
# TPC Benchmark C

# Full Disclosure Report

**SQL Anywhere 11**

*on*

**Dell PowerEdge 2950 III**

*with*

**Microsoft Windows 2003 Standard x64 R2 SP2**

FDR Version 1.0

July 28, 2008

# Special notices

Sybase 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. Sybase 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, Sybase 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. Sybase 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.



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

PowerEdge and PowerVault are registered trademarks of Dell Inc.

Microsoft and Windows 2003 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.

| SYBASE iAnywhere | **Dell PowerEdge 2950 III** | **TPCC Version 5.10**<br>**TPC Pricing 1.0.1**<br>Report Date<br>**July 28, 2008** |
|---|---|---|

| Total System Cost | TPCC Throughput | Price/Performance | Availability Date |
|---|---|---|---|
| **$17,540.61 USD** | **20,705 tpmC** | **$0.85 USD/tpmC** | **August 5, 2008** |

| Processors | Database Server | Operating System | Other Software | Number of Users |
|---|---|---|---|---|
| **1/4/4<br>Intel Xeon<br>E5420 2.5GHz** | **SQL Anywhere<br>11** | **Microsoft Windows<br>2003 Standard<br>x64 R2 SP2** | **Microsoft IIS<br>Microsoft COM+** | **16,500** |


| System Components | Quantity | Desc (Server-Client machine) |
|---|---|---|
| Processor | 1 | Intel 2.5GHz Xeon E5420 (4 core, 4 threads total) |
| Memory | 16GB | 8 x 2GB |
| Disk Controller | 1 | PERC 6/E |
| | 1 | PERC 5/i |
| Disk Drives | 2 | 36GB 15Krpm 2.5" SAS |
| | 1 | 73GB 15Krpm 2.5" SAS |
| | 5 | 146GB 10Krpm 2.5" SAS |
| | 15 | 73GB 15Krpm 3.5" SAS |
| Total Storage | 1728.7 GB | |

| SQL Anywhere 11<br>Dell PowerEdge 2950 III | TPC Pricing 1.0.1 | | | TPCC Rev 5.10 Executive Summary | | |
|---|---|---|---|---|---|---|
| **Description** | **Part No.** | **Pricing** | **Unit Price** | **Qty** | **Extended** | **3yr Maint** |
| **Client/Server Hardware** | | | | | | |
| *Dell PowerEdge 2950 III (Quad Xeon E5420)* | 223-4490 | *1* | *1073.00* | *1* | $ 1,073.00 | |
| PERC 6/e 256 | 341-5842 | *1* | 799.00 | 1 | $ 799.00 | |
| PERC 5/i | 341-3067 | *1* | 299.00 | 1 | $ 299.00 | |
| 36GB 15Krpm SAS 2.5in | 341-4726 | *1* | 329.00 | 2 | $ 658.00 | |
| 73GB 15Krpm SAS 2.5in | 341-4727 | *1* | 339.00 | 1 | $ 339.00 | |
| 146GB 10Krpm SAS 2.5in | 341-4732 | *1* | 299.00 | 5 | $ 1,495.00 | |
| 16GB RAM | 311-6199 | *1* | 1088.00 | 1 | $ 1,088.00 | |
| Dell ProSupport, 7x24 HW/SW, 4-hour onsite after diagnosis (3yr) | 986-8392 | *1* | | | | $ 540.00 |
| *Dell PowerVault MD1000* | *222-2299* | *2* | *2480.00* | *1* | $ 2,480.00 | |
| Dual encl mgt module, SAS only | 420-6220 | *2* | *500.00* | *1* | $ 500.00 | |
| SAS Cable | 310-7083 | *2* | 40.00 | 2 | $ 80.00 | |
| 73GB 15Krpm SAS 3.5in | 341-2818 | *2* | 299.00 | 15 | $ 4,485.00 | |
| Dell ProSupport, 7x24 HW/SW, 4-hour onsite after diagnosis (3yr) | 988-3080 | *2* | | | | $ 1,658.00 |
| **Client/Server Software** | | | | | | |
| Microsoft Windows Server 2003 x64 R2 SP2 | 420-7122 | *1* | 799.00 | 1 | $799.00 | |
| O/S Support | 986-8392 | *1* | (incl. in server) | 1 | | |
| Microsoft Visual Studio 2008 Standard | 127-00166 | *3* | 259.99 | 1 | $ 259.99 | |
| SQL Anywhere 11 (1 CPU) | 19866 | *6* | 2499.00 | 1 | $ 2,499.00 | |
| SQL Anywhere Support Incident | 98471 | *7* | 250.00 | 1 | | $ 250.00 |
| SQL Anywhere Bug Confirm Service (3yr) | 97832 | *8* | 250.00 | 1 | | $ 250.00 |
| **Client/Server Accessories** | | | | | | |
| 15" CRT display | CT500G-DT | *4* | 90.99 | 1 | $ 90.99 | |
| Keyboard/mouse | GKM502 | *5* | 24.99 | 1 | $ 24.99 | |
| **Subtotal** | | | | | $ 16,969.97 | $ 2,698.00 |
| **Discount** 16% on Dell hardware based upon total system cost | | | | | -$2,127.36 | |
| **Total** | | | | | **$ 17,540.61** | |

**Pricing Legend**

**1)** Dell, Inc. - Quote 441114806 (see Appendix D) (based on standard 16% discount at this volume)
**2)** Dell, Inc. - Quote 441115981 (see Appendix D) (based on standard 16% discount at this volume)
**3)** http://www.amazon.com/dp/B000WM1Z46?smid=ATVPDKIKX0DER
**4)** http://www.geeks.com/details.asp?invtid=CT500G-DT&cat=MON
**5)** http://accessories.us.dell.com/sna/products/I_O_Devices/productdetail.aspx?c=us&l=en&s=gen&sku=A1459866
**6)** http://eshop.sybase.com/eshop/search/results?id=847&s=Basic&st=Basic&q=catalog&qs=19866
**7)** http://eshop.sybase.com/eshop/search/results?id=847&s=Basic&st=Basic&q=catalog&qs=98471
**8)** http://eshop.sybase.com/eshop/search/results?id=847&s=Basic&st=Basic&q=catalog&qs=98732

Benchmark implementation and results independently audited by
François 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.

| MQTh, Maximum Qualified Throughput | | | | 20705.77 tpmC |
|---|---|---|---|---|
| **Response Times (seconds)** | **Minimum** | **Average** | **90th** | **Maximum** |
| New-Order | 0.11 | 0.32 | 0.58 | 17.11 |
| Payment | 0.11 | 0.17 | 0.28 | 24.77 |
| Order-Status | 0.11 | 0.31 | 0.50 | 19.03 |
| Delivery (interactive portion) | 0.11 | 0.11 | 0.11 | 0.25 |
| Delivery (deferred portion) | 0.05 | 0.43 | 0.80 | 20.27 |
| Stock-Level | 0.11 | 1.42 | 3.12 | 15.95 |
| Menu | 0.11 | 0.11 | 0.11 | 0.98 |
| | | | | |
| **Transaction Mix** | | | **Percent** | **Number** |
| New-Order | | | 44.96% | 2484692 |
| Payment | | | 43.01% | 2376943 |
| Order-Status | | | 4.01% | 221582 |
| Delivery | | | 4.01% | 221643 |
| Stock-Level | | | 4.01% | 221531 |
| | | | | |
| **Keying Times** | | **Minimum** | **Average** | **Maximum** |
| New-Order | | 18.02 | 18.04 | 21.80 |
| Payment | | 3.02 | 3.03 | 6.80 |
| Order-Status | | 2.02 | 2.03 | 5.80 |
| Delivery | | 2.02 | 2.03 | 5.78 |
| Stock-Level | | 2.02 | 2.03 | 5.80 |
| | | | | |
| **Think Times** | | **Minimum** | **Average** | **Maximum** |
| New-Order | | 0.02 | 12.08 | 121.95 |
| Payment | | 0.02 | 12.07 | 120.72 |
| Order-Status | | 0.02 | 10.05 | 101.80 |
| Delivery | | 0.02 | 5.09 | 50.52 |
| Stock-Level | | 0.02 | 5.07 | 50.52 |
| | | | | |
| **Test Duration** | | | | |
| Ramp-up time | | | | 234 min |
| Measurement interval | | | | 120 min |
| Number of checkpoints | | | | 4 |
| Checkpoint interval | | | | 29.5 min |

# Abstract

We describe a TPC-C benchmark run sponsored by Sybase, run using SQL Anywhere 11 on a Dell PowerEdge 2950 III server with Microsoft Windows 2003 Standard x64 R2 SP2. 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 20705 tpmC at a price/performance of $ 0.85 USD/ tpmC. The priced system will available as of August 5, 2008.

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 5.10, released April, 2008.

## TPC Benchmark C Overview

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

*TPC Benchmark C is an On Line Transaction Processing (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 of 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 tpmC results must include the tpmC rate, the associated price-per-tpmC, and the availability date of the priced configuration.*

*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 other environments 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 benchmark 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 Sybase, Inc. The benchmark kit was developed by Sybase iAnywhere, and the benchmark was carried out at Sybase iAnywhere's 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 used a different keyboard, mouse, and video display than the ones included in the priced configuration.

**Figure 1. Priced and benchmarked configuration.**

# 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 36GB 15Krpm SAS | PE 2950 | RAID 1 | C: (O/S partition) | 12.0 GB |
| | | | F: | 21.0 GB |
| 1 x 73GB 15Krpm SAS | PE 2950 | Single disk | D: | 67.7 GB |
| 5 x 146GB 10Krpm SAS | PE 2950 | RAID 0 | M: | 680 GB |
| 15 x 73GB 15Krpm SAS | MD 1000 | RAID 0 | J: | 948 GB |

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

**Table 2. Database file locations.**

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

Two complete backup copies of the database are stored, one in J:\backup and one in M:\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_vhash.sql script, 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 | 59.97% |
| Order Status | Lookup by name | 59.97% |
| 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. Next, the RTE was run for 60 minutes, ensuring that at least one checkpoint occurred and that the throughput in the final five minutes was at least 18000 tpmCs. Then, consist.py was run against the database again.

## 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 log / 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 two backup copies of the entire database (tpcc.db, tpcc.log, cs.dbs, misc.dbs); put one in j:\backup and one in m:\backup.
3. Start the server with start_server.bat
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 3000 tpmC.
6. Remove one of the redundant RAID 1 drives (C:/F:). The system continues to process transactions at the same rate without noticeable interruption.
7. After another 2 minutes, remove a drive from J:. The system reports disk I/O failure and halts.
8. Shut down the database and web server.
9. Replace the removed physical disks with newly formatted disks. The virtual disks they corresponded to will be unreadable.
10. Reboot the machine.

11. Using MegaRAID Storage Manager, choose to make the cleared C/F disk available as the inconsistent portion of the RAID 1 array.

12. Using Dell Array Manager, rebuild the degraded C/F disk.

13. Using Dell Array Manager, create virtual drives spanning the D: and J: disks.

14. In Windows, create and format drives for D: and for J:.

15. Copy the backed up files in M:\backup to their respective locations on D:, J:, and M:. (Do not copy the backed-up log file).

16. Rename f:\tpcc\tpcc.log to f:\tpcc\tpcc2.log

17. Run: dbeng11 tpcc.db –a f:\tpcc\tpcc2.log

18. Start the server with start_server.bat

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

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

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

22. Copy this file 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 rows.

23. Once the test has completed, replace the redundant drive to the RAID 1 array and rebuild it (C:/F:).

## 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 start_server.bat

3. Start the RTE and let it run for about 15 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 6 minutes (verifying that the average throughput is above 19000 tpmC and that at least 110000 new order transactions have completed.
8. Kill the power to the entire SUT by turning off the powerbar to which all SUT components are connected.
9. After 10 seconds, reenable the power and restart the system.
10. Once the machine has rebooted, run start_server.bat to start the database server.
11. Run count_orders_after.bat to count the number of orders in the recovered database.
12. On the RTE machine, run convertorders.py to generate a flat file containing all orders reported as committed to the client.
13. Copy this file 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 rows.

# 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 | 59500000 |
| district | 16500 |
| history | 49500000 |
| item | 100000 |
| new_orders | 14850000 |
| order_line | 495024275 |
| orders | 49500000 |
| stock | 165000000 |
| warehouse | 1650 |

## 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 | 0.32 | 0.58 | 17.11 |
| Payment | 0.17 | 0.28 | 24.77 |
| Order Status | 0.31 | 0.50 | 19.03 |
| Delivery (interactive) | 0.11 | 0.11 | 0.25 |
| Delivery (deferred) | 0.43 | 0.80 | 20.27 |
| Stock Level | 1.42 | 3.12 | 15.95 |
| Menu | 0.11 | 0.11 | 0.98 |

## 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 | 18.02 | 18.04 | 21.80 |
| Payment | 3.02 | 3.03 | 6.80 |
| Order Status | 2.02 | 2.03 | 5.80 |
| Delivery (interactive) | 2.02 | 2.03 | 5.78 |
| Stock Level | 2.02 | 2.03 | 5.80 |

**Table 7. Think times.**

| Type | Minimum | Average | Maximum |
|---|---|---|---|
| New Order | 0.02 | 12.08 | 121.95 |
| Payment | 0.02 | 12.07 | 120.72 |
| Order Status | 0.02 | 10.05 | 101.80 |
| Delivery (interactive) | 0.02 | 5.09 | 50.52 |
| Stock Level | 0.02 | 5.07 | 50.52 |

