_LEN                    30
#define CREDIT_LEN                      2
#define C_DATA_LEN                      250
#define H_DATA_LEN                      24
#define DIST_INFO_LEN                   24
#define MAX_OL_ITEMS                    15
#define STATUS_LEN                      25
#define OL_DIST_INFO_LEN        24

#ifndef ODBCVER
struct TIMESTAMP_STRUCT {
   short /* SQLSMALLINT */          year;
   unsigned short /* SQLUSMALLINT */ month;
   unsigned short /* SQLUSMALLINT */ day;
   unsigned short /* SQLUSMALLINT */ hour;
   unsigned short /* SQLUSMALLINT */ minute;
   unsigned short /* SQLUSMALLINT */ second;
   unsigned int /* SQLUINTEGER */   fraction;
};
#endif
```

```
// possible values for exec_status_code after transaction completes
enum EXEC_STATUS
{
   eOK,                    // 0 "Transaction commited."
   eInvalidItem,           // 1 "Item number is not valid."
   eDeliveryFailed         // 2 "Delivery Post Failed."
};

typedef unsigned short      a_w_id;
typedef unsigned char       a_d_id;
typedef unsigned long       a_c_id;
typedef unsigned long       a_o_id;
typedef unsigned char       a_ol_id;
typedef unsigned long       a_i_id;
typedef unsigned short      a_qty;
typedef unsigned short      a_carrier_id;
typedef unsigned short      a_threshold;
typedef unsigned short      a_quantity;
typedef TIMESTAMP_STRUCT a_date;
typedef SQLLEN              an_indicator;


struct a_new_order {
   // in
   a_w_id          w_id;
   a_d_id          d_id;
   a_c_id          c_id;
   a_ol_id         o_ol_cnt;
   a_bool          o_all_local;
   // out
   EXEC_STATUS             exec_status_code;
   char            c_last[LAST_LEN+1];
   char            c_credit[CREDIT_LEN+1];
   double          c_discount;
   double          w_tax;
   double          d_tax;
   a_o_id          o_id;
   a_bool          o_commit_flag;
   a_date          o_entry_d;
   double          total_amount;
   struct {
         // in
         a_w_id            ol_supply_w_id;
         a_i_id            ol_i_id;
         a_quantity        ol_quantity;
         // out
         short             ol_li_no;
         char              ol_i_name[I_NAME_LEN+1];
         char              ol_brand_generic[BRAND_LEN+1];
         double            ol_i_price;
         double            ol_amount;
         a_quantity        ol_stock;
   }                       ol[MAX_OL_ITEMS];
};

struct a_payment {
   // in
   a_w_id          w_id;
   a_d_id          d_id;
   a_c_id          c_id;
   a_d_id          c_d_id;
   a_w_id          c_w_id;
   double          h_amount;
   char            c_last[LAST_LEN+1];
   // out
   EXEC_STATUS             exec_status_code;
   a_date          h_date;
```

```
    char            w_street_1[ADDRESS_LEN+1];
    char            w_street_2[ADDRESS_LEN+1];
    char            w_city[ADDRESS_LEN+1];
    char            w_state[STATE_LEN+1];
    char            w_zip[ZIP_LEN+1];
    char            d_street_1[ADDRESS_LEN+1];
    char            d_street_2[ADDRESS_LEN+1];
    char            d_city[ADDRESS_LEN+1];
    char            d_state[STATE_LEN+1];
    char            d_zip[ZIP_LEN+1];
    char            c_first[FIRST_LEN+1];
    char            c_middle[MIDDLE_LEN+1];
    char            c_street_1[ADDRESS_LEN+1];
    char            c_street_2[ADDRESS_LEN+1];
    char            c_city[ADDRESS_LEN+1];
    char            c_state[STATE_LEN+1];
    char            c_zip[ZIP_LEN+1];
    char            c_phone[PHONE_LEN+1];
    a_date          c_since;
    char            c_credit[CREDIT_LEN+1];
    double          c_credit_lim;
    double          c_discount;
    double          c_balance;
    char            c_data[200+1];
};

struct an_order_status {
    // in
    a_w_id          w_id;
    a_d_id          d_id;
    a_c_id          c_id;
    char            c_last[LAST_LEN+1];
    // out
    EXEC_STATUS             exec_status_code;
    char            c_first[FIRST_LEN+1];
    char            c_middle[MIDDLE_LEN+1];
    double          c_balance;
    a_o_id          o_id;
    a_date          o_entry_d;
    a_carrier_id    o_carrier_id;
    an_indicator    ind_carrier_id;
    struct {
            a_i_id              ol_i_id;
            a_w_id              ol_supply_w_id;
            a_quantity          ol_quantity;
            double              ol_amount;
            a_date              ol_delivery_d;
            an_indicator        ind_delivery_d;
    }                           ol[MAX_OL_ITEMS];
    a_ol_id         o_ol_cnt;
};

struct a_delivery {
    // in
    a_w_id          w_id;
    a_carrier_id    o_carrier_id;
    // out
    EXEC_STATUS             exec_status_code;
    a_time          queue_time;
    a_o_id          o_id[10]; // delivered order ids by district
};

struct a_stock_level {
    // in
    a_w_id          w_id;
    a_d_id          d_id;
    a_quantity              threshold;
```

```
    // out
    EXEC_STATUS             exec_status_code;
    a_quantity              low_stock;
};

class ITxn {
public:
    // these functions can throw exceptions of type char *
    virtual void new_order() = 0;
    virtual void payment() = 0;
    virtual void order_status() = 0;
    virtual void delivery() = 0;
    virtual void stock_level() = 0;
    virtual void get_maximum() = 0;       // get maximum w_id in db
    virtual void dump_server_properties( FILE * file = NULL ) { (void)( file );  };
    virtual void dump_database_properties( FILE * file = NULL ) { (void)( file );  };
    virtual ~ITxn() { };
public:
    union {
            a_new_order         new_order;
            a_payment           payment;
            an_order_status     order_status;
            a_deliverydelivery;
            a_stock_level       stock_level;
            a_w_id              max_w_id;
    } _params;
};

// new_Txn throws an exception with type char * if it fails
extern ITxn *       new_Txn( void );
extern void         free_Txn( ITxn * txn );
extern a_bool       init_Txn();
extern void         fini_Txn();

#endif



/*c:\original\kit\utils.cpp*/
// ********************************************************************************
// Copyright (c) 2001-2014 SAP SE or an SAP affiliate company. All rights reserved.
// ********************************************************************************

// implements utility functions

#if defined( UNIX )
#include <sys/stat.h>
#include <sys/types.h>
#else
#include <direct.h>
#endif
#include <stdio.h>
#include "utils.hpp"

// for SetErrorFileName and WriteError
static char errorFileName[200] = "";
static Mutex errorFileMutex;

// Must be in the same order as a_txn_type, and must have the
// same text as the suffix of the ISAPI DLL RequestNames
char *const TxnName[NUM_TXN_TYPES] = {
    "New-Order",
    "Payment",
    "Order-Status",
    "Delivery",
    "Stock-Level"
```

```
};


#ifdef DEBUG
// called by _ASSERT macro
void AssertFail( char *file, int line )
/************************************/
{
   char assert_buf[100];

   sprintf( assert_buf, "assertion failed file %s line %d",
                                 file, line );
   ThrowError( assert_buf );
}
#endif

// convert file_time to a string.  File fill_str with the string representation
// (mm/dd hh:mm:ss.ff format)
void FileTimeToTimeStr( FILETIME *file_time, char *fill_str )
/************************************************************/
{
#if defined( UNIX )
   struct tm           result;
   localtime_r( file_time, &result );

   sprintf( fill_str,
               "%02d/%02d %02d:%02d:%02d",
               result.tm_mon+1,
               result.tm_mday,
               result.tm_hour,
               result.tm_min,
               result.tm_sec );

#else
   SYSTEMTIME         UTC_time;
   SYSTEMTIME         local_time;