## Response Time Frequency Distribution

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

**Figure 2. New order response time.**

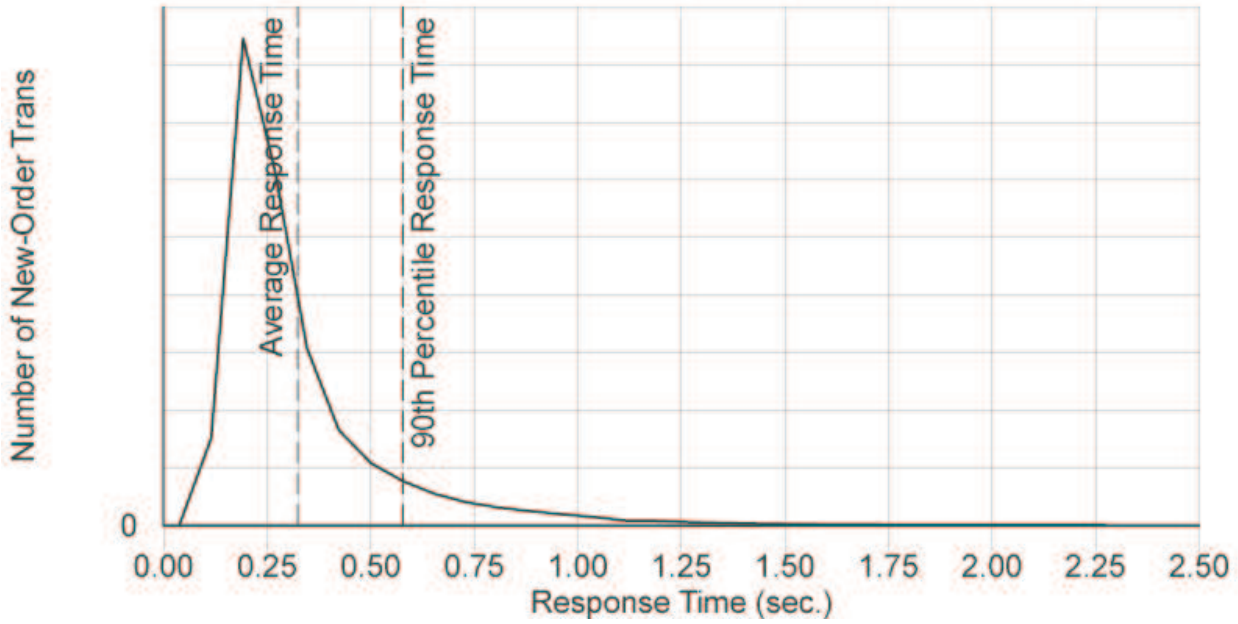



**Figure 3. Payment response time.**

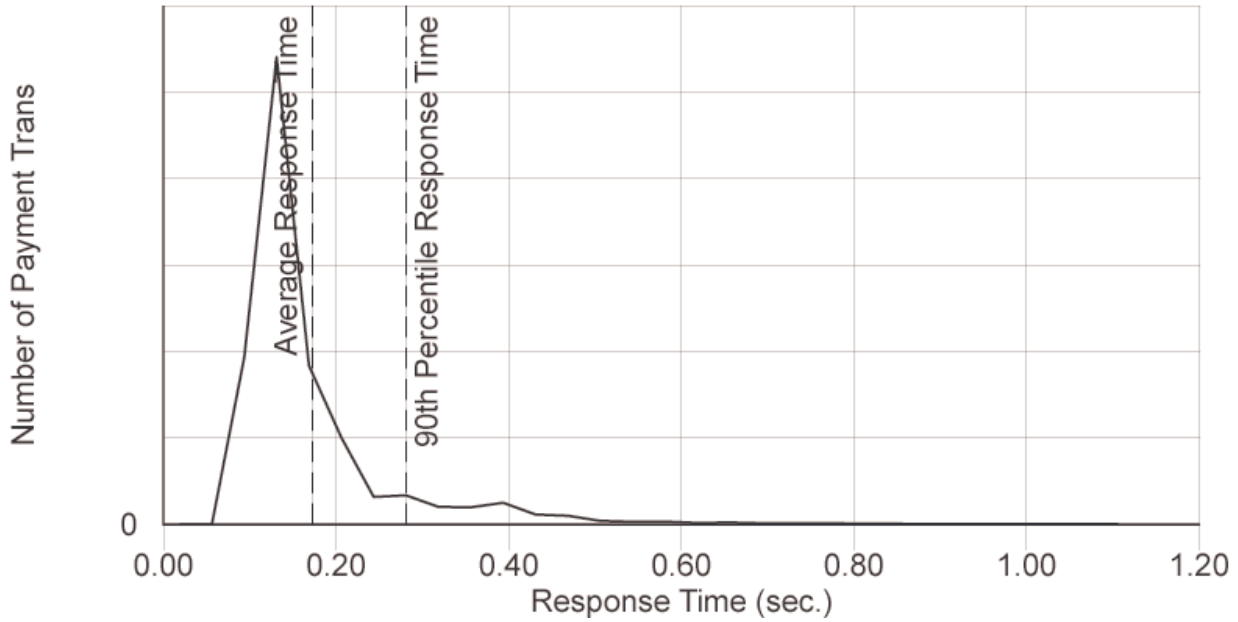


**Figure 4. Order status response time.**

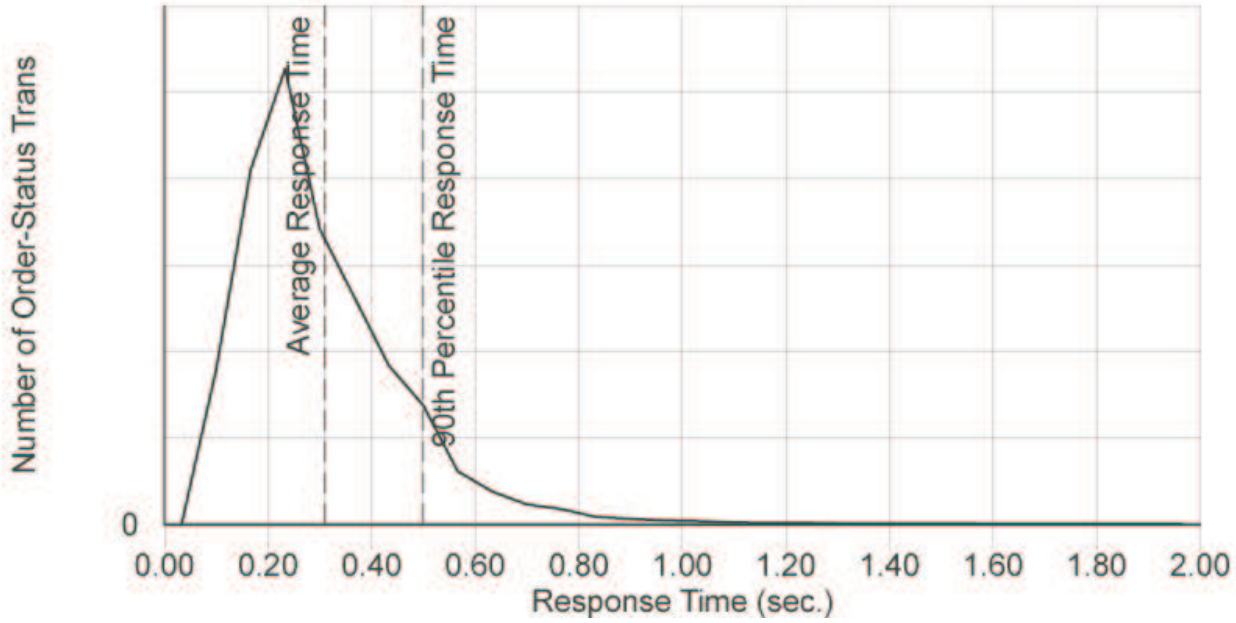



**Figure 5. Interactive delivery response time.**

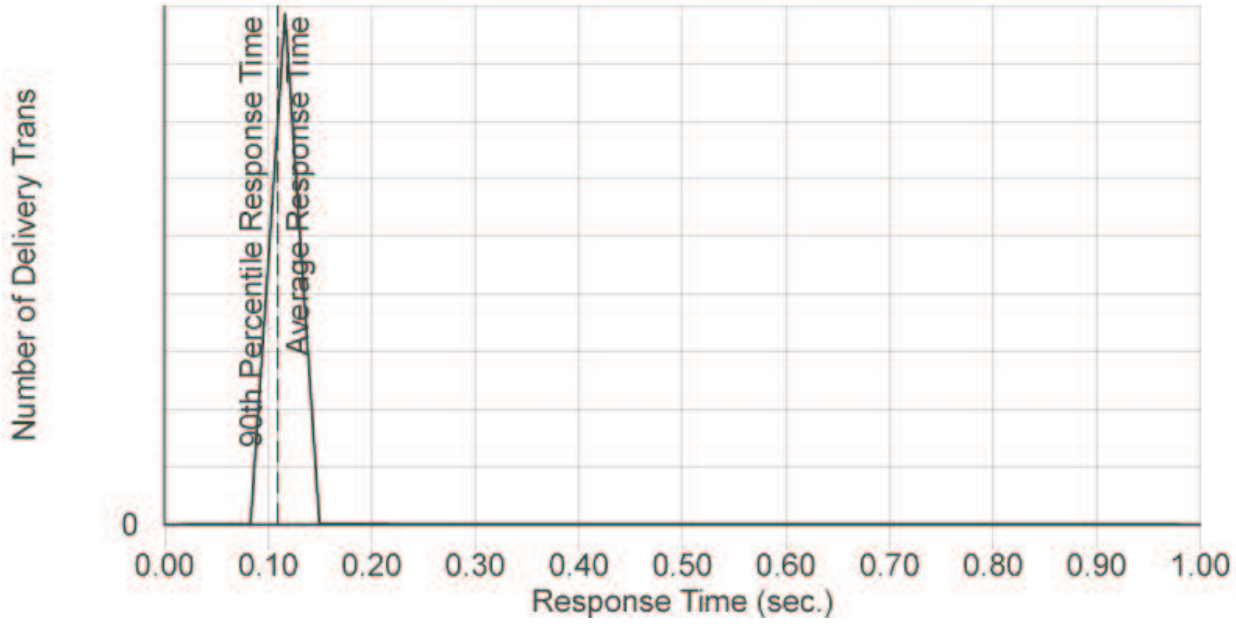


**Figure 6. Deferred delivery response time.**

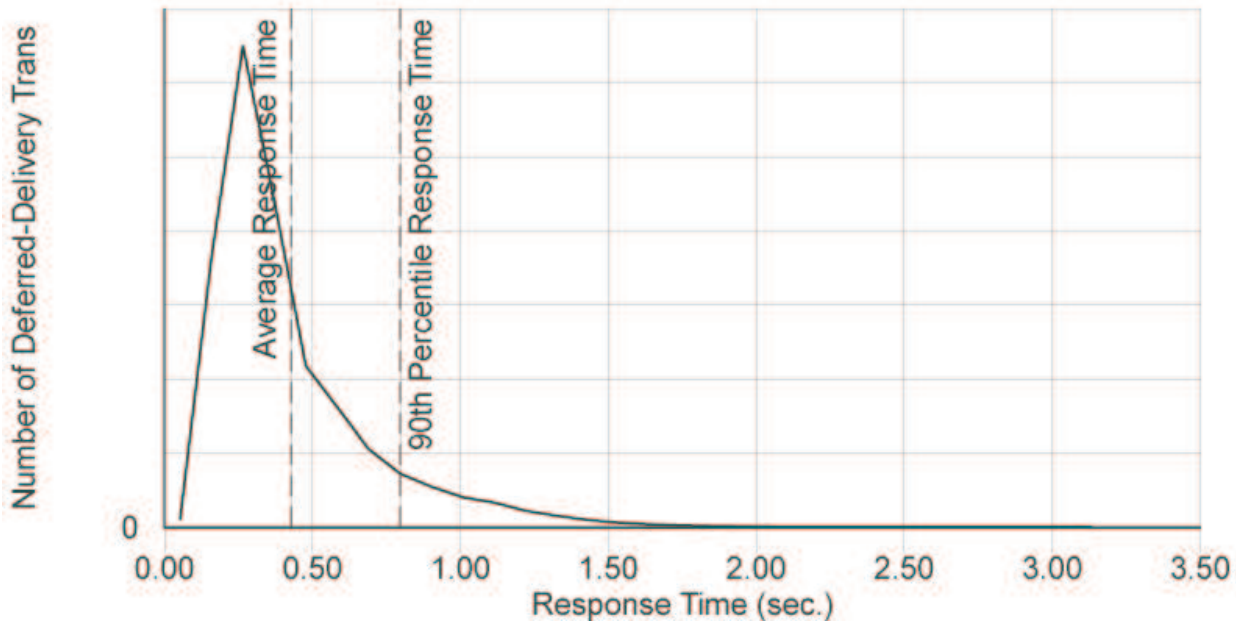
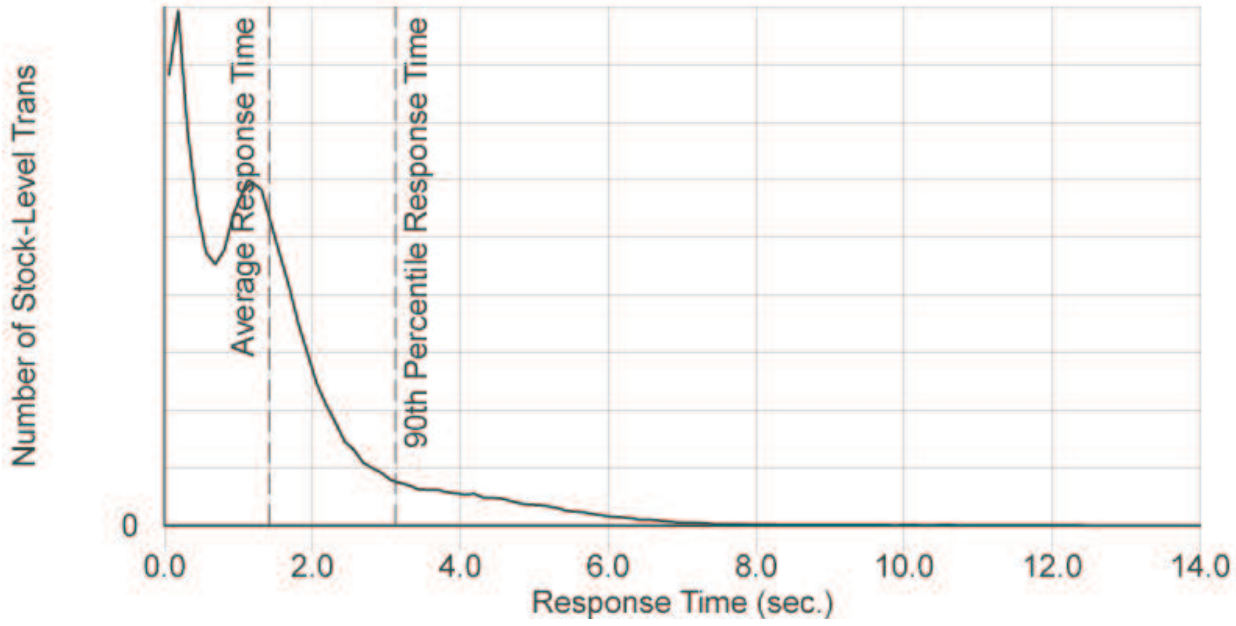