   FileTimeToSystemTime( file_time, &UTC_time );
   SystemTimeToTzSpecificLocalTime( NULL, &UTC_time, &local_time );
   sprintf( fill_str,
               "%02d/%02d %02d:%02d:%02d.%02d",
               local_time.wMonth,
               local_time.wDay,
               local_time.wHour,
               local_time.wMinute,
               local_time.wSecond,
               local_time.wMilliseconds / 10 );
#endif
}

// get current time and fill_str with string representation
// (same format as FileTimeToTimeStr)
void GetCurrentTimeStr( char *fill_str )
/**************************************/
{
   FILETIME          now;

#if defined( UNIX )
   time( &now );
#else
   GetSystemTimeAsFileTime( &now );
#endif
   FileTimeToTimeStr( &now, fill_str );
}

// set file name only.  Directory will be TPCC_OUTPUT_DIRECTORY
void SetErrorFileName( char *file_name )
```

```
/*************************************/
{
#if defined( UNIX )
   mkdir( TPCC_OUTPUT_DIRECTORY, 0766 );
#else
   _mkdir( TPCC_OUTPUT_DIRECTORY );
#endif

#if defined( UNIX )
   sprintf( errorFileName, TPCC_OUTPUT_DIRECTORY "/%s", file_name );
#else
   sprintf( errorFileName, TPCC_OUTPUT_DIRECTORY "\\%s", file_name );
#endif
}

// write error to log file
void WriteError( char *context, char *msg )
/*****************************************/
{
   char    time_str[30];
   FILE *  file;

   errorFileMutex.get();
   GetCurrentTimeStr( time_str );
   file = fopen( errorFileName, "a" );
   if( file != NULL ) {
          fprintf( file, "%s (%s):\n%s\n\n",
                        context == NULL ? "" : context,
                        time_str,
                        msg );
          fclose( file );
   }
   errorFileMutex.give();
}

#if !defined( UNIX )
// return TRUE if running on Windows 2000
a_bool IsWindows2000()
/********************/
{
   DWORD dwVersion = GetVersion();
   DWORD dwMajor   = (DWORD)(LOBYTE(LOWORD(dwVersion)));
   DWORD dwMinor   = (DWORD)(HIBYTE(LOWORD(dwVersion)));
   return( dwMajor == 5 && dwMinor == 0 );
}
#endif


// Statistic dumping functions common to ISAPI DLL and RTE
/*****************************************************/

// Fill fill_buff with stat file name of the form
// TPCC_OUTPUT_DIR\<prefix>_mmddhhmm.<ext>
void SetStatFileName( char   *fill_buff,
                      char   *prefix,
                      a_time start_time,
                      char   *ext )
/*************************************/
{
#if defined( UNIX )
   FILETIME          file_time;
   struct tm         result;

   TimeToFileTime( start_time, &file_time );
   localtime_r( &file_time, &result );
   sprintf( fill_buff,
            "%s/%s_%02d%02d%02d%02d.%s",
```

```
                TPCC_OUTPUT_DIRECTORY,
                prefix,
                result.tm_mon+1,
                result.tm_mday,
                result.tm_hour,
                result.tm_min,
                ext );

#else
   FILETIME                 file_time;
   SYSTEMTIME               UTC_time;
   SYSTEMTIME               local_time;

   TimeToFileTime( start_time, &file_time );
   FileTimeToSystemTime( &file_time, &UTC_time );
   SystemTimeToTzSpecificLocalTime( NULL, &UTC_time, &local_time );
   sprintf( fill_buff,
                "%s\\%s_%02d%02d%02d%02d.%s",
                TPCC_OUTPUT_DIRECTORY,
                prefix,
                local_time.wMonth,
                local_time.wDay,
                local_time.wHour,
                local_time.wMinute,
                ext );
#endif
}

// dump delivery stats to TPCC_OUTPUT_DIR\<prefix>_mmddhhmm.dmp
// These stats can be read by tpccstats
void DumpDeliveryStats( char            *file_name_prefix,
                        a_delivery_info *delivery_array,
                        int             num_in_array,
                        a_time                  start_time,
                        a_time                  end_time,
                             int                num_errors,
                        int             last_w_id )
/*******************************************************/
{
   char             file_name[100];
   char             buffer[100];
   FILE *           file;
   a_stat_dat_header        header;

   SetStatFileName( file_name, file_name_prefix, start_time, "dmp" );
   file = fopen( file_name, "wb" );
   if( file == NULL ) {
          sprintf( buffer, "Unable to open %s", file_name );
          ThrowError( buffer );
   }
   header.header_len        = sizeof( header );
   header.data_type= STAT_DELIVERY;
   header.data_len  = sizeof( a_delivery_info );
   header.num_data_elems = num_in_array;
   header.start_time        = start_time;
   header.end_time = end_time;
   header.num_errors        = num_errors;
   header.first_w_id= 1;
   header.last_w_id = last_w_id;
   if( fwrite( &header, sizeof( header ), 1, file ) != 1 ) {
          ThrowError( "Error writing stat file" );
   }
   if( fwrite( delivery_array, sizeof( a_delivery_info ), num_in_array, file )
          != num_in_array ) {
          ThrowError( "Error writing stat file" );
   }
   fclose( file );
```

```
}



/*c:\original\kit\utils.hpp*/
// ************************************************************************
// Copyright 2002-2008 iAnywhere Solutions, Inc.  All rights reserved.
// ************************************************************************

// utility classes and functions

#ifndef UTILS_HPP
#define UTILS_HPP

#include <string.h>
#include "random.hpp"

#if defined( UNIX )
#include <pthread.h>
#include <semaphore.h>
#include "clibext.h"
#else
#include <windows.h>
// disable compiler warnings for deprecated functions
#pragma warning( disable: 4996 )
#endif

#ifdef _ASSERT
   #undef _ASSERT
#endif
#ifdef DEBUG
   #define _ASSERT( cond ) if( !( cond ) ) { AssertFail( __FILE__, __LINE__ ); }
#else
   #define _ASSERT( cond )
#endif
#define _ASSERT_LEN_VALID( buf ) _ASSERT( strlen( buf ) < sizeof( buf ) )

#define _PERCENT( num, denom ) \
   ( (denom) == 0 ? (double)0 : ( (double)(num) / (denom) ) * 100 )

// If the ISAPI DLL generates an error, the returned HTML is prefixed with
// this string
#define HTML_ERROR_PREFIX "<!--Error-->"
#define HTML_ERROR_PREFIX_LEN ( sizeof( HTML_ERROR_PREFIX ) - 1 )

#if defined( UNIX )
   #define __cdecl
   #define _cdecl
   #define FILETIME  time_t
   #include "ifsbase.h"
#else
   typedef unsigned __int64         uint64;
   typedef __int64                  int64;
   typedef unsigned __int32         uint32;
   typedef __int32                  int32;
   typedef unsigned short           uint16;
   typedef short                    int16;
   typedef unsigned char            uint8;
   typedef unsigned char            a_bool;
   #define _i64_const( x ) x##I64
#endif

typedef uint64                      a_time;         // time since Epoch in milliseconds


// transaction types
```

```
/******************/

// transaction types
typedef enum a_txn_type {
    NEW_ORDER,
    PAYMENT,
    ORDER_STATUS,
    DELIVERY,
    STOCK_LEVEL,
    NUM_TXN_TYPES = STOCK_LEVEL + 1
} a_txn_type;

// This global must be in the same order as a_txn_type, and must have the
// same text as the suffix of the ISAPI DLL RequestNames
extern char *const TxnName[NUM_TXN_TYPES];


// random constants and functions
/*******************************/

// C constants for NURand function (see section 2.1.6)
// abs( NU_C_C_LAST_LOAD - NU_C_C_LAST_RUN )
// must be in the range 65-119 excluding 96 and 112
#define NU_C_C_LAST_LOAD    123                 // must be in range 0-255
#define NU_C_C_LAST_RUN      208                // must be in range 0-999

#define NU_C_C_ID    1329   // must be in range 0-3000
#define NU_C_OL_ID               57413  // must be in range 0-100000