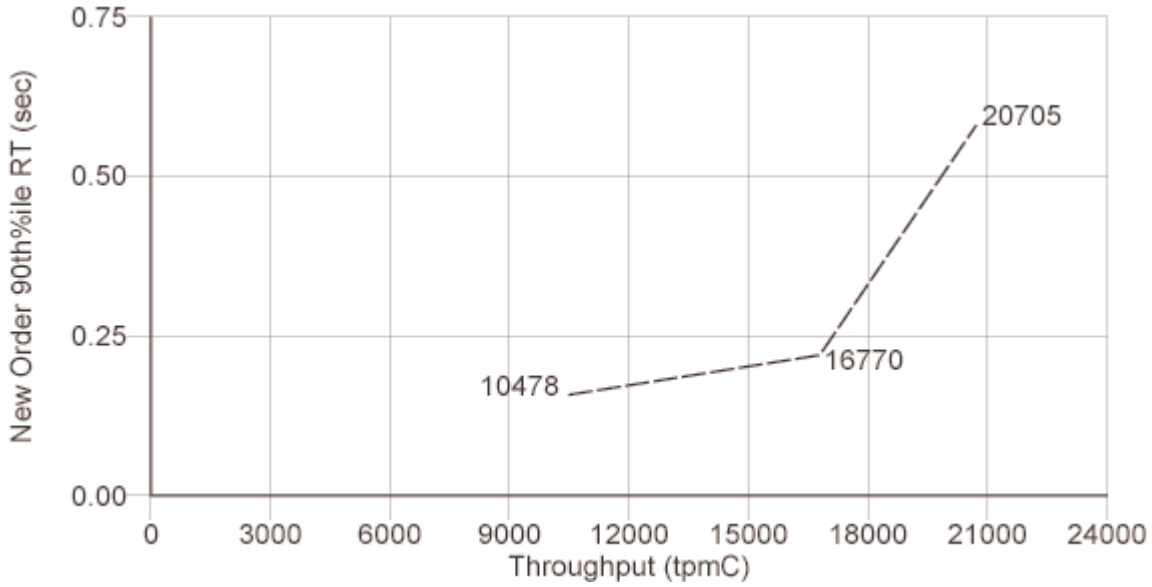

**Figure 7. Stock level response time.**



## 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).*

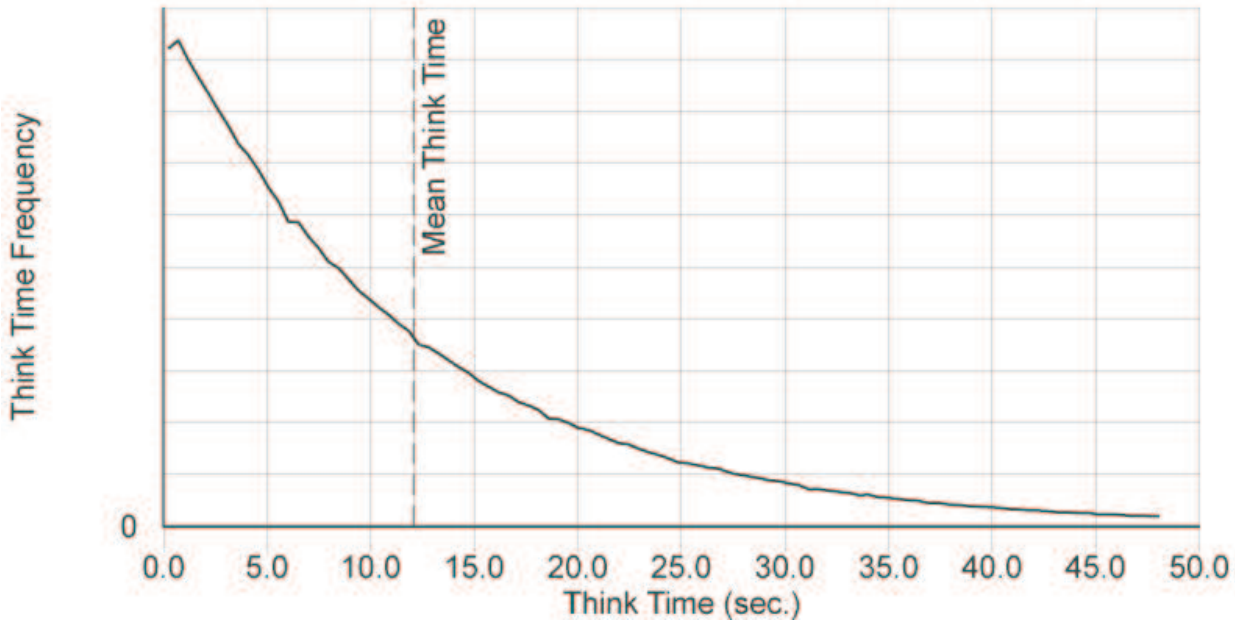
**Figure 8. Response time vs. throughput.**




## Think Time Frequency Distribution

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

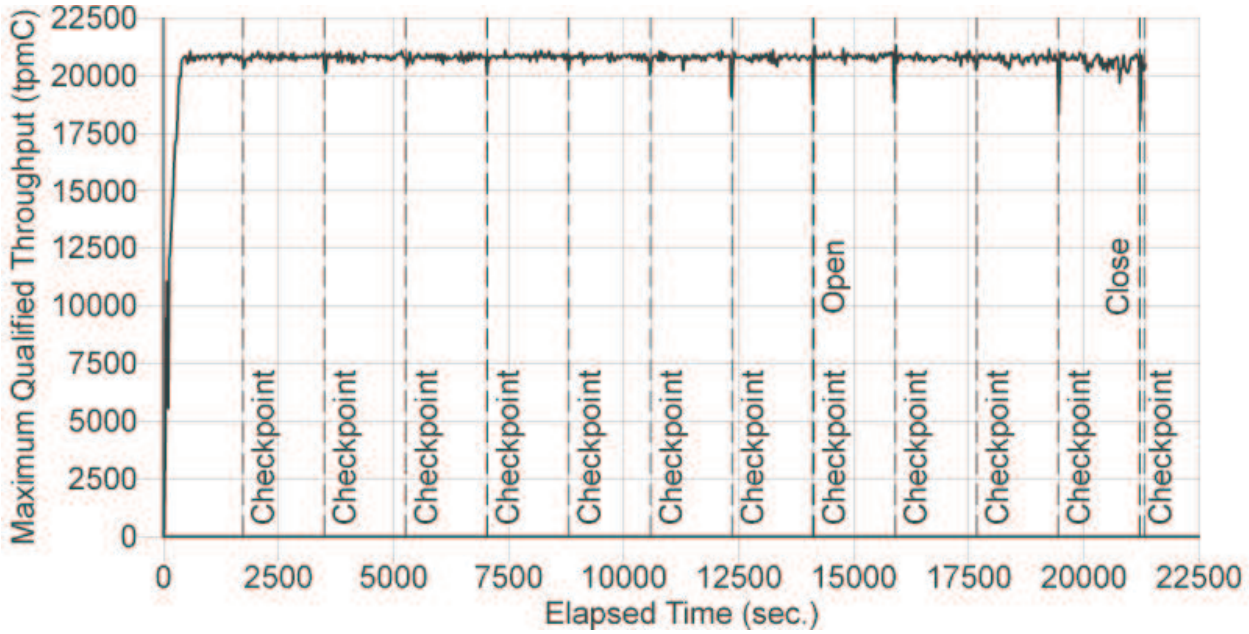
**Figure 9. Think time distribution.**



## Throughput versus Elapsed Time

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

**Figure 10. Throughput over ramp-up and 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 an eight 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 entire eight hour period.

## 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 | 23:46:58 | 00:11:27 | 24 min |
| Checkpoint #2 | 00:16:30 | 00:41:12 | 25 min |
| Checkpoint #3 | 00:46:02 | 01:11:28 | 25 min |
| Checkpoint #4 | 01:15:34 | 01:42:03 | 26 min |
| Measurement Interval | 23:46:40 | 01:46:40 | 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 was run during the benchmark with the following command line:

```
tpccrte.exe –h 192.168.5.5 –r 10000 –n 23000000 –w 1650 –t 600
```

(Connect to the web server, running on newton, displaying a performance metric every 10K transactions, running 23 million transactions for 1650 warehouses using 600 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 RTE was connected to the SUT (client and server on a single machine) by a 1Gbit ethernet crossover cable.

## 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 8 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, and Microsoft Windows 2003 and Visual Studio 2008, are currently available at the time of report publication.

SQL Anywhere 11 will be available on August 5, 2008.

## 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 | 20705 tpmC |
| Total system cost | $17,540.61 USD |
| Price per tpmC | $0.85 USD/ tpmC |
| Availability date | August 5, 2008 |

# 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.

INFO SIZING

TPC TRANSACTION PROCESSING PERFORMANCE COUNCIL CERTIFIED AUDITOR

Benchmark Sponsor: Dan Farrar
SQL Anywhere Query Processing
Sybase iAnywhere
445 Wes Graham Way
Waterloo, ON N2L 6R2
Canada

July 25, 2008

I verified the TPC Benchmark™ C performance of the following Client Server configuration:

Platform: Dell PowerEdge 2950 III
Operating system: Microsoft Windows 2003 Standard x64 R2 SP2
Database Manager: Sybase SQL Anywhere 11
Transaction Manager: Microsoft COM+

The results were:

| CPU's Speed | Memory | Disks | NewOrder 90% Response Time | tpmC |
|---|---|---|---|---|
| **Server: Dell PowerEdge 2950 III** | | | | |
| 1 x Intel Xeon E5420 (2.5GHz) | 16 GB (12 MB L2) | 2 x 36 GB 2.5" SAS<br>1 x 73 GB 2.5" SAS<br>5 x 146 GB 2.5" SAS<br>15 x 73 GB 3.5" SAS | 0.58 Seconds | 20,705.77 |

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:

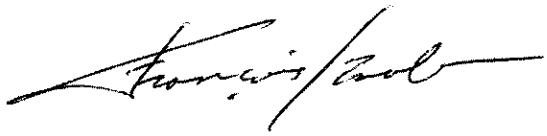
- The transactions were correctly implemented
- The database records were the proper size
- The database was properly scaled and populated
- The ACID properties were met
- Input data was generated according to the specified percentages

- 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 taken 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** | 1650 | **tpmC** | 21000 | **tpmC/W** | 12.73 | |
| **#wh used in test** | 1650 | | | | | |

| **Table** | **Rows** | **Data** | **Index** | **5% Space** | **8H Space** | **Total Space** |
|---|---|---|---|---|---|---|
| 'customer' | 49500000 | 27249736 | 2723890.4 | 599,473 | | 30,573,099 |
| 'district' | 16500 | 1532 | 101.2 | 82 | | 1,715 |
| 'history' | 49500000 | 2318456 | | | 385,614 | 2,704,070 |
| 'item' | 100000 | 8012 | 127.6 | 163 | | 8,302 |
| 'new_order' | 14850000 | 184484 | 6261.2 | | 31,725 | 222,471 |
| 'order_line' | 495024275 | 28402400 | 3102374 | | 5,239,989 | 36,744,763 |
| 'orders' | 49500000 | 1420220 | 1266311.2 | | 446,834 | 3,133,365 |
| 'stock' | 165000000 | 55000100 | 739464 | 1,114,791 | | 56,854,355 |
| 'warehouse' | 1650 | 236 | 4.4 | 12 | | 252 |
| **Totals** | | 114,585,176 | 7,838,534 | 1,714,520 | 6,104,162 | 130,242,392 |

| **Segment No.** | **Segment Name** | **Seg. Size** | **Needed** | **Overhead** | **Not Needed** |
|---|---|---|---|---|---|
| 0 | 'system' | 1052276 | 10000 | 100 | 1,042,276 |
| 15 | 'temporary' | 1048708 | 10000 | 100 | 1,038,708 |
| 1 | 'cs_space' | 87519628 | 84,803,741 | 848,037 | 2,715,887 |
| 2 | 'misc_space' | 50000268 | 46,741,075 | 467,411 | 3,259,193 |
| **Totals** | | 139,620,880 | 131,564,816 | 1,315,648 | 8,056,064 |

| | | |
|---|---|---|
| **Dynamic space** | 32,141,076 | Sum of Data for Order, Order-Line and History (excluding free extents) |
| **Static space** | 93,312,802 | Data + Index + 5% Space + Overhead - Dynamic space |
| **Free space** | 6,110,937 | Total Seg. Size - Dynamic Space - Static Space - Not Needed |
| **Daily growth** | 6,545,092 | (Dynamic space/W * 62.5)* tpmC |
| **Daily spread** | -3,706,700 | Free space - 1.5 * Daily growth (zero if negative) |
| **60 day (KB)** | 486,018,313 | Static space + 60 (daily growth + daily spread) |
| **60 day (GB)** | 463.50 | Excludes OS, Paging and RDBMS Logs |
| configured data dev size (GB) | 1628.00 | |

| | | | | | |
|---|---|---|---|---|---|
| **Log growth** | Experiment: | Number NO | 441,463 | Log size (KB) | 582016 |
| **Log (KB) per N-O txn** | 1.318380023 | | | | |
| **8 Hour Log (GB)** | 12.67 | | | | |
| configured log dev (f:) size (GB) | 21.00 | | | | |

# APPENDIX B: THIRD PARTY PRICING

**DELL**

***QUOTATION***

**QUOTE #: 441114806**

**Customer #: 2067317**

**Quote Date: 7/24/08**

**Date: 7/24/08 3:18:21 PM**

**Customer Name: Sybase**

| TOTAL QUOTE AMOUNT: | $6,684.73 | | |
|---|---|---|---|
| Product Subtotal: | $6,169.84 | | |
| Tax: | $431.89 | | |
| Shipping & Handling: | $83.00 | | |
| Shipping Method: | Ground | Total Number of System Groups: | 1 |

| GROUP: 1 | QUANTITY: 1 | SYSTEM PRICE: $6,169.84 | GROUP TOTAL: $6,169.84 |
|---|---|---|---|
| | | | |
| Base Unit: | | Quad Core Xeon E5420 Processor2x6MB Cache, 2.5GHz, 1333MHz FSB, PE2950 (223-4490) | |
| Processor: | | Information,No Second Processor (311-1193) | |
| Memory: | | 16GB 667MHz (8x2GB), Dual Ranked DIMMs (311-6199) | |
| Video Card: | | LOM NICs are TOE Ready (430-2968) | |
| Video Memory: | | Riser with 3 PCIe Slots for PowerEdge 2950 (320-4607) | |
| Hard Drive: | | Hard Drive MultiSelect Option (465-5674) | |
| Hard Drive Controller: | | PERC 5/i, x8 Backplane Integrated Controller Card (341-3067) | |
| Floppy Disk Drive: | | No Floppy Drive w/Filler Panel (341-3078) | |
| Operating System: | | Windows Server 2003 R2 Standard x64 Edition with SP2 Includes 5 CALs (420-7122) | |
| NIC: | | ONBOARD BROADCOM 5708 1GBE NETWORKING (430-1764) | |
| Modem: | | Dell Remote Access Card, 5th Generation for PowerEdge Remote Management (313-3923) | |
| CD-ROM or DVD-ROM Drive: | | 24X IDE CD-RW/DVD ROM Drive for PowerEdge 2950 (313-3934) | |
| Sound Card: | | Bezel for PE 2950 (313-3920) | |

| | |
|---|---|
| Speakers: | 1x8 Backplane for 2.5-inch Hard Drives, PE2950 III (311-7934) |
| Documentation Diskette: | Electronic Documentation and OpenManage DVD Kit (310-7415) |
| Controller Option: | PERC6E SAS RAID Controller 2x4 Connectors, External, PCIe256MB cache (341-5842) |
| Feature | Integrated SAS/SATA RAID 6, PERC 6/i Integrated (341-5729) |
| Feature | No Rack Rails Included (310-7411) |
| Service: | Dell Hardware Warranty Plus Onsite Service Inital YR (984-1399) |
| Service: | SILVER Enterprise Support: 4 Hour 7x24 Onsite Service Post Problem Diagnosis,2 YR Ext (960-8292) |
| Service: | SILVER Enterprise Support: 7x24 Hardware/Software Tech PhoneSupport, Enterprise Command C enter, 3Yr (960-8562) |
| Service: | SILVER Enterprise Support: 4 Hour 7x24 Onsite Service Post Problem Diagnosis,Init YR (970-4160) |
| Service: | Dell Hardware Warranty, Extended Year(s) (984-1417) |
| Installation: | On-Site Installation Declined (900-9997) |
| Misc: | Non-Redundant Power Supply (310-9895) |
| Misc: | Power Cord, NEMA 5-15P to C14,15 amp, wall plug, 10 feet / 3 meter (310-8509) |
| | 36GB 15K RPM Serial-Attach SCSI 3Gbps 2.5-in HotPlug HardDrive (341-4726) - Quantity 2 |
| | 146G,10K,SAS3G,2.5,HP (341-4732) - Quantity 5 |
| | 73GB 15K RPM Serial-Attach SCSI 3Gbps 2.5-in HotPlug HardDrive (341-4727) |
| | |

| COMMENTS |
|---|
| |
| 16% HARDWARE DISCOUNT |
| |

| | | | |
|---|---|---|---|
| SALES REP: | AMANDA MILLER | PHONE: | 1800-234-9999 |
| Email Address: | Amanda_Miller@Dell.com | Phone Ext: | 7244623 |