// Return integers uniformly distributed in range [i,j].
inline unsigned Random( a_seed_info *seed_info, unsigned i, unsigned j )
/************************************************************************/
{
    return i + (uint32) (((uint64)genrand( seed_info )*(j + 1 - i))
                                    / _i64_const(0x100000000));
}

// Return doubles uniformly distributed in range [0,1).
inline double Random01( a_seed_info *seed_info )
/***********************************************/
{
    return ( (double)genrand( seed_info ) / (double)_i64_const(0x100000000) );

}

// Returns non-uniform random number as defined by section 2.1.6
inline unsigned NURand( a_seed_info *seed_info,
                        unsigned A, unsigned x, unsigned y, unsigned C )
/*****************************************************************/
{
    return (((Random( seed_info, 0, A )|Random( seed_info, x, y )) + C)
              %(y - x + 1)) + x;
}


// thread syncronization classes
/*******************************/
#if defined( UNIX )
class Mutex {
    public:
            Mutex()
            {
                memset( this, 0, sizeof(this));
                pthread_mutex_init( &_impl, NULL );
            }
            ~Mutex()
            {
```

```
            pthread_mutex_destroy( &_impl );
        }
        void get()
        {
            pthread_mutex_lock( &_impl );
        }
        void give()
        {
            pthread_mutex_unlock( &_impl );
        }
    private:
        pthread_mutex_t     _impl;
};
class Semaphore {
    public:
        inline Semaphore( unsigned count = 0 )
        {
            sem_init( &_native_sem, 0, count );
        }
        inline ~Semaphore()
        {
            sem_destroy( &_native_sem );
        }
        inline void post()
        {
            sem_post( &_native_sem );
        }
        inline void wait()
        {
            sem_wait( &_native_sem );
        }
    private:
        sem_t               _native_sem;
};


#else
class Mutex {
public:
    Mutex()
    {
        InitializeCriticalSection( &_impl );
    }
    ~Mutex()
    {
        DeleteCriticalSection( &_impl );
    }
    void get()
    {
        EnterCriticalSection( &_impl );
    }
    void give()
    {
        LeaveCriticalSection( &_impl );
    }
private:
    CRITICAL_SECTION          _impl;
};

class Semaphore {
public:
    Semaphore( unsigned count = 0 )
        : _native_sem( CreateSemaphore( NULL, count, 0x7fffffff, NULL ) )
    {
    }
    ~Semaphore()
    {
```

```
        CloseHandle( _native_sem );
    }
    void post()
    {
        ReleaseSemaphore( _native_sem, 1, NULL );
    }
    void wait()
    {
        WaitForSingleObject( _native_sem, INFINITE );
    }
private:
    HANDLE                  _native_sem;
};
#endif

// Time convertion functions
/*************************/

// convert a_time to seconds
inline double ToSec( a_time t )
/*****************************/
{
#if defined( UNIX )
    return (t/1000.0);
#else
    return ((__int64) t)/10000000.0;
#endif
}

// convert a_time to milliseconds
inline double ToMillisec( a_time t )
/**********************************/
{
#if defined( UNIX )
    return (t/1.0);
#else
    return ((__int64) t)/10000.0;
#endif
}

// convert seconds to a_time
inline a_time ToTime( double t )
/******************************/
{
#if defined( UNIX )
    return (a_time)(t*1000.0);
#else
    return( (a_time)(t*10000000) );
#endif
}

// convert FILETIME to a_time
inline a_time ToTime( FILETIME *ft )
/***********************************/
{
#if defined( UNIX )
    return ( (a_time) ((*ft)*1000) );
#else
    return( *(a_time *)ft );
#endif
}

// convert a_time to FILETIME
inline void TimeToFileTime( a_time t, FILETIME *ft )
/***************************************************/
{
#if defined( UNIX )
```

```
   *ft = (FILETIME)(t/1000);
#else
    *(a_time *)ft = t;
#endif
}

// convert file_time to a string.  File fill_str with the string representation
// (mm/dd hh:mm:ss.fff format)
extern void FileTimeToTimeStr( FILETIME *file_time, char *fill_str );


// get current time and fill_str with string representation
// (same format as FileTimeToTimeStr)
extern void GetCurrentTimeStr( char *fill_str );


// Error and Logging functions
/*****************************/

// throw an error
inline void ThrowError( char *msg )
/*********************************/
{
    throw strdup( msg );
}

// called by _ASSERT
extern void AssertFail( char *file, int line );

#if defined( UNIX )
#define TPCC_OUTPUT_DIRECTORY "tpcc_output_logs"
#else
#define TPCC_OUTPUT_DIRECTORY "c:\\tpcc_output_logs"
#endif

// set file name only.  Directory will be TPCC_OUTPUT_DIRECTORY
extern void SetErrorFileName( char *file_name );

// write error to log file.  SetErrorFileName must be called before first use
extern void WriteError( char *context, char *msg );

// return TRUE if running on Windows 2000
extern a_bool IsWindows2000();


// Statistic dumping structures and functions common to ISAPI DLL and RTE
/*************************************************************************/

// binary statistic data file format information
enum a_stat_dat_type {
    STAT_TXN = 0,   // RTE dump of transactions
    STAT_DELIVERY // SUT dump of delayed deliveries
};

// the first data in each statistic data file
struct a_stat_dat_header {
    int          header_len;          // Length of this structure
    a_stat_dat_type data_type;// Type of stat data
    int          data_len;            // Length of each data structure which follows
                                      // this header
    int          num_data_elems; // Number of data elements following header
    a_time  start_time;               // Approximate start time of run
    a_time  end_time;                 // Approximate end time of run
    int          num_errors;          // Number of errors during run
    int          first_w_id;          // First and Last w_id for terminals included
    int          last_w_id;
};
```

```
// For delivery transactions.
// Used by ISAPI DLL to keep track of deliveries and to dump delivery stats
typedef struct {
    a_time     queued_time;
    a_time     completed_time;
    uint16     w_id;
    uint16                carrier_id;
    uint32     o_id[10];              // delivered order ids by district
} a_delivery_info;


// Fill fill_buff with stat file name of the form
// TPCC_OUTPUT_DIR\<prefix>_mmddhhmm.<ext>
extern void SetStatFileName( char   *fill_buff,
                                    char   *prefix,
                                    a_time start_time,
                                    char   *ext );

// dump delivery stats to TPCC_OUTPUT_DIR\<prefix>_mmddhhmm.dmp
extern void DumpDeliveryStats( char *               file_name_prefix,
                                    a_delivery_info *delivery_array,
                                    int              num_in_array,
                                    a_time                  start_time,
                                    a_time                  end_time,
                                    int              num_errors,
                                    int              last_w_id );

#endif



/*c:\original\kit\acid\acid.py*/
import iatest

def printRow(r):
    out = ''
    for v in range(0,len(r)):
            out += str(r[v]) + ','
    print out

def getRow(r):
    out = []
    for v in range(0,len(r)):
            out.append( str(r[v]) )
    return out

def printQuery( query, _conn ):
    if _conn is None:
            _conn = conn
    crsr = iatest.Cursor( _conn )
    print query
    crsr.execute( query )
    while 1:
            row = crsr.fetch()
            if row is None:
                break
            else:
                printRow( row )
    crsr.close()

def rowsEqual( r1, r2 ):
    if len(r1) != len(r2):
            return False
    for v in range(0,len(r1)):
            if r1[v] != r2[v]:
```