**For your convenience, your sales representative, quote number and customer number have been**
**included to provide you with faster service when you are ready to place your order. You may also**
**place your order online at www.dell.com/quote**

**This quote is subject to the terms of the agreement signed by you and Dell, or absent such agreement,**
**is subject to the applicable Dell terms and conditions agreement.**

**Prices and tax rates are valid in the U.S. only and are subject to change.**

***Sales/use tax is a destination charge, i.e. based on the "ship to" address on your purchase**

***order.***
***Please indicate your taxability status on your PO. If exempt, please fax exemption certificate to***
***Dell Tax Department at 800-433-9023, referencing your customer number.***
***If you have any questions regarding tax please call 800-433-9019 or email Tax_Department@dell.com *****

**All product and pricing information is based on latest information available. Subject to change without**
**notice or obligation.**

**LCD panels in Dell products contain mercury, please dispose properly. Please contact Dell's Asset Recovery Services group for EPA compliant disposal options at US_Dell_ARS_Requests@dell.com. Minimum quantities may apply.**

# DELL

## *QUOTATION*

**QUOTE #: 441115981**

**Customer #: 2067317**

**Quote Date: 7/24/08**

**Date: 7/24/08 3:18:32 PM**

**Customer Name: Sybase**

| TOTAL QUOTE AMOUNT: | $8,597.50 | | |
|---|---|---|---|
| Product Subtotal: | $7,995.80 | | |
| Tax: | $559.70 | | |
| Shipping & Handling: | $42.00 | | |
| Shipping Method: | Ground | Total Number of System Groups: | 1 |

| GROUP: 1 | QUANTITY: 1 | SYSTEM PRICE: $7,995.80 | GROUP TOTAL: $7,995.80 |
|---|---|---|---|
| | | | |
| Base Unit: | | PowerVault MD1000, Rack, 3U, 15 Bay External SAS/SATA Storage Array with Locking Bezel (222-2299) | |
| Hard Drive Controller: | | Two Enclosure Management Modules, PowerVault MD1000 SAS/SATA (420-6220) | |
| Cable: | | Cable MultiSelect Option (465-5675) | |
| Controller Option: | | Customer already has a PERC5/EController Card (Required to operate MD1000) (341-3154) | |
| Feature | | No Rails Included (310-7079) | |
| Service: | | Type 2 Contract - Same Day 4-Hour 7x24 Parts and Labor Onsite Response, 2 Year Extended (981-1102) | |
| Service: | | Silver Enterprise Support: Enterprise Command Center, 3 Years (981-6062) | |
| Service: | | Premier Enterprise Silver Support-Com plex Resolutions w/Advanced Software Support-3 Pack-Exp. 3 Y ears (981-6122) | |
| Service: | | Type 2 Contract - Same Day 4-Hour 7x24 Parts and Labor Onsite Response, Initial Year (981-6650) | |
| Service: | | Dell Hardware Warranty Plus Onsite Service Initial Year (985-7579) | |
| Service: | | Dell Hardware Warranty Plus Onsite Service Extended Year(s) (985-9948) | |
| Installation: | | Remote Implementation of a Dell PowerVault MD1xxx Subsystem (to schedule, please call 800-945-DELL or 800-945-3355) (983-7217) | |

| | |
|---|---|
| | 73GB 15K RPM Serial-Attach SCSI 3Gbps 3.5-in HotPlug HardDrive (341-2818) - Quantity 15 |
| | SAS Cable, 2 Meter, MDx000 (310-7083) - Quantity 2 |
| | |

| COMMENTS |
|---|
| |
| 16% HARDWARE DISCOUNT |
| |

| | | | |
|---|---|---|---|
| **SALES REP:** | **AMANDA MILLER** | **PHONE:** | **1800-234-9999** |
| **Email Address:** | **Amanda_Miller@Dell.com** | **Phone Ext:** | **7244623** |

**For your convenience, your sales representative, quote number and customer number have been
included to provide you with faster service when you are ready to place your order. You may also
place your order online at www.dell.com/quote**

**This quote is subject to the terms of the agreement signed by you and Dell, or absent such agreement,
is subject to the applicable Dell terms and conditions agreement.**

**Prices and tax rates are valid in the U.S. only and are subject to change.**

***Sales/use tax is a destination charge, i.e. based on the "ship to" address on your purchase order.
Please indicate your taxability status on your PO. If exempt, please fax exemption certificate to
Dell Tax Department at 800-433-9023, referencing your customer number.
If you have any questions regarding tax please call 800-433-9019 or email
Tax_Department@dell.com ****

**All product and pricing information is based on latest information available. Subject to change without
notice or obligation.**

**LCD panels in Dell products contain mercury, please dispose properly. Please contact Dell's Asset Recovery Services group for EPA compliant disposal options at US_Dell_ARS_Requests@dell.com. Minimum quantities may apply.**

# 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 "DBA"."collect_statistics_on_dml_updates"='off'
go

SET OPTION "DBA"."update_statistics"='off'
go

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

### Database command line

-x tcpip –n tpcc_newton –ca0 –ch14500m –cl14500m –c14500m –gr9999 –gc59 –gn 500

### Operating system

Page file: 3000MB fixed size, drive C:

### TPCCIsapi.dll

Worker threads: 420

Delivery threads: 80

### COM

Tpcccom.TPCCTran.1 component: Minimum pool size = Maximum pool size = 300

### Virtual disks:

All disks: Write Through cache policy; no patrol reads

JDrive: Disk cache enabled

# APPENDIX D: DATABASE DESIGN

```
--
-- This command file reloads a database that was unloaded using "dbunload".
--
-- (Version:  11.0.0.1264)
--


-- Database file: d:\tpcc\tpcc.db
-- Database CHAR collation: 1252LATIN1, NCHAR collation: UCA
-- Connection Character Set: windows-1252
--
-- CREATE DATABASE command: CREATE DATABASE 'd:\\tpcc\\tpcc.db' LOG ON 'f:\\tpcc\\tpcc.log' CASE IGNORE ACCENT IGNORE
PAGE SIZE 4096 COLLATION '1252LATIN1' NCHAR COLLATION 'UCA' BLANK PADDING OFF JCONNECT OFF CHECKSUM OFF
--


SET OPTION date_order          = 'YMD'
go

SET OPTION PUBLIC.preserve_source_format = 'OFF'
go

SET TEMPORARY OPTION tsql_outer_joins = 'ON'
go


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

CREATE DBSPACE "cs_space" AS 'j:\tpcc\cs.dbs'
go

CREATE DBSPACE "misc_space" AS 'm:\tpcc\misc.dbs'
go


-------------------------------------------------
--   Create login policies
-------------------------------------------------


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

GRANT CONNECT,DBA,RESOURCE TO "DBA" IDENTIFIED BY sql
go

GRANT CONNECT,DBA,GROUP,RESOURCE TO "dbo"
go


-------------------------------------------------
--   Create user types
-------------------------------------------------


-------------------------------------------------
--   Create group memberships
-------------------------------------------------
```

```
------------------------------------------------
--   Create remote servers
------------------------------------------------


------------------------------------------------
--   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 external environments
------------------------------------------------

IF EXISTS( SELECT * FROM SYS.SYSEXTERNENV WHERE name = 'java' ) THEN
   ALTER EXTERNAL ENVIRONMENT "java"
      LOCATION ''
END IF
go

IF EXISTS( SELECT * FROM SYS.SYSEXTERNENV WHERE name = 'perl' ) THEN
   ALTER EXTERNAL ENVIRONMENT "perl"
      LOCATION 'perl'
END IF
go

IF EXISTS( SELECT * FROM SYS.SYSEXTERNENV WHERE name = 'clr' ) THEN
   ALTER EXTERNAL ENVIRONMENT "clr"
      LOCATION 'dbextclr11'
END IF
go

IF EXISTS( SELECT * FROM SYS.SYSEXTERNENV WHERE name = 'php' ) THEN
   ALTER EXTERNAL ENVIRONMENT "php"
      LOCATION 'php'
END IF
go

IF EXISTS( SELECT * FROM SYS.SYSEXTERNENV WHERE name = 'c_esql32' ) THEN
```

```
    ALTER EXTERNAL ENVIRONMENT "c_esql32"
        LOCATION 'bin32[SLASH]dbexternc11'
END IF
go

IF EXISTS( SELECT * FROM SYS.SYSEXTERNENV WHERE name = 'c_odbc32' ) THEN
    ALTER EXTERNAL ENVIRONMENT "c_odbc32"
        LOCATION 'bin32[SLASH]dbexternc11'
END IF
go

IF EXISTS( SELECT * FROM SYS.SYSEXTERNENV WHERE name = 'c_esql64' ) THEN
    ALTER EXTERNAL ENVIRONMENT "c_esql64"
        LOCATION 'bin64[SLASH]dbexternc11'
END IF
go

IF EXISTS( SELECT * FROM SYS.SYSEXTERNENV WHERE name = 'c_odbc64' ) THEN
    ALTER EXTERNAL ENVIRONMENT "c_odbc64"
        LOCATION 'bin64[SLASH]dbexternc11'
END IF
go


------------------------------------------------
--   Create external environment objects
------------------------------------------------


------------------------------------------------
--   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
) 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

CREATE TABLE "DBA"."audit" (
    "start_d"                     "datetime" NULL
)
go

CREATE TABLE "DBA"."sample_w" (
    "sample_w_id"                  integer NULL
   ,"sample_number"                smallint NULL
)
go

CREATE TABLE "DBA"."t_stock" (
    "s_order_cnt"                  integer NULL
   ,"s_remote_cnt"                 integer NULL
   ,"cnt"                        integer NULL
)
go

create temporary procedure sa_unload_display_table_status(
    msgid int, ord int, numtabs int, user_name char(128), table_name char(128) )
begin
  declare @fullmsg long varchar;
  set @fullmsg = lang_message( msgid ) ||
      ' (' || ord || '/' || numtabs || ') ' ||
      '"' || user_name || '"."' || table_name || '"';
  message @fullmsg to client;
end
go

commit work
```

```
go


------------------------------------------------
--  Create text configurations
------------------------------------------------

commit
go



------------------------------------------------
--  Create materialized views
------------------------------------------------

commit
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 "misc_space"
go

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

call sa_unload_display_table_status( 17738, 12, 13, 'DBA', 'sample_w' )
go

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

commit work
go


-------------------------------------------------
--   Create immediate materialized views
-------------------------------------------------

commit
go



-------------------------------------------------
--   Create functions
-------------------------------------------------

commit
go



-------------------------------------------------
--   Create views
-------------------------------------------------
```

```
commit
go


SET TEMPORARY OPTION force_view_creation='ON'
go

SET TEMPORARY OPTION force_view_creation='OFF'
go

call dbo.sa_recompile_views(1)
go


------------------------------------------------
--   Create user messages
------------------------------------------------


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

commit
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 )
on exception resume
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 )
on exception resume
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

create procedure DBA.tpcc_orderstatus(
  in @w_id smallint,
  in @d_id tinyint,
  in @c_id integer,
  in @c_last char(16) default '' )
on exception resume
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 '' )
on exception resume
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 '' )
on exception resume
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 '' )
on exception resume
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 )
on exception resume
```

```
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 )
on exception resume
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_delivery(
  in @w_id smallint,
  in @o_carrier_id smallint )
on exception resume
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 )
on exception resume
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()
on exception resume
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()
on exception resume
begin
  declare @max_w_id smallint;
  select max(w_id) into @max_w_id
    from warehouse;
  rollback work;
  select @max_w_id
end
}
go

call dbo.sa_recompile_views(0)
go


------------------------------------------------
--   Create triggers
------------------------------------------------

commit
go



------------------------------------------------
--   Create SQL Remote definitions
------------------------------------------------


------------------------------------------------
--   Create MobiLink definitions
------------------------------------------------
```

```
------------------------------------------------
--  Create Synchronization profiles
------------------------------------------------


------------------------------------------------
--  Create logins
------------------------------------------------


------------------------------------------------
--  Create events
------------------------------------------------

commit
go



------------------------------------------------
--  Create services
------------------------------------------------

commit
go



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

GRANT CONNECT TO DBA IDENTIFIED BY ENCRYPTED
'\x01\x43\x23\xab\x07\x6a\xdc\x3e\x41\x08\xf0\xf7\x0e\x97\xa0\x87\x35\xe1\x49\xb0\xf1\x8d\xd5\x2d\x38\xde\x5b\x32\xdf\x4e
\x33\x04\xb1\xd0\x8d\x64\xae'
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

SET OPTION "DBA"."collect_statistics_on_dml_updates"='off'
go

SET OPTION "DBA"."update_statistics"='off'
go
```

# APPENDIX E: SYSTEM CONFIGURATION

'Engine properties'
0,'ActiveReq','Active requests','1'
1,'AvailIO','Number of available I/O control blocks','255'
2,'BytesReceived','Bytes received by server','5746'
3,'BytesReceivedUncomp','Bytes received after decompression','5746'
4,'BytesSent','Bytes sent to client','3901'
5,'BytesSentUncomp','Bytes sent before compression','3901'
6,'CacheAllocated','Cache pages that have been allocated for server data structures','7925'
7,'CacheFile','Cache pages used to hold data from database files','5681'
8,'CacheFileDirty','Cache pages that are dirty (needing a write)','5'
9,'CacheFree','Number of cache pages not being used','9'
10,'CacheHits','Cache Hits','264682'
13,'CachePanics','Number of times the cache manager has failed to find a page to allocate','0'
14,'CachePinned','Pinned cache pages','7949'
15,'CacheRead','Cache reads','269884'
16,'CacheReplacements','Cache replacements','0'
17,'CacheScavengeVisited','Number of pages visited while scavenging for a page to allocate','13615'
18,'CacheScavenges','Number of times the cache manager has scavenged for a page to allocate','13615'
21,'CarverHeapPages','Cache pages used for carvers','0'
39,'ClientStmtCacheHits','Number of prepares not required because of the client statement cache','1'
40,'ClientStmtCacheMisses','Number of prepares in the client statement cache which were prepared again','0'
44,'CurrentCacheSize','Current cache size in kilobytes','14848000'
51,'DiskRead','Disk reads','5626'
54,'DiskReadHintScatterLimit','Imposed limit on the size (in bytes) of a scatter read hint','0'
55,'DiskRetryRead','Disk read retries','0'
56,'DiskRetryReadScatter','Disk read retries for scattered reads','0'
57,'DiskRetryWrite','Disk write retries','0'
67,'ExchangeTasks','Exchange tasks currently being executed','0'
68,'ExchangeTasksCompleted','Total number of exchange tasks that have been executed','0'
79,'FreeBuffers','Free communication buffers','8'
86,'HeapsCarver','Number of heaps used for carvers','0'
87,'HeapsLocked','Number of relocatable heaps currently locked in cache','3'
88,'HeapsQuery','Number of heaps used for query processing (hash and sort operations)','0'
89,'HeapsRelocatable','Number of relocatable heaps','10'
99,'LockedCursorPages','Number of pages used to keep cursor heaps pinned in memory','2'
100,'LockedHeapPages','Heap pages locked in cache','24'
101,'MainHeapBytes','Main heap bytes in cache','15545968'
102,'MainHeapPages','Main heap pages in cache','7343'
103,'MapPhysicalMemoryEng','Map physical memory','0'
104,'MaxCacheSize','Maximum cache size in kilobytes','14848000'
108,'MinCacheSize','Minimum cache size in kilobytes','14848000'
109,'MultiPacketsReceived','Number of multi-packet receives','0'
110,'MultiPacketsSent','Number of multi-packet sends','0'
111,'MultiPageAllocs','Number of multi-page allocations','5'
113,'PacketsReceived','Packets received by server','58'
114,'PacketsReceivedUncomp','Packets received after decompression','58'
115,'PacketsSent','Packets sent to client','57'
116,'PacketsSentUncomp','Packets sent before compression','57'
118,'PeakCacheSize','Peak cache size in kilobytes','14848000'
123,'QueryHeapPages','Cache pages used for query processing (hash and sort operations)','0'
126,'QueryMemActiveCurr','The current number of requests actively using query memory','0'
127,'QueryMemActiveEst','The server''s estimate of the steady state average of the number of requests actively using query memory','0'
128,'QueryMemExtraAvail','The amount of memory available to grant beyond the base memory-intensive grant','0'
129,'QueryMemGrantFailed','The total number of times any request waited for query memory and failed to get it','0'
130,'QueryMemGrantGranted','The number of pages currently granted to requests','0'
131,'QueryMemGrantRequested','The total number of times any request attempted to acquire query memory','0'
132,'QueryMemGrantWaited','The total number of times any request waited for query memory','0'
133,'QueryMemGrantWaiting','The current number of requests waiting for query memory','0'
147,'RemoteputWait','Remote put waits','0'
148,'Req','Requests','59'

149,'RequestsReceived','Requests received by server','56'
152,'SendFail','Failed communication sends','0'
161,'StreamsUsed','Number of engine streams in use','1'
164,'TotalBuffers','Total communication buffers','10'
166,'UniqueClientAddresses','Number of unique client network addresses connected','0'
167,'UnschReq','Unscheduled requests','0'
171,'Name','Name','tpcc_newton'
174,'PageSize','Database page size','4096'
186,'TcpIpAddresses','TCP/IP addresses that can be used to connect via TCP/IP','192.168.5.5:2638;10.25.107.85:2638'
192,'Platform','Operating system platform','Windows2003'
193,'PlatformVer','Operating system platform version','Windows 2003 Build 3790 Service Pack 2'
194,'MessageWindowSize','Maximum number of messages retained from server window','400'
195,'MaxMessage','Maximum message number in server window','30'
196,'Message','Message from server window','07/15 19:43:13. SQL Anywhere Network Server Version
11.0.0.1264\x0D\x0A\x0D\x0A'
197,'MessageText','Message text from server window','SQL Anywhere Network Server Version 11.0.0.1264\x0D\x0A\x0D\x0A'
198,'MessageTime','Message time from server window','2008-07-15 19:43:13.391'
200,'ProductName','Product name','SQL Anywhere'
201,'ProductVersion','Product version','11.0.0.1264'
202,'CompanyName','Company name','iAnywhere Solutions, Inc.'
203,'LegalCopyright','Copyright notice','Copyright ? 2001-2008, iAnywhere Solutions, Inc. Portions copyright ? 1988-2008, Sybase,
Inc. All rights reserved. Use of this software is governed by the Sybase License Agreement. Refer to
http://www.sybase.com/softwarelicenses'
204,'LegalTrademarks','Legal trademarks','Sybase is a trademark of Sybase, Inc.'
205,'QuittingTime','Server quitting time','none'
206,'ConnsDisabled','Connections disabled','No'
207,'RequestLogging','Request logging','NONE'
208,'RequestLogFile','Request logging filename',''
209,'RequestLogNumFiles','Request log file count','-1'
210,'RequestLogMaxSize','Request log maximum size','0'
211,'RequestFilterConn','Request log connection filter','-1'
212,'RequestFilterDB','Request log database filter','-1'
213,'LivenessTimeout','Client liveness timeout default','120'
214,'ProfileFilterConn','Procedure profile connection filter','-1'
215,'ProfileFilterUser','Procedure profile user filter',''
218,'CharSet','Character set used for CHAR data','windows-1252'
221,'Language','Language','us_english'
225,'DefaultCollation','Default Collation','1252LATIN1'
226,'DefaultNcharCollation','Default NCHAR Collation','UCA'
234,'TimeZoneAdjustment','Time zone adjustment from UTC time in minutes','-240'
236,'NumPhysicalProcessors','Number of physical processors on server machine','1'
237,'NumPhysicalProcessorsUsed','Number of physical processors which the server will use','1'
238,'NumLogicalProcessors','Number of logical processors on server machine','4'
239,'NumLogicalProcessorsUsed','Number of logical processors which the server will use','4'
240,'LicensedCompany','Name of the licensed company','Sybase, Inc. (Waterloo)'
241,'LicensedUser','Name of the licensed user','Internal Use'
242,'LicenseType','License type','networked seat (per-seat)'
243,'LicenseCount','Number of licensed seats','10000'
248,'MachineName','Name of the machine','NEWTON'
250,'CompactPlatformVer','Operating system platform version in compact form','W.NET #3790 SP 2 '
251,'CommandLine','Expanded command line used to start the server','-x tcpip -n tpcc_newton -ca0 -ch14500m -cl14500m -c
14500m -cr- -gr9999 -gc59 -gn 500 -o c:\\tpcc_output_logs\\serverlog.out d:\\tpcc\\tpcc.db '
252,'CollectStatistics','Server and database performance statistics are collected','Yes'
254,'MessageCategoryLimit','The minimum number of console messages of each type stored by the server','400'
256,'ProcessCPU','Process CPU usage','4.406250'
257,'ProcessCPUUser','Process CPU User usage','3.984375'
258,'ProcessCPUSystem','Process CPU System usage','0.421875'
259,'IsNetworkServer','Is Network Server','Yes'
260,'IsRuntimeServer','Is Runtime Server','No'
261,'IsService','Is Service','Yes'
262,'BuildClient','Codeline name','dbbuild_11_1100_ga'
263,'BuildChange','Codeline revision','565181'
264,'BuildReproducible','Codeline state','0'
265,'BuildProduction','Build type','Yes'
275,'RememberLastStatement','Remember last statement','No'
276,'RememberLastPlan','Remember last plan','No'
279,'FunctionName','Function name','abs'

280,'FunctionMinParms','Function minimum number of parameters','1'
281,'FunctionMaxParms','Function maximum number of parameters','1'
282,'OmniIdentifier','Omni Identifier','1056304543:tpcc:tpcc_newton:13922000'
283,'IdleTimeout','Idle timeout','240'
284,'ProcessorArchitecture','Processor architecture','X86_64'
285,'NativeProcessorArchitecture','Architecture of native processor when current processor is emulated','X86_64'
286,'TempDir','Temporary directory','j:\\temp'
287,'StartTime','Server start time','2008-07-15 19:43:17.328'
289,'IsIQ','Is IQ Server','No'
290,'MultiProgrammingLevel','The maximum number of tasks which the server can process concurrently','500'
293,'ConsoleLogFile','Console log filename','c:\\tpcc_output_logs\\serverlog.out'
300,'HttpPorts','Http Server TCP/IP port number(s)',''
301,'HttpsPorts','Https Server TCP/IP port number(s)',''
302,'HttpAddresses','TCP/IP addresses that can be used to connect via HTTP',''
303,'HttpsAddresses','TCP/IP addresses that can be used to connect via HTTPS',''
305,'HttpNumConnections','Total number of HTTP connections','0'
306,'HttpsNumConnections','Total number of HTTPS (secure) connections','0'
307,'HttpNumActiveReq','Total number of active HTTP connections','0'
308,'HttpsNumActiveReq','Total number of active HTTPS (secure) connections','0'
309,'HttpNumSessions','Total number of HTTP sessions (passive and active)','0'
316,'IsFipsAvailable','FIPS Available','No'
317,'FipsMode','FIPS Mode','No'
319,'StartDBPermission','Start database permission','DBA'
324,'ServerName','Server name','tpcc_newton'
330,'RequestTiming','Monitor the cost of requests','No'
343,'CacheSizingStatistics','Display cache sizing statistics','No'
344,'ConsoleLogMaxSize','Console log maximum size','0'
345,'DebuggingInformation','Display debugging information','No'
346,'WebClientLogging','Enable HTTP web client procedure debug logging','No'
347,'WebClientLogFile','Set HTTP web client procedure debugging log filename',''
350,'IsRsaAvailable','RSA Available','Yes'
351,'IsEccAvailable','ECC Available','Yes'
352,'MaxConnections','The maximum number of concurrent connections to the server','32766'
354,'FirstOption','First option property number','398'
355,'LastOption','Last option property number','512'
356,'LastConnectionProperty','Last connection property number','512'
357,'LastDatabaseProperty','Last database property number','379'
358,'LastServerProperty','Last server property number','379'
364,'QueryMemPercentOfCache','The amount of memory that is available for query execution algorithms, expressed as a percent of maximum cache size','50'
365,'QueryMemPages','The amount of memory that is available for query execution algorithms, expressed as a number of pages','1798734'
366,'QueryMemGrantBase','The minimum amount of memory granted to all requests','100'
367,'QueryMemGrantBaseMI','The minimum amount of memory granted to memory-intensive requests','3597'
368,'QueryMemGrantExtra','The number of query memory pages that can be distributed beyond the base memory-intensive grant','1361609'
369,'QueryMemActiveMax','Maximum number of requests actively using query memory','125'
375,'RemoteCapability','Remote capability',
376,'MaxRemoteCapability','Maximum remote capability','287'
377,'EventTypeName','Event type name','GrowDB'
378,'EventTypeDesc','Event type description','database file extended'
379,'MaxEventType','Maximum event type','16'
'

DB Properties'
0,10,'CacheHits','Cache Hits','265001'
0,11,'CacheReadIndInt','Cache index interior reads','215'
0,12,'CacheReadIndLeaf','Cache index leaf reads','5689'
0,15,'CacheRead','Cache reads','270203'
0,19,'CacheReadTable','Cache table reads','1445'
0,20,'CacheReadWorkTable','Cache work table reads','1'
0,22,'ChkptFlush','Checkpoint flushed pages','2'
0,23,'ChkptPage','Checkpoint log page images saved','7'
0,24,'CheckpointUrgency','Checkpoint Urgency','1'
0,25,'Chkpt','Checkpoints','1'
0,26,'CheckpointLogBitmapSize','Checkpoint log bitmap size','0'
0,27,'CheckpointLogBitmapPagesWritten','Checkpoint log writes to bitmap','0'

0,28,'CheckpointLogCommitToDisk','Checkpoint log commit to disk','4'
0,29,'CheckpointLogPageInUse','Checkpoint log pages in use','5'
0,30,'CheckpointLogPagesRelocated','Checkpoint log pages relocated','0'
0,31,'CheckpointLogSavePreimage','Checkpoint log save preimage','7'
0,32,'CheckpointLogSize','Checkpoint log size in pages','5'
0,33,'CheckpointLogPagesWritten','Checkpoint log pages written','2'
0,34,'CheckpointLogWrites','Checkpoint log disk writes','2'
0,35,'CleanablePagesAdded','Cleanable Pages Added','15937'
0,36,'CleanablePagesCleaned','Cleanable Pages Cleaned','5016'
0,37,'CleanableRowsAdded','Cleanable Rows Added','0'
0,38,'CleanableRowsCleaned','Cleanable Rows Cleaned','0'
0,42,'CommitFile','Commit writes to disk','27'
0,43,'ConnCount','Number of active connections','1'
0,45,'CurrIO','Active disk read/write requests','0'
0,46,'CurrRead','Active disk read requests','0'
0,47,'CurrWrite','Active disk write requests','0'
0,49,'DiskReadIndInt','Disk index interior reads','35'
0,50,'DiskReadIndLeaf','Disk index leaf reads','5026'
0,51,'DiskRead','Disk reads','5626'
0,52,'DiskReadHint','Disk read hints','416'
0,53,'DiskReadHintPages','Disk read hint pages','475'
0,56,'DiskRetryReadScatter','Disk read retries for scattered reads','0'
0,58,'DiskSyncRead','Disk reads issued synchronously','5210'
0,59,'DiskSyncWrite','Disk writes issued synchronously','0'
0,60,'DiskReadTable','Disk table reads','484'
0,61,'DiskWaitRead','Number of times the server waited for an asynchronous read','413'
0,62,'DiskWaitWrite','Number of times the server waited for an asynchronous write','7'
0,63,'DiskReadWorkTable','Disk work table reads','0'
0,64,'DiskWrite','Disk writes','268'
0,65,'DiskWriteHint','Disk write hints','0'
0,66,'DiskWriteHintPages','Disk write hint pages','0'
0,69,'ExprCacheAbandons','Number of time that the expression cache was completely abandoned due to the hit rate being too low','0'
0,70,'ExprCacheDropsToReadOnly','Number of times that the expression cache dropped to read-only status due to the hit rate being low','0'
0,71,'ExprCacheEvicts','Number of evictions from the expression cache','0'
0,72,'ExprCacheHits','Number of hits in the expression cache','0'
0,73,'ExprCacheInserts','Number of values inserted into the expression cache','0'
0,74,'ExprCacheLookups','Number of lookups done in the expression cache','0'
0,75,'ExprCacheResumesOfReadWrite','Number of times that the expression cache resumed read-write status due to the hit rate rising again','0'
0,76,'ExprCacheStarts','Number of times the expression cache was started','0'
0,77,'ExtendDB','Extend database file writes','0'
0,78,'ExtendTempWrite','Extend temporary file writes','524352'
0,80,'FullCompare','Number of comparisons beyond the hash value','0'
0,81,'GetData','GETDATA requests','2'
0,82,'HashForcedPartitions','Times that a hash operator was forced to partition due to competition for memory','0'
0,83,'HashRowsFiltered','Number of probe rows rejected by bit-vector filters','0'
0,84,'HashRowsPartitioned','Number of rows written to hash work tables','0'
0,85,'HashWorkTables','Number of work tables created for hash-based operations','0'
0,90,'IOToRecover','Recovery I/O Estimate','9'
0,91,'IdleCheck','Idle I/O checked','24'
0,92,'IdleChkTime','Idle I/O checkpoint time','0'
0,93,'IdleChkpt','Idle I/O checkpoints','0'
0,94,'IdleWrite','Idle I/O writes','0'
0,95,'IndAdd','Number of index insertions','0'
0,96,'IndLookup','Number of index lookups','0'
0,97,'LockCount','Number of locks','0'
0,98,'LockTablePages','Lock table pages','0'
0,105,'MaxIO','Maximum active disk read/write requests','2'
0,106,'MaxRead','Maximum active disk read requests','2'
0,107,'MaxWrite','Maximum active disk write requests','2'
0,117,'PageRelocations','Page relocations','0'
0,119,'ProcedurePages','Procedure relocatable heap pages','15'
0,120,'QueryBypassed','Queries bypassing optimization','20'
0,121,'QueryCachePages','Pages used to cache query plans','0'
0,122,'QueryCachedPlans','Query plans stored in cache','0'