```
                return False
    return True

def getQuery( query, _conn ):
    if _conn is None:
            _conn = conn
    crsr = iatest.Cursor( _conn )
    crsr.execute( query )
    res = []
    while 1:
            row = crsr.fetch()
            print row
            try:
                print row[9]
            except:
                pass
            if row is None:
                break
            else:
                res.append( getRow( row ) )
    return res

def fetch( query, conn ):
    print query
    return conn.fetch( query )

def makeVariables( conn ):
    conn.execute( 'create variable time0 timestamp' )
    conn.execute( 'create variable time1 timestamp' )
    conn.execute( 'create variable time2 timestamp' )
    conn.execute( 'create variable time3 timestamp' )
    conn.execute( 'create variable msg0 long varchar' )
    conn.execute( 'create variable msg1 long varchar' )
    conn.execute( 'create variable msg2 long varchar' )
    conn.execute( 'create variable msg3 long varchar' )
    conn.execute( 'set time0 = current timestamp;' )

def getTimes( conn ):
    times = []
    times.append( str( conn.fetch( 'select time0' )[0] ) )
    times.append( str( conn.fetch( 'select time1' )[0] ) )
    times.append( str( conn.fetch( 'select time2' )[0] ) )
    times.append( str( conn.fetch( 'select time3' )[0] ) )
    return times

def getMessages( conn ):
    msgs = []
    msgs.append( str( conn.fetch( 'select msg0' )[0] ) )
    msgs.append( str( conn.fetch( 'select msg1' )[0] ) )
    msgs.append( str( conn.fetch( 'select msg2' )[0] ) )
    msgs.append( str( conn.fetch( 'select msg3' )[0] ) )
    return msgs

def printTimes(times):
    for (desc,time) in times:
            print desc, ": ", time

def checkTimes(times):
    old = ''
    for (desc,time) in times:
            if time < old:
                return False
            old = time
    return True
```

```
/*c:\original\kit\acid\atom.py*/
import iatest
import acid
import random

c_w_id = random.randint(1,5)
c_d_id = random.randint(1,10)
c_id = random.randint(1,1000)
h_amount = 999.99

conn = iatest.Connection( "uid=dba;pwd=sql" )
acid.makeVariables(conn)

failure = 0

def atomTest( doRollback ):
    if doRollback:
            print "ROLLBACK TEST FOR ATOMICITY"
    else:
            print "COMMIT TEST FOR ATOMICITY"
    print "========================="

    print "Row from customer before transaction"
    print "==================================="

    query = '''
    select c_w_id, c_d_id, c_id, c_payment_cnt, c_ytd_payment, c_balance
    from customer
    where c_w_id = %i
    and   c_d_id = %i
    and   c_id   = %i
    ''' % (c_w_id, c_d_id, c_id)
    print query
    cbrow = conn.fetch(query)
    acid.printRow(cbrow)

    print "Row from district before transaction"
    print"====================================="

    query = '''select d_w_id, d_id, d_ytd
    from district
    where d_w_id = %i
    and   d_id   = %i
    ''' % (c_w_id, c_d_id)
    print query
    dbrow = conn.fetch(query)
    acid.printRow(dbrow)

    print "Row from warehouse before transaction"
    print"======================================="

    query = '''
    select w_id, w_ytd
    from warehouse
    where w_id = %i
    ''' % c_w_id
    print query
    wbrow = conn.fetch(query)
    acid.printRow(wbrow)

    if doRollback:
            print "Execution of payment_byid transaction (ROLLBACK)"
            print "================================================"
            row = conn.fetch( 'call tpcc_payment_rollback( %i, %i, %d, %i, %i, %i )' % (c_w_id, c_w_id, h_amount, c_d_id, c_d_id,
c_id ) )
    else:
```

```
            print "Execution of payment_byid transaction (COMMIT)"
            print "============================================="
            row = conn.fetch( 'call tpcc_payment_commit( %i, %i, %d, %i, %i, %i )' % (c_w_id, c_w_id, h_amount, c_d_id, c_d_id,
c_id ) )
    acid.printRow(row)

    print "Row from customer after transaction"
    print "==================================="

    query = '''
    select c_w_id, c_d_id, c_id, c_payment_cnt, c_ytd_payment, c_balance
    from customer
    where c_w_id = %i
    and   c_d_id = %i
    and   c_id   = %i
    ''' % (c_w_id, c_d_id, c_id)
    print query
    carow = conn.fetch(query)
    acid.printRow(carow)

    print "Row from district after transaction"
    print"====================================="

    query = '''select d_w_id, d_id, d_ytd
    from district
    where d_w_id = %i
    and   d_id   = %i
    ''' % (c_w_id, c_d_id)
    print query
    darow = conn.fetch(query)
    acid.printRow(darow)

    print "Row from warehouse after transaction"
    print"======================================"

    query = '''
    select w_id, w_ytd
    from warehouse
    where w_id = %i
    ''' % c_w_id
    print query
    warow = conn.fetch(query)
    acid.printRow(warow)

    if acid.rowsEqual( cbrow, carow ) and acid.rowsEqual( dbrow, darow ) and acid.rowsEqual( wbrow, warow ):
            if doRollback:
                print 'Test passed'
            else:
                print 'TEST FAILED'
                failure = 1
    else:
            if doRollback:
                print 'TEST FAILED'
                failure = 1
            else:
                print 'Test passed'


atomTest(True)
atomTest(False)

if failure == 0:
    print 'All atomicity tests passed'
else:
    print 'AT LEAST ONE ATOMICITY TEST FAILED!'

conn.close()
```

```
/*c:\original\kit\acid\consist.py*/
import iatest
import acid

conn = iatest.Connection( "uid=dba;pwd=sql" )
failure = False

print "Consistency test: current time:"
acid.printQuery( 'select current timestamp', conn )
print "\n\n\n"

try:
    conn.execute( 'drop table temp_w' )
except:
    pass
conn.execute( 'create table temp_w (t_w_id      smallint)' )

min_w_id = int(conn.fetch('select min(w_id) from warehouse')[0])
max_w_id = int(conn.fetch('select max(w_id) from warehouse')[0])

conn.execute('insert into temp_w values(%i)' % min_w_id )
conn.execute('insert into temp_w values(%i)' % int(0.1*max_w_id) )
conn.execute('insert into temp_w values(%i)' % int(0.2*max_w_id) )
conn.execute('insert into temp_w values(%i)' % int(0.3*max_w_id) )
conn.execute('insert into temp_w values(%i)' % int(0.4*max_w_id) )
conn.execute('insert into temp_w values(%i)' % int(0.5*max_w_id) )
conn.execute('insert into temp_w values(%i)' % int(0.6*max_w_id) )
conn.execute('insert into temp_w values(%i)' % int(0.7*max_w_id) )
conn.execute('insert into temp_w values(%i)' % int(0.8*max_w_id) )
conn.execute('insert into temp_w values(%i)' % int(0.9*max_w_id) )
conn.execute('insert into temp_w values(%i)' % max_w_id )
conn.commit()

print "Warehouses on which consistency check has been run"
print "=================================="
acid.printQuery( 'select * from temp_w', conn )

print "Size checks"
print "================================="
acid.printQuery( 'select count(*) from customer', conn )
acid.printQuery( 'select count(*) from district', conn )
acid.printQuery( 'select count(*) from new_order', conn )
acid.printQuery( 'select count_big(*) from order_line', conn )
acid.printQuery( 'select count(*) from orders', conn )
acid.printQuery( 'select count(*) from warehouse', conn )

print "Results of consistency condition 1"
print "================================="
row = acid.fetch('''select d_w_id, (w_ytd  - sum(d_ytd)) diff
            from warehouse, district
            where d_w_id=w_id
            group by d_w_id, w_ytd
            having  (w_ytd  - sum(d_ytd)) != 0
            ''', conn)
if row is not None:
    print "Failed consistency condition 1"
    failure = True

print "Results of consistency condition 2"
print "================================="
row = acid.fetch('''
   select * from
    (select d_w_id, d_id, (d_next_o_id - 1)  as onext , max(o_id) as omax
```