0,124,'QueryJHToJNLOptUsed','Number of times a hash join was converted to a nested loop join','0'
0,125,'QueryLowMemoryStrategy','Low memory strategies used by queries','0'
0,134,'QueryOptimized','Queries optimized','5'
0,135,'QueryReused','Reused query plans','0'
0,136,'QueryRowsBufferFetch','Number of rows fetched using buffering','0'
0,137,'QueryRowsMaterialized','Number of rows written to work tables during query processing','159'
0,138,'RecoveryUrgency','Recovery Urgency','0'
0,139,'RecursiveIterations','Number of iterations for recursive unions','0'
0,140,'RecursiveIterationsHash','Number of times recursive hash join used hash strategy','0'
0,141,'RecursiveIterationsNested','Number of times recursive hash join used nested loop strategy','0'
0,142,'RecursiveJNLMisses','Number of index probe cache misses for recursive hash join','0'
0,143,'RecursiveJNLProbes','Number of times recursive hash join attempted an index probe','0'
0,144,'LogFreeCommit','Transaction log group commits','0'
0,145,'LogWrite','Transaction log page writes','0'
0,146,'RelocatableHeapPages','Relocatable heap pages','59'
0,151,'RollbackLogPages','Rollback log pages','0'
0,153,'SnapshotCount','Number of active snapshots','0'
0,154,'SortMergePasses','Number of merge passes used during sorting','0'
0,155,'SortRowsMaterialized','Number of rows written to sort work tables','0'
0,156,'SortRunsWritten','Number of sorted runs written during sorting','0'
0,157,'SortSortedRuns','Number of sorted runs created during run formation','0'
0,158,'SortWorkTables','Number of work tables created for sorting','0'
0,163,'TempTablePages','Temporary table pages','4'
0,165,'TriggerPages','Trigger relocatable heap pages','0'
0,168,'VersionStorePages','Version store pages','0'
0,169,'ViewPages','View relocatable heap pages','2'
0,170,'XPathCompiles','XPath Compiles','0'
0,171,'Name','Name','tpcc'
0,172,'Alias','Mounted database name','tpcc'
0,173,'File','Database file','d:\\tpcc\\tpcc.db'
0,174,'PageSize','Database page size','4096'
0,175,'LogName','Database log file name','f:\\tpcc\\tpcc.log'
0,176,'LogMirrorName','Database log mirror file name',
0,177,'TempFileName','Database temporary file name','j:\\temp\\sqla0000.tmp'
0,199,'IQStore','IQ store is on/off','Off'
0,206,'ConnsDisabled','Connections disabled','Off'
0,218,'CharSet','Character set used for CHAR data','windows-1252'
0,219,'NcharCharSet','Character set used for NCHAR data','UTF-8'
0,220,'MultiByteCharSet','Multi Byte Character Set ( on/off )','Off'
0,221,'Language','Language','nl,da,it,pt,de,fr,en'
0,222,'Collation','Name of collation used for CHAR data','1252LATIN1'
0,223,'NcharCollation','Name of collation used for NCHAR data','UCA'
0,224,'ReadOnly','Database read-only mode','Off'
0,227,'LTMTrunc','LTM truncation point','18446744073709551615'
0,228,'RemoteTrunc','SQL Remote truncation point','18446744073709551615'
0,229,'SyncTrunc','DBMLSync truncation point','18446744073709551615'
0,230,'GlobalDBID','Global database identifier','2147483647'
0,231,'CurrentRedoPos','Current redo position','429044'
0,232,'LTMGeneration','LTM generation number','0'
0,233,'ProcedureProfiling','Procedure profiling','Off'
0,244,'CaseSensitive','Case sensitivity','Off'
0,245,'AccentSensitive','Accent sensitivity for UCA collations','Off'
0,246,'BlankPadding','Blank padding','Off'
0,249,'Checksum','Page checksum','Off'
0,253,'CatalogCollation','Name of collation used for catalog data','1252LATIN1'
0,255,'DriveType','Drive type where the database is located','FIXED'
0,268,'Encryption','Encryption type','None'
0,269,'Capabilities','Database capability bits','3000D000DEDFF7FDD'
0,271,'DBFileFragments','Database file fragments','4'
0,272,'LogFileFragments','Log file fragments','102'
0,292,'AuditingTypes','Auditing types enabled','all'
0,295,'FreePages','Number of free pages in dbspace','262092'
0,296,'IOParallelism','Estimated number of simultaneous I/O operations supported by dbspace','1'
0,297,'FileSize','File size in pages','263069'
0,311,'NextScheduleTime','Next schedule time',
0,318,'IdentitySignature','Identity signature','214780334'
0,320,'PartnerState','Partner state',

0,321,'ArbiterState','Arbiter state',
0,322,'MirrorState','Mirror state',
0,323,'MirrorMode','Mirror mode',
0,325,'AlternateServerName','Alternate server name',
0,326,'AlternateMirrorServerName','Alternate mirror server name',
0,327,'SnapshotIsolationState','State of snapshot isolation','Off'
0,328,'SendingTracingTo','Database where diagnostic tracing is being sent',''
0,329,'ReceivingTracingFrom','Database from which diagnostic tracing is being received',''
0,348,'EncryptionScope','Scope of encryption','None'
0,353,'JavaVM','The JAVA VM that will be used to run JAVA classes','F:\\sa\\sa11\\sun\\jre160_x64\\bin\\java.exe'
0,359,'DatabaseCleaner','State of the Database Cleaner','On'
0,360,'HasCollationTailoring','Has collation tailoring','Off'
0,363,'HasEndianSwapFix','Fixed endian swap implementation','On'
0,372,'OptionWatchList','Option watch list',''
0,373,'OptionWatchAction','Option watch action','message'
0,374,'HasNCHARLegacyCollationFix','Fixed NCHAR legacy collation implementation','On'
'

Default DBA Connection Properties'
2,2,'BytesReceived','Bytes received by server','5187'
2,3,'BytesReceivedUncomp','Bytes received after decompression','5271'
2,4,'BytesSent','Bytes sent to client','27450'
2,5,'BytesSentUncomp','Bytes sent before compression','27450'
2,10,'CacheHits','Cache Hits','995'
2,11,'CacheReadIndInt','Cache index interior reads','17'
2,12,'CacheReadIndLeaf','Cache index leaf reads','66'
2,15,'CacheRead','Cache reads','1072'
2,19,'CacheReadTable','Cache table reads','73'
2,20,'CacheReadWorkTable','Cache work table reads','3'
2,21,'CarverHeapPages','Cache pages used for carvers','0'
2,39,'ClientStmtCacheHits','Number of prepares not required because of the client statement cache','0'
2,40,'ClientStmtCacheMisses','Number of prepares in the client statement cache which were prepared again','0'
2,41,'Commit','Number of commit requests','1'
2,48,'Cursor','Declared cursors','3'
2,49,'DiskReadIndInt','Disk index interior reads','0'
2,50,'DiskReadIndLeaf','Disk index leaf reads','4'
2,51,'DiskRead','Disk reads','77'
2,52,'DiskReadHint','Disk read hints','0'
2,53,'DiskReadHintPages','Disk read hint pages','0'
2,58,'DiskSyncRead','Disk reads issued synchronously','77'
2,59,'DiskSyncWrite','Disk writes issued synchronously','0'
2,60,'DiskReadTable','Disk table reads','8'
2,61,'DiskWaitRead','Number of times the server waited for an asynchronous read','0'
2,62,'DiskWaitWrite','Number of times the server waited for an asynchronous write','0'
2,63,'DiskReadWorkTable','Disk work table reads','0'
2,64,'DiskWrite','Disk writes','0'
2,65,'DiskWriteHint','Disk write hints','0'
2,66,'DiskWriteHintPages','Disk write hint pages','0'
2,69,'ExprCacheAbandons','Number of time that the expression cache was completely abandoned due to the hit rate being too low','0'
2,70,'ExprCacheDropsToReadOnly','Number of times that the expression cache dropped to read-only status due to the hit rate being low','0'
2,71,'ExprCacheEvicts','Number of evictions from the expression cache','0'
2,72,'ExprCacheHits','Number of hits in the expression cache','0'
2,73,'ExprCacheInserts','Number of values inserted into the expression cache','0'
2,74,'ExprCacheLookups','Number of lookups done in the expression cache','0'
2,75,'ExprCacheResumesOfReadWrite','Number of times that the expression cache resumed read-write status due to the hit rate rising again','0'
2,76,'ExprCacheStarts','Number of times the expression cache was started','0'
2,80,'FullCompare','Number of comparisons beyond the hash value','0'
2,81,'GetData','GETDATA requests','3'
2,82,'HashForcedPartitions','Times that a hash operator was forced to partition due to competition for memory','0'
2,83,'HashRowsFiltered','Number of probe rows rejected by bit-vector filters','0'
2,84,'HashRowsPartitioned','Number of rows written to hash work tables','0'
2,85,'HashWorkTables','Number of work tables created for hash-based operations','0'
2,86,'HeapsCarver','Number of heaps used for carvers','0'
2,87,'HeapsLocked','Number of relocatable heaps currently locked in cache','3'

2,88,'HeapsQuery','Number of heaps used for query processing (hash and sort operations)','0'
2,89,'HeapsRelocatable','Number of relocatable heaps','16'
2,95,'IndAdd','Number of index insertions','0'
2,96,'IndLookup','Number of index lookups','0'
2,97,'LockCount','Number of locks','0'
2,112,'CursorOpen','Open cursors','3'
2,113,'PacketsReceived','Packets received by server','68'
2,114,'PacketsReceivedUncomp','Packets received after decompression','68'
2,115,'PacketsSent','Packets sent to client','68'
2,116,'PacketsSentUncomp','Packets sent before compression','68'
2,120,'QueryBypassed','Queries bypassing optimization','16'
2,121,'QueryCachePages','Pages used to cache query plans','0'
2,122,'QueryCachedPlans','Query plans stored in cache','0'
2,123,'QueryHeapPages','Cache pages used for query processing (hash and sort operations)','0'
2,124,'QueryJHToJNLOptUsed','Number of times a hash join was converted to a nested loop join','0'
2,125,'QueryLowMemoryStrategy','Low memory strategies used by queries','0'
2,126,'QueryMemActiveCurr','The current number of requests actively using query memory','0'
2,129,'QueryMemGrantFailed','The total number of times any request waited for query memory and failed to get it','0'
2,130,'QueryMemGrantGranted','The number of pages currently granted to requests','0'
2,131,'QueryMemGrantRequested','The total number of times any request attempted to acquire query memory','0'
2,132,'QueryMemGrantWaited','The total number of times any request waited for query memory','0'
2,133,'QueryMemGrantWaiting','The current number of requests waiting for query memory','0'
2,134,'QueryOptimized','Queries optimized','6'
2,135,'QueryReused','Reused query plans','0'
2,136,'QueryRowsBufferFetch','Number of rows fetched using buffering','0'
2,137,'QueryRowsMaterialized','Number of rows written to work tables during query processing','424'
2,139,'RecursiveIterations','Number of iterations for recursive unions','0'
2,140,'RecursiveIterationsHash','Number of times recursive hash join used hash strategy','0'
2,141,'RecursiveIterationsNested','Number of times recursive hash join used nested loop strategy','0'
2,142,'RecursiveJNLMisses','Number of index probe cache misses for recursive hash join','0'
2,143,'RecursiveJNLProbes','Number of times recursive hash join attempted an index probe','0'
2,144,'LogFreeCommit','Transaction log group commits','0'
2,145,'LogWrite','Transaction log page writes','0'
2,149,'RequestsReceived','Requests received by server','68'
2,150,'Rlbk','Rollback requests handled','0'
2,151,'RollbackLogPages','Rollback log pages','0'
2,153,'SnapshotCount','Number of active snapshots','0'
2,154,'SortMergePasses','Number of merge passes used during sorting','0'
2,155,'SortRowsMaterialized','Number of rows written to sort work tables','0'
2,156,'SortRunsWritten','Number of sorted runs written during sorting','0'
2,157,'SortSortedRuns','Number of sorted runs created during run formation','0'
2,158,'SortWorkTables','Number of work tables created for sorting','0'
2,159,'PrepStmt','Prepared statements','9'
2,160,'Prepares','Number of statement prepares','26'
2,162,'TempFilePages','Number of temporary file pages used by the connection','70'
2,163,'TempTablePages','Temporary table pages','4'
2,171,'Name','Name','SQL_DBC_d864e18'
2,178,'DBNumber','Database number','0'
2,179,'UserID','User ID','DBA'
2,180,'LastReqTime','Last request time','2008-07-15 19:44:05.758'
2,181,'ReqType','Type of active request','PREFETCH'
2,182,'CommLink','Communication link','local'
2,183,'NodeAddress','Client node address',''
2,184,'ClientNodeAddress','Client node address',''
2,185,'ServerNodeAddress','Server node address',''
2,187,'Number','Connection ID','2'
2,188,'LastIdle','Ticks between requests','0'
2,189,'BlockedOn','Connection blocked on','0'
2,190,'LockName','Lock name blocked on','0'
2,191,'UncommitOp','Uncommitted operations','0'
2,213,'LivenessTimeout','Client liveness timeout default','0'
2,216,'CommProtocol','Communication protocol name','CmdSeq'
2,217,'ClientLibrary','Client library name','CmdSeq'
2,218,'CharSet','Character set used for CHAR data','windows-1252'
2,219,'NcharCharSet','Character set used for NCHAR data','UTF-8'
2,221,'Language','Language','us_english'
2,234,'TimeZoneAdjustment','Time zone adjustment from UTC time in minutes','-240'