```
        from district, orders
        where    d_w_id = o_w_id
        and      d_id = o_d_id
              and          d_w_id in (select t_w_id from temp_w)
        group by d_w_id, d_id, onext
        ) dt where dt.onext != dt.omax
              ''', conn)
if row is not None:
    print "Failed consistency condition 2"
    failure = True


row = acid.fetch('''
    select * from
    (select d_w_id, d_id, (d_next_o_id - 1)  as onext , max(no_o_id) as omax
        from district, new_order
        where    d_w_id = no_w_id
        and      d_id = no_d_id
              and          d_w_id in (select t_w_id from temp_w)
        group by d_w_id, d_id, onext
        ) dt where onext != omax
              ''', conn)
if row is not None:
    print "Failed consistency condition 2"
    failure = True


print "Results of consistency condition 3"
print "=================================="
row = acid.fetch('''
    select * from
    (select count(*) as nocount, (max(no_o_id)  - min(no_o_id)  + 1)
                  as total
                  from new_order
                  group by no_w_id, no_d_id
    ) dt where nocount != total
    ''', conn)
if row is not None:
    print "Failed consistency condition 3"
    failure = True


print "Results of consistency condition 4"
print "=================================="
row = acid.fetch('''
    select * from
    (select o_w_id, o_d_id, sum(o_ol_cnt) as ol_sum
                  from orders, temp_w
                  where o_w_id = t_w_id
                  group by o_w_id, o_d_id
    ) consist1,
    (select ol_w_id, ol_d_id, count(*) as ol_count
                  from order_line, temp_w
                  where ol_w_id = t_w_id
                  group by ol_w_id, ol_d_id
    ) consist2
    where o_w_id = ol_w_id
    and      o_d_id = ol_d_id
    and      ol_sum != ol_count
    ''', conn)
if row is not None:
    print "Failed consistency condition 4"
    failure = True

if failure:
    print "AT LEAST ONE CONSISTENCY TEST FAILED"
else:
```

```
    print "All consistency tests passed"



/*c:\original\kit\acid\isol.py*/
/************************************************/

import iatest

import isol1
print "\n\n\n\n"
import isol2
print "\n\n\n\n"
import isol3
print "\n\n\n\n"
import isol4
print "\n\n\n\n"
import isol5
print "\n\n\n\n"
import isol6
print "\n\n\n\n"
import isol7
print "\n\n\n\n"
import isol8
print "\n\n\n\n"
import isol9
print "\n\n\n\n"

if isol1.failure or isol2.failure or isol3.failure \
            or isol4.failure or isol5.failure or isol6.failure \
            or isol7.failure or isol8.failure or isol9.failure:
    print "AT LEAST ONE ISOLATION TEST FAILED"
else:
    print 'All isolation tests passed'




/*c:\original\kit\acid\isol1.py*/
/************************************************/

import iatest
import threading
import time
import acid

failure = False

class Worker(threading.Thread):
    def __init__( self, num ):
            threading.Thread.__init__(self)
            self.num = num
            self.res = None
            self.times = None
    def run( self ):
            conn = iatest.Connection( "uid=dba;pwd=sql" )
            acid.makeVariables( conn )
            if self.num == 0:
                self.res = acid.getQuery( "'call tpcc_neworder_wait_commit( 2, 3, 10, 5, 1,
                                            1000, 2, 1,
                                            2000, 2, 2,
                                            3000, 2, 3,
                                            4000, 2, 4,
                                            5000, 2, 5,
                                            0, 0, 0,
```

```
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0 )'", conn )
            else:
                self.res = acid.getQuery( "'call tpcc_orderstatus_isol( 2, 3, 10 )'", conn )
            self.times = acid.getTimes( conn )


w0 = Worker(0)
w1 = Worker(1)
w0.start()
time.sleep(20)
w1.start()
w0.join()
w1.join()
print w0.res
print w1.res
print w0.times
print w1.times

times = []
times.append( ("T1 started", w0.times[0]) )
times.append( ("T1 paused prior to commit", w0.times[1]) )
times.append( ("T2 started", w1.times[0]) )
times.append( ("T1 allowed to continue", w1.times[2]) )
times.append( ("T2 unblocked", w1.times[2]) )

acid.printTimes(times)
if not acid.checkTimes(times):
    print "Invalid sequence"
    failure = True

print ""
print "T1 Customer last name: " + w0.res[0][3]
print "T2 Customer last name: " + w1.res[0][1]
if w0.res[0][3] != w1.res[0][1]:
    failure = True
print ""
print "T1 Order ID: " + w0.res[0][2]
print "T2 Order ID: " + w1.res[0][0]
if w0.res[0][2] != w1.res[0][0]:
    failure = True
print ""
print "T1 # order lines: " + str(len(w0.res))
print "T2 # order lines: " + str(len(w1.res))
if len(w0.res) != len(w1.res):
    failure = True

print ""
if failure:
    print 'Isolation test 1 failed'
else:
    print 'Isolation test 1 passed'




/*c:\original\kit\acid\isol2.py*/
/**********************************************/
```

```
import iatest
import threading
import time
import acid

failure = False

class Worker(threading.Thread):
    def __init__( self, num ):
            threading.Thread.__init__(self)
            self.num = num
            self.row = None
    def run( self ):
            conn = iatest.Connection( "uid=dba;pwd=sql" )
            acid.makeVariables( conn )
            if self.num == 0:
                self.row = conn.fetch( "'call tpcc_orderstatus( 2, 3, 10 )'" )
                self.row1 = conn.fetch( "'call tpcc_neworder_wait_rollback
                                            ( 2, 3, 10, 5, 1,
                                            1000, 2, 1,
                                            2000, 2, 2,
                                            3000, 2, 3,
                                            4000, 2, 4,
                                            5000, 2, 5,
                                            0, 0, 0,
                                            0, 0, 0,
                                            0, 0, 0,
                                            0, 0, 0,
                                            0, 0, 0,
                                            0, 0, 0,
                                            0, 0, 0,
                                            0, 0, 0,
                                            0, 0, 0,
                                            0, 0, 0 )'" )

            else:
                procname = 'tpcc_orderstatus'
            self.row = conn.fetch( "'call tpcc_orderstatus_isol( 2, 3, 10 ) '")
            self.times = acid.getTimes( conn )

w0 = Worker(0)
w1 = Worker(1)
w0.start()
time.sleep(5)
w1.start()
w0.join()
w1.join()

times = []
times.append( ("T0 started", w0.times[0]) )
times.append( ("T1 paused prior to commit", w0.times[1]) )
times.append( ("T2 started", w1.times[0]) )
times.append( ("T1 allowed to continue", w1.times[2]) )
times.append( ("T2 unblocked", w1.times[2]) )

acid.printTimes(times)
if not acid.checkTimes(times):
    print "Invalid sequence"
    failure = True

print ""
print "T0 data: "
acid.printRow(w0.row)
print "T2 data: "
acid.printRow(w1.row)

if not acid.rowsEqual(w0.row, w1.row):
```

```
    failure = True

print ""
if failure:
    print 'Isolation test 2 failed'
else:
    print 'Isolation test 2 passed'




/*c:\original\kit\acid\isol3.py*/
/*********************************************/

import iatest
import threading
import time
import acid

failure = False

tempconn = iatest.Connection( 'uid=dba;pwd=sql' )
initial_d_next_o_id = int(tempconn.fetch('select d_next_o_id from district where d_id = 3 and d_w_id = 2' )[0])

class Worker(threading.Thread):
    def __init__( self, num ):
            threading.Thread.__init__(self)
            self.num = num
            self.res = None
    def run( self ):
            conn = iatest.Connection( "uid=dba;pwd=sql" )
            acid.makeVariables( conn )
            if self.num == 0:
                procname = 'tpcc_neworder_wait_commit'
            else:
                procname = 'tpcc_neworder_nowait_commit'
            self.res = acid.getQuery( '''call %s( 2, 3, 10, 5, 1,
                                                  1000, 2, 1,
                                                  2000, 2, 2,
                                                  3000, 2, 3,
                                                  4000, 2, 4,
                                                  5000, 2, 5,
                                                  0, 0, 0,
                                                  0, 0, 0,
                                                  0, 0, 0,
                                                  0, 0, 0,
                                                  0, 0, 0,
                                                  0, 0, 0,
                                                  0, 0, 0,
                                                  0, 0, 0,
                                                  0, 0, 0,
                                                  0, 0, 0 )''' % procname, conn )
            self.times = acid.getTimes( conn )

w0 = Worker(0)
w1 = Worker(1)
w0.start()
time.sleep(5)
w1.start()
w0.join()
w1.join()

post_d_next_o_id = int(tempconn.fetch('select d_next_o_id from district where d_id = 3 and d_w_id = 2' )[0])

times = []
times.append( ("T1 started", w0.times[0]) )
```

```
times.append( ("T1 paused prior to commit", w0.times[1]) )
times.append( ("T2 started", w1.times[0]) )
times.append( ("T1 allowed to continue", w0.times[2]) )
times.append( ("T2 finished", w1.times[2]) )

acid.printTimes(times)
if not acid.checkTimes(times):
    print "Invalid sequence"
    failure = True


print "Initial d_next_o_id: " + str(initial_d_next_o_id)
print "Order id of T1: " + w0.res[0][2]
print "Order id of T2: " + w1.res[0][2]
print "Final d_next_o_id: " + str(post_d_next_o_id)

if int(w0.res[0][2]) != initial_d_next_o_id:
    failure = True
if int(w1.res[0][2]) != (initial_d_next_o_id + 1):
    failure = True
if post_d_next_o_id != (initial_d_next_o_id + 2):
    failure = True

print ""
if failure:
    print 'Isolation test 3 failed'
else:
    print 'Isolation test 3 passed'




/*c:\original\kit\acid\isol4.py*/
/************************************************/

import iatest
import threading
import time
import acid

failure = False

tempconn = iatest.Connection( 'uid=dba;pwd=sql' )
initial_d_next_o_id = int(tempconn.fetch('select d_next_o_id from district where d_id = 3 and d_w_id = 2' )[0])

class Worker(threading.Thread):
    def __init__( self, num ):
            threading.Thread.__init__(self)
            self.num = num
            self.res = None
    def run( self ):
            conn = iatest.Connection( "uid=dba;pwd=sql" )
            acid.makeVariables( conn )
            if self.num == 0:
                procname = 'tpcc_neworder_wait_rollback'
            else:
                procname = 'tpcc_neworder_nowait_commit'
            self.res = acid.getQuery( '''call %s( 2, 3, 10, 5, 1,
                                                  1000, 2, 1,
                                                  2000, 2, 2,
                                                  3000, 2, 3,
                                                  4000, 2, 4,
                                                  5000, 2, 5,
                                                  0, 0, 0,
                                                  0, 0, 0,
                                                  0, 0, 0,
```

```
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0 )''' % procname, conn )
            self.times = acid.getTimes( conn )

w0 = Worker(0)
w1 = Worker(1)
w0.start()
time.sleep(5)
w1.start()
w0.join()
w1.join()

post_d_next_o_id = int(tempconn.fetch('select d_next_o_id from district where d_id = 3 and d_w_id = 2' )[0])

times = []
times.append( ("T1 started", w0.times[0]) )
times.append( ("T1 paused prior to commit", w0.times[1]) )
times.append( ("T2 started", w1.times[0]) )
times.append( ("T1 allowed to continue", w0.times[2]) )
times.append( ("T2 finished", w1.times[2]) )

acid.printTimes(times)
if not acid.checkTimes(times):
    print "Invalid sequence"
    failure = True

print "Initial d_next_o_id: " + str(initial_d_next_o_id)
print "Order id of T2: " + w1.res[0][2]
print "Final d_next_o_id: " + str(post_d_next_o_id)

if int(w1.res[0][2]) != initial_d_next_o_id:
    failure = True
if post_d_next_o_id != (initial_d_next_o_id + 1):
    failure = True

print ""
if failure:
    print 'Isolation test 4 failed'
else:
    print 'Isolation test 4 passed'




/*c:\original\kit\acid\isol5.py*/
/***********************************************/

import iatest
import threading
import time
import acid

failure = False

tempconn = iatest.Connection('uid=dba;pwd=sql')
payment = 13.0
o_c_id = int(tempconn.fetch('''
                select first
            o_c_id
            from new_order, orders
            where no_w_id = 2 and no_d_id = 1
```

```
            and no_o_id = o_id and no_d_id = o_d_id and no_w_id = o_w_id
        order by no_w_id asc,no_d_id asc,no_o_id asc ''')[0])

starting_balance = float(tempconn.fetch('''
            select c_balance
            from customer
            where c_id = %i and c_d_id = 1 and c_w_id = 2''' % o_c_id)[0])


class Worker(threading.Thread):
    def __init__( self, num ):
            threading.Thread.__init__(self)
            self.num = num
            self.res = None
    def run( self ):
            conn = iatest.Connection( "uid=dba;pwd=sql" )
            acid.makeVariables( conn )
            if self.num == 0:
                self.res = acid.getQuery( '''call tpcc_delivery_commit( 2, 5 )''', conn )
            else:
                self.res = acid.getQuery( '''call tpcc_payment_commit( 2, 2, %f, 1, 1, %i ) ''' % (payment, o_c_id), conn)
            self.times = acid.getTimes( conn )
            self.messages = acid.getMessages( conn )

w0 = Worker(0)
w1 = Worker(1)
w0.start()
time.sleep(5)
w1.start()
w0.join()
w1.join()

ending_balance = float(tempconn.fetch('''
            select c_balance
            from customer
            where c_id = %i and c_d_id = 1 and c_w_id = 2''' % o_c_id)[0])


times = []
times.append( ("T1 started", w0.times[0]) )
times.append( ("T1 paused prior to commit", w0.times[1]) )
times.append( ("T2 started", w1.times[0]) )
times.append( ("T1 allowed to continue", w0.times[2]) )
times.append( ("T2 finished", w1.times[2]) )

acid.printTimes(times)
if not acid.checkTimes(times):
    print "Invalid sequence"
    failure = True


print w0.messages[0]
print "Using customer: " + str(o_c_id)
print "Starting balance: " + str(starting_balance)
print "Add delivery: " + w0.messages[1]
print "Subtract payment: " + str(payment)
print "Final balance: " + str(ending_balance)

if( starting_balance + float(w0.messages[1]) - payment ) != ending_balance:
    print 'Customer balance incorrect'
    failure = True

print ""
if failure:
    print 'Isolation test 5 failed'
else:
    print 'Isolation test 5 passed'
```

```
/*c:\original\kit\acid\isol6.py*/
/***********************************************/

import iatest
import threading
import time
import acid

failure = False

tempconn = iatest.Connection('uid=dba;pwd=sql')
payment = 9.0
o_c_id = int(tempconn.fetch('''
            select first
        o_c_id
        from new_order, orders
        where no_w_id = 2 and no_d_id = 1
            and no_o_id = o_id and no_d_id = o_d_id and no_w_id = o_w_id
        order by no_w_id asc,no_d_id asc,no_o_id asc ''')[0])

starting_balance = float(tempconn.fetch('''
            select c_balance
            from customer
            where c_id = %i and c_d_id = 1 and c_w_id = 2''' % o_c_id)[0])


class Worker(threading.Thread):
   def __init__( self, num ):
            threading.Thread.__init__(self)
            self.num = num
            self.res = None
   def run( self ):
            conn = iatest.Connection( "uid=dba;pwd=sql" )
            acid.makeVariables( conn )
            if self.num == 0:
               self.res = acid.getQuery( '''call tpcc_delivery_rollback( 2, 5 )''', conn )
            else:
               self.res = acid.getQuery( '''call tpcc_payment_commit( 2, 2, %f, 1, 1, %i ) ''' % (payment, o_c_id), conn)
            self.times = acid.getTimes( conn )
            self.messages = acid.getMessages( conn )

w0 = Worker(0)
w1 = Worker(1)
w0.start()
time.sleep(5)
w1.start()
w0.join()
w1.join()

ending_balance = float(tempconn.fetch('''
            select c_balance
            from customer
            where c_id = %i and c_d_id = 1 and c_w_id = 2''' % o_c_id)[0])

times = []
times.append( ("T1 started", w0.times[0]) )
times.append( ("T1 paused prior to commit", w0.times[1]) )
times.append( ("T2 started", w1.times[0]) )
times.append( ("T1 allowed to continue", w0.times[2]) )
times.append( ("T2 finished", w1.times[2]) )

acid.printTimes(times)
```

```
if not acid.checkTimes(times):
    print "Invalid sequence"
    failure = True

print w0.messages[0]
print "Using customer: " + str(o_c_id)
print "Starting balance: " + str(starting_balance)
print "Subtract payment: " + str(payment)
print "Final balance: " + str(ending_balance)

if( starting_balance - payment ) != ending_balance:
    print 'Customer balance incorrect'
    failure = True

print ""
if failure:
    print 'Isolation test 6 failed'
else:
    print 'Isolation test 6 passed'