2,235,'EventName','Name of the event this connection is handling',''
2,247,'Compression','Compression enabled','Off'
2,266,'AppInfo','Application Information','IP=192.168.5.5;HOST=NEWTON;OSUSER=Administrator;OS='"Windows 2003 Build 3790 Service Pack
2";EXE=F:\\sa\\sa11\\bin32\\dbisql.com;PID=0x90;THREAD=0xecc;VERSION=11.0.0.1405;API=iAnywhereJDBC;TIMEZONEADJUSTMENT=-240'
2,267,'UserAppInfo','User application information',''
2,268,'Encryption','Encryption type','None'
2,270,'UtilCmdsPermitted','Utility commands permitted','On'
2,273,'PacketSize','Packet size','7300'
2,274,'LastStatement','Last statement',''
2,277,'LastPlanText','Last text plan',''
2,278,'CurrentLineNumber','Current line number',''
2,283,'IdleTimeout','Idle timeout','240'
2,288,'TransactionStartTime','Transaction start time',''
2,291,'CurrentProcedure','Stored procedure currently being executed',''
2,294,'CommNetworkLink','Communication link (never local)','SharedMemory'
2,298,'ClientPort','Client TCP/IP port number','0'
2,299,'ServerPort','Server TCP/IP port number','0'
2,304,'HttpServiceName','Http service name',''
2,310,'LoginTime','Login time','2008-07-15 19:44:05.630'
2,312,'MessageReceived','Message received',''
2,313,'SessionID','HTTP session ID',''
2,314,'SessionCreateTime','HTTP session create time',''
2,315,'SessionLastTime','HTTP session last request time',''
2,331,'ReqStatus','Current request status','Executing'
2,332,'ReqCountUnscheduled','Number of times waited for scheduling',
2,333,'ReqCountActive','Number of requests processed',
2,334,'ReqCountBlockIO','Number of times waited for I/O to complete',
2,335,'ReqCountBlockLock','Number of times waited for a lock',
2,336,'ReqCountBlockContention','Number of times waited for atomic access',
2,337,'ReqTimeUnscheduled','Time spent unscheduled',
2,338,'ReqTimeActive','Time spent processing requests',
2,339,'ReqTimeBlockIO','Time spent waiting for I/O to complete',
2,340,'ReqTimeBlockLock','Time spent waiting for a lock',
2,341,'ReqTimeBlockContention','Time spent waiting for atomic access',
2,342,'ApproximateCPUTime','Approximate CPU time used','0'
2,349,'LockTableOID','Object ID of locked table','0'
2,361,'LockRowID','Identifier of the locked row','0'
2,362,'LockIndexID','Identifier of the locked index',''
2,370,'AuthType','The authentication type used when establishing the connection','Standard'
2,371,'OSUser','The operating system user name associated with the client process','Administrator'
2,399,'blocking_timeout','Controls the time a transaction waits to obtain a lock','0'
2,400,'checkpoint_time','Maximum number of minutes between checkpoints','59'
2,401,'conversion_error','Controls datatype conversion errors','On'
2,402,'date_format','Controls format for DATE values','YYYY-MM-DD'
2,403,'date_order','Controls order of date components','YMD'
2,404,'isolation_level','Controls the locking isolation level','0'
2,405,'updatable_statement_isolation','The isolation level for updatable statements when using readonly-statement-snapshot','0'
2,406,'lock_rejected_rows','Reserved','Off'
2,407,'login_procedure','Procedure to be run during login','sp_login_environment'
2,408,'on_tsql_error','Controls error handling in stored procedures','Conditional'
2,409,'precision','Maximum number of digits in decimal arithmetic','30'
2,410,'recovery_time','Maximum time to allow for database recovery','9999'
2,411,'replicate_all','Allows entire database to act as primary site for Rep Server','Off'
2,412,'row_counts','Controls whether row counts are estimates or exact','Off'
2,413,'scale','Minimum number of digits after decimal point','6'
2,414,'timestamp_format','Controls format for TIMESTAMP values','YYYY-MM-DD HH:NN:SS.SSS'
2,415,'time_format','Controls format for TIME values','HH:NN:SS.SSS'
2,416,'wait_for_commit','Controls when foreign key integrity is checked','Off'
2,417,'quoted_identifier','Controls interpretation of strings enclosed in double quotes','On'
2,418,'allow_nulls_by_default','Controls NULL values for new columns','On'
2,419,'cooperative_commits','Controls when COMMITs are written to disk','On'
2,420,'cooperative_commit_timeout','Controls delay before a COMMIT is written to disk','250'
2,421,'delayed_commits','Controls when server returns control to application following a COMMIT','Off'
2,422,'delayed_commit_timeout','Controls delay before server returns control while waiting to do COMMIT','500'
2,423,'non_keywords','Controls what identifiers are keywords',''

2,424,'sql_flagger_error_level','Controls errors for SQL that is not from specified set of SQL/92','Off'
2,425,'sql_flagger_warning_level','Controls warnings for SQL that is not from specified set of SQL/92','Off'
2,426,'ansi_blanks','Controls truncation errors','Off'
2,427,'string_rtruncation','Controls truncation errors on INSERT or UPDATE','On'
2,428,'ansinull','Controls interpretation of NULL values','On'
2,429,'ansi_permissions','Controls permissions checking for DELETE and UPDATE statements','On'
2,430,'close_on_endtrans','Controls closing of cursors at end of transaction','On'
2,431,'tsql_variables','Controls whether @ can be used as host variable name prefix','Off'
2,432,'chained','Controls transaction mode if BEGIN TRANSACTION not used','On'
2,433,'nearest_century','Controls interpretation of two-digit years','50'
2,434,'fire_triggers','Controls whether triggers are fired in the database','On'
2,435,'background_priority','Controls priority of current connection','Off'
2,436,'login_mode','Controls integrated and Kerberos logins','Standard'
2,437,'integrated_server_name','Server name for determining user groups',''
2,438,'connection_authentication','Authentication string for
connection','Company=Sybase;Application=DBTools;Signature=Company=Sybase;Application=DBTools;Signature=000fa55157edb8
e14d818eb4fe3db41447146f'
2,439,'java_main_userid','Userid used by external java vm to connect to the database',''
2,440,'java_location','File path pointing to the external java vm',''
2,441,'java_vm_options','Command line options used to launch external java vm',''
2,442,'suppress_tds_debugging','Controls display of TDS debug information','On'
2,443,'upgrade_database_capability','Reserved',''
2,444,'database_authentication','Authentication string for database',''
2,445,'default_timestamp_increment','Number of microseconds to add to TIMESTAMP for next value','1'
2,446,'escape_character','Reserved','On'
2,447,'prefetch','Controls prefetching of rows','Conditional'
2,448,'continue_after_raiserror','Controls behavior following a RAISERROR statement','On'
2,449,'cis_option','Controls display of debug information for remote data access','0'
2,450,'cis_rowset_size','Controls number of fetched rows returned from remote servers','50'
2,451,'ansi_close_cursors_on_rollback','Controls whether WITH HOLD cursors are closed on ROLLBACK','Off'
2,452,'max_statement_count','Maximum number of prepared statements for a connection','50'
2,453,'max_cursor_count','Maximum number of cursors allowed for a connection','50'
2,454,'min_password_length','Minimum length for new database passwords','0'
2,455,'auditing','Enables and disables auditing','Off'
2,456,'conn_auditing','Enables and disables auditing on a connection','On'
2,457,'auditing_options','Reserved','4294967295'
2,458,'tds_empty_string_is_null','Controls whether TDS connections return empty strings as NULL or one space','Off'
2,459,'optimization_level','Controls amount of effort made by the query optimizer to find an access plan','9'
2,460,'extended_join_syntax','Controls errors when using duplicate correlation names in joins','On'
2,461,'ansi_update_constraints','Controls the range of updates that are permitted','Cursors'
2,462,'optimization_goal','Optimize queries for first row or all rows','All-rows'
2,463,'global_database_id','Controls initial value for DEFAULT GLOBAL AUTOINCREMENT columns','2147483647'
2,464,'max_hash_size','Deprecated','10'
2,465,'prevent_article_pkey_update','Controls updates to primary keys used in publications','On'
2,466,'truncate_timestamp_values','Controls precision of TIMESTAMP values','Off'
2,467,'return_date_time_as_string','Controls how DATE, TIME and TIMESTAMP values are fetched','On'
2,468,'first_day_of_week','Sets the numbering of the days of the week','7'
2,469,'preserve_source_format','Controls preservation of source for procedures, triggers, views, events','On'
2,470,'time_zone_adjustment','Numbers of minutes to add to UTC for this time zone','-240'
2,471,'exclude_operators','Reserved',''
2,472,'user_estimates','Controls whether to respect user estimates','Override-magic'
2,473,'max_plans_cached','Maximum number of cached execution plans for a connection','100'
2,474,'sort_collation','Controls implicit use of SORTKEY on ORDER BY of characters','Internal'
2,475,'pinned_cursor_percent_of_cache','Controls amount of server cache to be used for pinning cursors','10'
2,476,'on_charset_conversion_failure','Controls error handling for character set conversion failure','Ignore'
2,477,'update_statistics','Controls collection of statistics during query execution','Off'
2,478,'collect_statistics_on_dml_updates','Controls collection of statistics during INSERT/UPDATE/DELETE statement execution','Off'
2,479,'max_recursive_iterations','Maximum number of recursions for common table expressions','100'
2,480,'max_query_tasks','Maximum number of tasks that may be used by a parallel execution plan for a single query','0'
2,481,'for_xml_null_treatment','Controls treatment of NULL values in queries that use FOR XML','Omit'
2,482,'subsume_row_locks','Controls when server acquires row locks for table','On'
2,483,'read_past_deleted','Controls server behavior on uncommitted deletes','On'
2,484,'temp_space_limit_check','Controls whether a connection''s temp file usage is checked against an upper limit','On'
2,485,'force_view_creation','Controls errors when CREATE VIEW references non-existent view','Off'
2,486,'dedicated_task','Dedicates a server task to the current connection','Off'
2,487,'debug_messages','Controls whether MESSAGE ... DEBUG ONLY statements are executed','Off'
2,488,'optimization_workload','Controls whether optimizing for OLAP or mixed queries','Mixed'

2,489,'odbc_distinguish_char_and_varchar','Controls whether ODBC distinguishes CHAR and VARCHAR columns','Off'
2,490,'remote_idle_timeout','Controls the time that an HTTP procedure will wait for a response from a server','15'
2,491,'odbc_describe_binary_as_varbinary','Controls whether ODBC describes BINARY columns as VARBINARY','Off'
2,492,'rollback_on_deadlock','Controls whether or not a transaction does a rollback upon deadlock','On'
2,493,'log_deadlocks','Controls whether deadlocks are logged','Off'
2,494,'webservice_namespace_host','Server hostname specification for use in web services namespace',''
2,495,'http_session_timeout','Timeout setting for an HTTP session','30'
2,496,'request_timeout','Controls the maximum time a request is allowed to execute','0'
2,497,'synchronize_mirror_on_commit','Controls whether database mirror server is synchronized on commit','Off'
2,498,'verify_password_function','Function to be run to verify a new password conforms to password rules',''
2,499,'allow_snapshot_isolation','Controls enabling of snapshot isolation','Off'
2,500,'default_dbspace','Sets the default dbspace for table creation',''
2,501,'oem_string','Sets the OEM string in the database file header',''
2,502,'max_temp_space','Sets the maximum temp space that a connection may use','0'
2,503,'secure_feature_key','Sets the secure feature override key',''
2,504,'materialized_view_optimization','Controls the use of materialized views during query optimization','Stale'
2,505,'tsql_outer_joins','Controls whether TSQL outer joins can be used in DML statements','Off'
2,506,'post_login_procedure','Procedure to be called by the application when connecting to return
messages','dbo.sa_post_login_procedure'
2,507,'max_client_statements_cached','Maximum number of prepared statements cached by the client for a connection','10'
2,508,'query_mem_timeout','Controls the time a transaction waits to obtain query memory for query execution algorithms','-1'
2,509,'allow_read_client_file','Controls whether the use of reading data from the client machine is allowed','Off'
2,510,'allow_write_client_file','Controls whether the use of writing data to the client machine is allowed','Off'
2,511,'priority','Controls the priority level of current connection','normal'
2,512,'max_priority','Controls the maximum priority level a connection can have','normal'
'

DB Files'
0,0,'system','d:\\tpcc\\tpcc.db',
15,15,'temporary','j:\\temp\\sqla0000.tmp',
1,1,'cs_space','j:\\tpcc\\cs.dbs',
2,2,'misc_space','m:\\tpcc\\misc.dbs',
'

Options'
'DBA','collect_statistics_on_dml_updates','off'
'DBA','update_statistics','off'
'PUBLIC','allow_nulls_by_default','On'
'PUBLIC','allow_read_client_file','Off'
'PUBLIC','allow_snapshot_isolation','Off'
'PUBLIC','allow_write_client_file','Off'
'PUBLIC','ansi_blanks','Off'
'PUBLIC','ansi_close_cursors_on_rollback','Off'
'PUBLIC','ansi_permissions','On'
'PUBLIC','ansi_update_constraints','Cursors'
'PUBLIC','ansinull','On'
'PUBLIC','assume_distinct_servers','Off'
'PUBLIC','auditing','Off'
'PUBLIC','auditing_options','4294967295'
'PUBLIC','background_priority','Off'
'PUBLIC','blob_threshold','256'
'PUBLIC','blocking','On'
'PUBLIC','blocking_timeout','0'
'PUBLIC','chained','On'
'PUBLIC','checkpoint_time','60'
'PUBLIC','cis_option','0'
'PUBLIC','cis_rowset_size','50'
'PUBLIC','close_on_endtrans','On'
'PUBLIC','collect_statistics_on_dml_updates','On'
'PUBLIC','compression','6'
'PUBLIC','conn_auditing','On'
'PUBLIC','connection_authentication',''
'PUBLIC','continue_after_raiserror','On'
'PUBLIC','conversion_error','On'
'PUBLIC','cooperative_commit_timeout','250'
'PUBLIC','cooperative_commits','On'
'PUBLIC','database_authentication',''