/*c:\original\kit\acid\isol7.py*/
/***********************************************/

import iatest
import threading
import time
import acid
import random

failure = False

item1 = random.randint(1000,2000)
item2 = random.randint(1000,2000)


class Worker(threading.Thread):
    def __init__( self, num ):
            threading.Thread.__init__(self)
            self.num = num
            self.res = None
            self.price0 = None
            self.price1 = None
            self.newprice0 = None
            self.newprice1 = None
    def run( self ):
            conn = iatest.Connection( "uid=dba;pwd=sql" )
            conn.execute("set temporary option max_plans_cached = 0" )
            acid.makeVariables( conn )
            if self.num == 0:
                crsr = iatest.Cursor( conn )
                crsr.execute( '"select i_price from item where i_id = %i"' % item1 )
                self.price0 = float(crsr.fetch()[0])
                crsr.execute( '"select i_price from item where i_id = %i"' % item2 )
                self.price1 = float(crsr.fetch()[0])
                conn.commit()
                conn.execute('set time0 = current timestamp')
                time.sleep(10)
                conn.execute( 'set time1 = current timestamp')
                conn.execute( 'update item set i_price = i_price * 1.10 where i_id in (%i)' %(item1) )
                conn.execute( 'update item set i_price = i_price * 1.10 where i_id in (%i)' %(item2) )
                conn.commit()
                conn.execute('set time2 = current timestamp')
                crsr.execute( '"select i_price from item
```

```
                                        where i_id in (%i,%i)
                                        order by i_id"' %(item1, item2) )
            conn.execute('set time3 = current timestamp')
            self.newprice0 = float(crsr.fetch()[0])
            self.newprice1 = float(crsr.fetch()[0])
        else:
            self.res = acid.getQuery( "'call tpcc_neworder_isol7( 2, 3, 10, 3, 1,
                                        %i, 2, 1,
                                        %i, 2, 2,
                                        %i, 2, 3,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0 )"' %(item1,item1,item2), conn )
        self.times = acid.getTimes( conn )


w0 = Worker(0)
w1 = Worker(1)
w0.start()
time.sleep(5)
w1.start()
w0.join()
w1.join()

times = []
times.append( ("T1 committed", w0.times[0]) )
times.append( ("T2 started", w1.times[0]) )
times.append( ("T2 paused after scanning first item (x)", w1.times[1]) )
times.append( ("T3 started",w0.times[1] ) )
times.append( ("Case A: T3 stalled",w0.times[1] ) )
times.append( ("T2 allowed to continue", w1.times[2]) )
times.append( ("T3 finished", w0.times[2]) )

acid.printTimes(times)
if not acid.checkTimes(times):
    print "Invalid sequence"
    failure = True

print "Old prices"
print item1, w0.price0
print item2, w0.price1
print ""
print "New prices"
print item1, w0.newprice0
print item2, w0.newprice1
print ""
print "Prices included in order:"
print item1, w1.res[0][12]
print item1, w1.res[1][12]
print item2, w1.res[2][12]

if float(w0.price0) != float(w1.res[0][12]) \
    or float(w0.price0) != float(w1.res[1][12]) \
    or float(w0.price1) != float(w1.res[2][12]):
    print 'Price does not match original price'
    failure = 1

print ""
```

```
if failure:
   print 'Isolation test 7 failed'
else:
   print 'Isolation test 7 passed'




/*c:\original\kit\acid\isol8.py*/
/***********************************************/

import iatest
import threading
import time
import acid
import random

failure = False

district = random.randint(1,10)
warehouse = random.randint(1,5)

myconn = iatest.Connection('uid=dba;pwd=sql')
myconn.execute( '''update new_order set no_d_id = 255
                        where no_w_id = %i
                        and no_d_id = %i''' %(warehouse,district) )
myconn.commit()


class Worker(threading.Thread):
   def __init__( self, num ):
           threading.Thread.__init__(self)
           self.num = num
           self.res = None
   def run( self ):
           conn = iatest.Connection( "uid=dba;pwd=sql" )
           acid.makeVariables( conn )
           if self.num == 0:
              self.res = acid.getQuery( '''call tpcc_delivery_phantom( %i, 5 )''' % warehouse, conn )
           else:
              self.res = acid.getQuery( '''call tpcc_neworder_nowait_commit( %i, %i, 10, 5, 1,
                                          1000, 2, 1,
                                          2000, 2, 2,
                                          3000, 2, 3,
                                          4000, 2, 4,
                                          5000, 2, 5,
                                          0, 0, 0,
                                          0, 0, 0,
                                          0, 0, 0,
                                          0, 0, 0,
                                          0, 0, 0,
                                          0, 0, 0,
                                          0, 0, 0,
                                          0, 0, 0,
                                          0, 0, 0,
                                          0, 0, 0,
                                          0, 0, 0 )''' %(warehouse,district), conn )
           self.times = acid.getTimes( conn )
           self.messages = acid.getMessages( conn )

try:
   w0 = Worker(0)
   w1 = Worker(1)
   w0.start()
   time.sleep(5)
   w1.start()
   w0.join()
```

```
    w1.join()
except Exception,e:
    print e
#ensure temporarily invalid district ids are replaced before test finishes
myconn.execute( '''update new_order set no_d_id = %i
                        where no_w_id = %i and no_d_id = 255''' % (district,warehouse) )
myconn.commit()

times = []
times.append( ("T1 started", w0.times[0]) )
times.append( ("T1 paused prior to commit", w0.times[1]) )
times.append( ("T2 started", w1.times[0]) )
times.append( ("Case A: T2 stalled",w1.times[0] ) )
times.append( ("T1 allowed to continue", w0.times[2]) )
times.append( ("T2 finished", w1.times[2]) )

acid.printTimes(times)
if not acid.checkTimes(times):
    print "Invalid sequence"
    failure = True


print "Delivery results"
print w0.res
print "New order results"
print w1.res
print w0.messages[0]

if len(w1.res) < 1:
    failure = True

if w0.messages[0].find('No new rows found') != 0:
    failure = True

print ""
if failure:
    print 'Isolation test 8 failed'
else:
    print 'Isolation test 8 passed'




/*c:\original\kit\acid\isol9.py*/
/***********************************************/

import iatest
import threading
import time
import acid

failure = False

class Worker(threading.Thread):
    def __init__( self, num ):
            threading.Thread.__init__(self)
            self.num = num
            self.res = None
    def run( self ):
            conn = iatest.Connection( "uid=dba;pwd=sql" )
            acid.makeVariables( conn )
            if self.num == 0:
                self.res = acid.getQuery( '''call tpcc_orderstatus_phantom( 2, 3, 10 )''', conn )
            else:
                self.res = acid.getQuery( '''call tpcc_neworder_nowait_commit( 2, 3, 10, 5, 1,
                                            1000, 2, 1,
```

```
                                        2000, 2, 2,
                                        3000, 2, 3,
                                        4000, 2, 4,
                                        5000, 2, 5,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0,
                                        0, 0, 0 )'", conn )
        self.times = acid.getTimes( conn )
        self.messages = acid.getMessages( conn )


w0 = Worker(0)
w1 = Worker(1)
w0.start()
time.sleep(5)
w1.start()
w0.join()
w1.join()


times = []
times.append( ("T1 started", w0.times[0]) )
times.append( ("T1 paused prior to commit", w0.times[1]) )
times.append( ("T2 started", w1.times[0]) )
times.append( ("Case A: T2 stalled",w1.times[0] ) )
times.append( ("T1 allowed to continue", w0.times[2]) )
times.append( ("T2 finished", w1.times[2]) )

acid.printTimes(times)
if not acid.checkTimes(times):
    print "Invalid sequence"
    failure = True

print "Order status results"
print w0.res
print "\nNew order results"
print w1.res
print w0.messages[0]
print w0.messages[1]

#compare last ten characters of messages
if w0.messages[0][-10:] != w0.messages[1][-10:]:
    failure = True

print ""
if failure:
    print 'Isolation test 9 failed'
else:
    print 'Isolation test 9 passed'




/*c:\original\kit\acid\procs.py*/
/*********************************************/

import iatest

conn = iatest.Connection('uid=dba;pwd=sql')
```

```
try:
   conn.execute( 'drop procedure tpcc_payment_rollback' )
except:
   pass
proc = str(conn.fetch( """select source from sysprocedure where proc_name = 'tpcc_payment' """ )[0])
proc = proc.replace( 'tpcc_payment', 'tpcc_payment_rollback' )
proc = proc.replace( '/*ACID_DELAY*/commit;', 'set time1 = current timestamp; rollback; set time2 = current timestamp;' )
print '\n' + proc
conn.execute( proc )

try:
   conn.execute( 'drop procedure tpcc_payment_commit' )
except:
   pass
proc = str(conn.fetch( """select source from sysprocedure where proc_name = 'tpcc_payment' """ )[0])
proc = proc.replace( 'tpcc_payment', 'tpcc_payment_commit' )
proc = proc.replace( '/*ACID_DELAY*/commit', 'set time1 = current timestamp; commit; set time2 = current timestamp;' )
print '\n' + proc
conn.execute( proc )

try:
   conn.execute( 'drop procedure tpcc_orderstatus_phantom' )
except:
   pass
proc = str(conn.fetch( """select source from sysprocedure where proc_name = 'tpcc_orderstatus' """ )[0])
proc = proc.replace( 'tpcc_orderstatus', 'tpcc_orderstatus_phantom' )
start = proc.find('/*ACID_START*/')
stop = proc.find('/*ACID_STOP*/')
newproc = proc[:stop] + """\nset time1 = current_timestamp;
set msg0 = 'After 1st select, o_id = ' || @o_id;
waitfor delay '00:00:10';
set time2 = current_timestamp;
""" + proc[start:stop] + """\n
set msg1 = 'After 2nd select, o_id = ' || @o_id;
""" + proc[stop:]
print '\n' + newproc
conn.execute( newproc )

try:
   conn.execute( 'drop procedure tpcc_orderstatus_isol' )
except:
   pass
proc = str(conn.fetch( """select source from sysprocedure where proc_name = 'tpcc_orderstatus' """ )[0])
proc = proc.replace( 'tpcc_orderstatus', 'tpcc_orderstatus_isol' )
start = proc.find('/*ACID_START*/')
stop = proc.find('/*ACID_STOP*/')
newproc = proc[:stop] + "\nset time2 = current timestamp; /*waitfor delay '00:00:10'*/; set time2 = current_timestamp;" +
proc[start:]
newproc = newproc.replace('/*ACID1*/','@o_id, @c_last, @o_entry_d, ')
print '\n' + newproc
conn.execute( newproc )

try:
   conn.execute( 'drop procedure tpcc_delivery_phantom' )
except:
   pass
proc = str(conn.fetch( """select source from sysprocedure where proc_name = 'tpcc_delivery' """ )[0])
proc = proc.replace( 'tpcc_delivery', 'tpcc_delivery_phantom' )
start = proc.find('/*ACID_START*/')
stop = proc.find('/*ACID_STOP*/')
newproc = proc[:stop] + """
if @@sqlstatus != 0 then
set time1 = current timestamp;
waitfor delay '00:00:10';
set time2 = current timestamp;
""" + proc[start:stop] + """
```

```
if @@sqlstatus != 0 then
    set msg0 = 'No new rows found when cursor reopened - warehouse ' || @w_id || ', district ' || @d_id;
    set @d_id = @d_id + 1;
    set @o_id = NULL;
    --commit;
else
    raiserror 23000 'New rows found when cursor reopened';
    return;
end if;
end if;
""" + proc[stop:]
print '\n' + newproc
conn.execute( newproc )

try:
    conn.execute( 'drop procedure tpcc_delivery_commit' )
except:
    pass
proc = str(conn.fetch( """select source from sysprocedure where proc_name = 'tpcc_delivery' """ )[0])
proc = proc.replace( 'tpcc_delivery', 'tpcc_delivery_commit' )
proc = proc.replace( '/*ACID56*/', "if @d_id = 1 then set msg0 = 'Delivered for d_id = 1, c_id = ' || @c_id; set msg1 = @ol_total;
end if;" )
proc = proc.replace( '/*ACID_DELAY*/commit', "\nset time1 = current timestamp; waitfor delay '00:00:10'; set time2 = current
timestamp; commit; set time3 = current timestamp; " )
print '\n' + proc
conn.execute( proc )

try:
    conn.execute( 'drop procedure tpcc_delivery_rollback' )
except:
    pass
proc = str(conn.fetch( """select source from sysprocedure where proc_name = 'tpcc_delivery' """ )[0])
proc = proc.replace( 'tpcc_delivery', 'tpcc_delivery_rollback' )
proc = proc.replace( '/*ACID56*/', "if @d_id = 1 then set msg0 = 'Delivered for d_id = 1, c_id = ' || @c_id; set msg1 = @ol_total;
end if;" )
proc = proc.replace( '/*ACID_DELAY*/commit', "\nset time1 = current timestamp; waitfor delay '00:00:10'; set time2 = current
timestamp; rollback; set time3 = current timestamp;" )
print '\n' + proc
conn.execute( proc )

try:
    conn.execute( 'drop procedure tpcc_neworder_isol7' )
except:
    pass
proc = str(conn.fetch( """select source from sysprocedure where proc_name = 'tpcc_neworder' """ )[0])
proc = proc.replace( 'tpcc_neworder', 'tpcc_neworder_isol7' )
#proc = proc.replace( '/*ACID_DELAY*/commit', "\nset time1 = current timestamp; waitfor delay '00:00:10'; set time2 = current
timestamp; rollback; set time3 = current timestamp;" )
proc = proc.replace( '/*ACID7*/', """
if @@sqlstatus != 0 then
    raiserror 28000 'Deadlock detected';
    return;
end if;
if @li_no = 1 then
    set time1 = current timestamp;
    waitfor delay '00:00:10';
    set time2 = current timestamp;
end if;
""" )
print '\n' + proc
conn.execute( proc )

try:
    conn.execute( 'drop procedure tpcc_neworder_nowait_commit' )
except:
    pass
proc = str(conn.fetch( """select source from sysprocedure where proc_name = 'tpcc_neworder' """ )[0])
```

```
proc = proc.replace( 'tpcc_neworder', 'tpcc_neworder_nowait_commit' )
proc = proc.replace( '/*ACID_DELAY*/commit', "set time1 = current timestamp; commit; set time2 = current timestamp" )
print '\n' + proc
conn.execute( proc )

try:
    conn.execute( 'drop procedure tpcc_neworder_wait_commit' )
except:
    pass
proc = str(conn.fetch( """select source from sysprocedure where proc_name = 'tpcc_neworder' """ )[0])
proc = proc.replace( 'tpcc_neworder', 'tpcc_neworder_wait_commit' )
proc = proc.replace( '/*ACID_DELAY*/commit', "\nset time1 = current timestamp; waitfor delay '00:00:30'; set time2 = current
timestamp; commit; set time3 = current timestamp;" )
print '\n' + proc
conn.execute( proc )

try:
    conn.execute( 'drop procedure tpcc_neworder_wait_rollback' )
except:
    pass
proc = str(conn.fetch( """select source from sysprocedure where proc_name = 'tpcc_neworder' """ )[0])
proc = proc.replace( 'tpcc_neworder', 'tpcc_neworder_wait_rollback' )
proc = proc.replace( '/*ACID_DELAY*/commit', "\nset time1 = current timestamp; waitfor delay '00:00:10'; set time2 = current
timestamp; rollback; set time3 = current timestamp;" )
print '\n' + proc
conn.execute( proc )
```