'PUBLIC','date_format','YYYY-MM-DD'
'PUBLIC','date_order','YMD'
'PUBLIC','debug_messages','Off'
'PUBLIC','dedicated_task','Off'
'PUBLIC','default_dbspace',''
'PUBLIC','default_timestamp_increment','1'
'PUBLIC','delayed_commit_timeout','500'
'PUBLIC','delayed_commits','Off'
'PUBLIC','delete_old_logs','Off'
'PUBLIC','escape_character','On'
'PUBLIC','exclude_operators',''
'PUBLIC','extended_join_syntax','On'
'PUBLIC','external_remote_options','Off'
'PUBLIC','fire_triggers','On'
'PUBLIC','first_day_of_week','7'
'PUBLIC','for_xml_null_treatment','Omit'
'PUBLIC','force_view_creation','Off'
'PUBLIC','global_database_id','2147483647'
'PUBLIC','http_session_timeout','30'
'PUBLIC','integrated_server_name',''
'PUBLIC','isolation_level','0'
'PUBLIC','java_location',''
'PUBLIC','java_main_userid',''
'PUBLIC','java_vm_options',''
'PUBLIC','lock_rejected_rows','Off'
'PUBLIC','log_deadlocks','Off'
'PUBLIC','login_mode','Standard'
'PUBLIC','login_procedure','sp_login_environment'
'PUBLIC','materialized_view_optimization','Stale'
'PUBLIC','max_client_statements_cached','10'
'PUBLIC','max_cursor_count','50'
'PUBLIC','max_hash_size','10'
'PUBLIC','max_plans_cached','100'
'PUBLIC','max_priority','normal'
'PUBLIC','max_query_tasks','0'
'PUBLIC','max_recursive_iterations','100'
'PUBLIC','max_statement_count','50'
'PUBLIC','max_temp_space','0'
'PUBLIC','min_password_length','0'
'PUBLIC','ml_remote_id',''
'PUBLIC','nearest_century','50'
'PUBLIC','non_keywords',''
'PUBLIC','odbc_describe_binary_as_varbinary','Off'
'PUBLIC','odbc_distinguish_char_and_varchar','Off'
'PUBLIC','oem_string',''
'PUBLIC','on_charset_conversion_failure','Ignore'
'PUBLIC','on_tsql_error','Conditional'
'PUBLIC','optimization_goal','All-rows'
'PUBLIC','optimization_level','9'
'PUBLIC','optimization_workload','Mixed'
'PUBLIC','pinned_cursor_percent_of_cache','10'
'PUBLIC','post_login_procedure','dbo.sa_post_login_procedure'
'PUBLIC','precision','30'
'PUBLIC','prefetch','Conditional'
'PUBLIC','preserve_source_format','On'
'PUBLIC','prevent_article_pkey_update','On'
'PUBLIC','priority','normal'
'PUBLIC','qualify_owners','On'
'PUBLIC','query_mem_timeout','-1'
'PUBLIC','quote_all_identifiers','Off'
'PUBLIC','quoted_identifier','On'
'PUBLIC','read_past_deleted','On'
'PUBLIC','recovery_time','2'
'PUBLIC','remote_idle_timeout','15'
'PUBLIC','replicate_all','Off'
'PUBLIC','replication_error',''
'PUBLIC','replication_error_piece',''

'PUBLIC','request_timeout','0'
'PUBLIC','return_date_time_as_string','Off'
'PUBLIC','rollback_on_deadlock','On'
'PUBLIC','row_counts','Off'
'PUBLIC','save_remote_passwords','On'
'PUBLIC','scale','6'
'PUBLIC','secure_feature_key',''
'PUBLIC','sort_collation','Internal'
'PUBLIC','sql_flagger_error_level','W'
'PUBLIC','sql_flagger_warning_level','W'
'PUBLIC','sr_date_format','yyyy/mm/dd'
'PUBLIC','sr_time_format','hh:nn:ss.Ssssss'
'PUBLIC','sr_timestamp_format','hh:nn:ss.Ssssss yyyy/mm/dd'
'PUBLIC','string_rtruncation','On'
'PUBLIC','subscribe_by_remote','On'
'PUBLIC','subsume_row_locks','On'
'PUBLIC','suppress_tds_debugging','Off'
'PUBLIC','synchronize_mirror_on_commit','Off'
'PUBLIC','tds_empty_string_is_null','Off'
'PUBLIC','temp_space_limit_check','On'
'PUBLIC','time_format','HH:NN:SS.SSS'
'PUBLIC','time_zone_adjustment','0'
'PUBLIC','timestamp_format','YYYY-MM-DD HH:NN:SS.SSS'
'PUBLIC','truncate_timestamp_values','Off'
'PUBLIC','tsql_outer_joins','Off'
'PUBLIC','tsql_variables','Off'
'PUBLIC','updatable_statement_isolation','0'
'PUBLIC','update_statistics','On'
'PUBLIC','user_estimates','Override-magic'
'PUBLIC','verify_all_columns','Off'
'PUBLIC','verify_password_function',''
'PUBLIC','verify_threshold','1000'
'PUBLIC','wait_for_commit','Off'
'PUBLIC','webservice_namespace_host',''
""
""
""
""
""
""
""
""
""
""
""
ï»¿<?xml version ="1.0"?>
<configuration xmlns="urn:microsoft-catalog:XML_Metabase_V64_0">
<MBProperty>
<IIS_Global Location ="."
BINSchemaTimeStamp="32dcde1bdbc4c801"
ChangeNumber="3015"
HistoryMajorVersionNumber="376"

SessionKey="49634b62980000004c0000004000000001020000168000000a40000584239f5ea99532f69681360415c4cc39
2af3d6d16936d917a5cffe38b1bee210482cb1d2643468a7be5dcda5a167673f6e9141da8ee8c0aa76c66d89eea428d00000000b2fb9e3
601f7dfbb38fa434162a7cd3a824bb3c4262977c970fd016e088418b8de375b2bd57e394067dfcb998a8aeb260eb253f699b8cbe2d9701
bfb0b87623e"
XMLSchemaTimeStamp="32dcde1bdbc4c801"
>
</IIS_Global>
<IIS_ROOT Location ="/"

AdminACL="4963446210010000c800000040000000205de133b7ca0ce559fb46e0af0ddb808738d81614268a713280678621
ea23f276bab80e63298145312d199c23169b8626ae00494fccd1e8fd8439f0a133b6b7a76a0b2875883f88b2976f1ed2632ab799777d91
4e1140347a046133d8bdf2091c25262de83e7a37393f6a3c5f8952ed1b1b2abe434a4505350c78b2f4e8192f96cc4b21b9e18c0c7e6740
bd240d3363f36d95a250e72a7662359b78817e1011a5b9711089a794ae30c640687c48ca08a4c5406f584f33892c2655a63592cb74dca

```
92e572adf731ab31d86a151deb8aae6cb4b6193076e6915786d18b15b3efb11c7170031856d2984c4e17ddde2aa4a648783c1e3a1d06e
147715ed2e836b4377c32e3f8f567d4c"
        >
</IIS_ROOT>
<IIsComputer        Location ="/LM"
                    EnableEditWhileRunning="0"
                    EnableHistory="1"
                    MaxBandwidth="4294967295"
                    MaxHistoryFiles="10"
        >
</IIsComputer>
<IIsConfigObject    Location ="/LM/IISADMIN"
        >
</IIsConfigObject>
<IIsConfigObject    Location ="/LM/IISADMIN/EXTENSIONS"
        >
</IIsConfigObject>
<IIsConfigObject    Location ="/LM/IISADMIN/EXTENSIONS/DCOMCLSIDS"
        >
        <Custom
                    Name="MD_IISADMIN_EXTENSIONS"
                    ID="1028"
                    Value="{61738644-F196-11D0-9953-00C04FD919C1}"
                    Type="MULTISZ"
                    UserType="IIS_MD_UT_SERVER"
                    Attributes="NO_ATTRIBUTES"
        />
</IIsConfigObject>
<IIsConfigObject    Location ="/LM/IISADMIN/PROPERTYREGISTRATION"
        >
        <Custom
                    Name="MD_METADATA_ID_REGISTRATION"
                    ID="1030"
                    Value="0-65535;Microsoft Reserved
                            65536-524288;Microsoft IIS Admin Objects Reserved"
                    Type="MULTISZ"
                    UserType="IIS_MD_UT_SERVER"
                    Attributes="INHERIT"
        />
</IIsConfigObject>
<IIsLogModules      Location ="/LM/Logging"

        AdminACL="49634462a0000000580000004000000205de133b7ca0ce5c9fb46e05f0ddb808738d81614268a713280d78625
ea23f276ba840ec1298545312a199c23169b8613ae0049a9f4b7ec0074b3ad4c16fdcb68690b28558a2b883a956f1ed3622ab2b9777d9
47d13403419a79bbbca93e3788fa694f43e9f81459064fc0d06742c04c29f7040fc9c382994a9a0220e94d0bb8c137123c8fa94baf83082
092694d18ce55fcc5e0b003b75"
        >
</IIsLogModules>
<IIsCustomLogModule         Location ="/LM/Logging/Custom Logging"
                    LogCustomPropertyServicesString="W3SVC
                            MSFTPSVC
                            SMTPSVC
                            NNTPSVC"
        >
</IIsCustomLogModule>
<IIsCustomLogModule         Location ="/LM/Logging/Custom Logging/Date"
                    LogCustomPropertyDataType="6"
                    LogCustomPropertyHeader="date"
                    LogCustomPropertyID="4013"
                    LogCustomPropertyMask="1"
                    LogCustomPropertyName="Date"
                    LogCustomPropertyNodeID="1"
        >
</IIsCustomLogModule>
<IIsCustomLogModule         Location ="/LM/Logging/Custom Logging/Extended Properties"
                    LogCustomPropertyDataType="6"
                    LogCustomPropertyID="4013"
```

```
                LogCustomPropertyName="Extended Properties"
                LogCustomPropertyNodeID="3"
        >
</IIsCustomLogModule>
<IIsCustomLogModule         Location ="/LM/Logging/Custom Logging/Extended Properties/Bytes Received"
                LogCustomPropertyDataType="3"
                LogCustomPropertyHeader="cs-bytes"
                LogCustomPropertyMask="8192"
                LogCustomPropertyName="Bytes Received"
                LogCustomPropertyNodeID="17"
        >
</IIsCustomLogModule>
<IIsCustomLogModule         Location ="/LM/Logging/Custom Logging/Extended Properties/Bytes Sent"
                LogCustomPropertyDataType="3"
                LogCustomPropertyHeader="sc-bytes"
                LogCustomPropertyMask="4096"
                LogCustomPropertyName="Bytes Sent"
                LogCustomPropertyNodeID="16"
        >
</IIsCustomLogModule>
<IIsCustomLogModule         Location ="/LM/Logging/Custom Logging/Extended Properties/Client IP Address"
                LogCustomPropertyDataType="6"
                LogCustomPropertyHeader="c-ip"
                LogCustomPropertyMask="4"
                LogCustomPropertyName="Client IP Address"
                LogCustomPropertyNodeID="5"
        >
</IIsCustomLogModule>
<IIsCustomLogModule         Location ="/LM/Logging/Custom Logging/Extended Properties/Cookie"
                LogCustomPropertyDataType="6"
                LogCustomPropertyHeader="cs(Cookie)"
                LogCustomPropertyMask="131072"
                LogCustomPropertyName="Cookie"
                LogCustomPropertyNodeID="22"
        >
</IIsCustomLogModule>
<IIsCustomLogModule         Location ="/LM/Logging/Custom Logging/Extended Properties/Host"
                LogCustomPropertyDataType="6"
                LogCustomPropertyHeader="cs-host"
                LogCustomPropertyMask="1048576"
                LogCustomPropertyName="Host"
                LogCustomPropertyNodeID="20"
        >
</IIsCustomLogModule>
<IIsCustomLogModule         Location ="/LM/Logging/Custom Logging/Extended Properties/Method"
                LogCustomPropertyDataType="6"
                LogCustomPropertyHeader="cs-method"
                LogCustomPropertyMask="128"
                LogCustomPropertyName="Method"
                LogCustomPropertyNodeID="11"
        >
</IIsCustomLogModule>
<IIsCustomLogModule         Location ="/LM/Logging/Custom Logging/Extended Properties/Protocol Status"
                LogCustomPropertyDataType="3"
                LogCustomPropertyHeader="sc-status"
                LogCustomPropertyMask="1024"
                LogCustomPropertyName="Protocol Status"
                LogCustomPropertyNodeID="14"
        >
</IIsCustomLogModule>
<IIsCustomLogModule         Location ="/LM/Logging/Custom Logging/Extended Properties/Protocol Substatus"
                LogCustomPropertyDataType="3"
                LogCustomPropertyHeader="sc-substatus"
                LogCustomPropertyMask="2097152"
                LogCustomPropertyName="Protocol Substatus"
                LogCustomPropertyNodeID="32"
        >
```

```
</IIsCustomLogModule>
<IIsCustomLogModule          Location ="/LM/Logging/Custom Logging/Extended Properties/Protocol Version"
                        LogCustomPropertyDataType="6"
                        LogCustomPropertyHeader="cs-version"
                        LogCustomPropertyMask="524288"
                        LogCustomPropertyName="Protocol Version"
                        LogCustomPropertyNodeID="19"
          >
</IIsCustomLogModule>
<IIsCustomLogModule          Location ="/LM/Logging/Custom Logging/Extended Properties/Referer"
                        LogCustomPropertyDataType="6"
                        LogCustomPropertyHeader="cs(Referer)"
                        LogCustomPropertyMask="262144"
                        LogCustomPropertyName="Referer"
                        LogCustomPropertyNodeID="23"
          >
</IIsCustomLogModule>
<IIsCustomLogModule          Location ="/LM/Logging/Custom Logging/Extended Properties/Server IP"
                        LogCustomPropertyDataType="6"
                        LogCustomPropertyHeader="s-ip"
                        LogCustomPropertyMask="64"
                        LogCustomPropertyName="Server IP Address"
                        LogCustomPropertyNodeID="9"
          >
</IIsCustomLogModule>
<IIsCustomLogModule          Location ="/LM/Logging/Custom Logging/Extended Properties/Server Name"
                        LogCustomPropertyDataType="6"
                        LogCustomPropertyHeader="s-computername"
                        LogCustomPropertyMask="32"
                        LogCustomPropertyName="Server Name"
                        LogCustomPropertyNodeID="8"
          >
</IIsCustomLogModule>
<IIsCustomLogModule          Location ="/LM/Logging/Custom Logging/Extended Properties/Server Port"
                        LogCustomPropertyDataType="3"
                        LogCustomPropertyHeader="s-port"
                        LogCustomPropertyMask="32768"
                        LogCustomPropertyName="Server Port"
                        LogCustomPropertyNodeID="10"
          >
</IIsCustomLogModule>
<IIsCustomLogModule          Location ="/LM/Logging/Custom Logging/Extended Properties/Service Name"
                        LogCustomPropertyDataType="6"
                        LogCustomPropertyHeader="s-sitename"
                        LogCustomPropertyMask="16"
                        LogCustomPropertyName="Service Name"
                        LogCustomPropertyNodeID="7"
          >
</IIsCustomLogModule>
<IIsCustomLogModule          Location ="/LM/Logging/Custom Logging/Extended Properties/Time Taken"
                        LogCustomPropertyDataType="3"
                        LogCustomPropertyHeader="time-taken"
                        LogCustomPropertyMask="16384"
                        LogCustomPropertyName="Time Taken"
                        LogCustomPropertyNodeID="18"
          >
</IIsCustomLogModule>
<IIsCustomLogModule          Location ="/LM/Logging/Custom Logging/Extended Properties/URI Query"
                        LogCustomPropertyDataType="6"
                        LogCustomPropertyHeader="cs-uri-query"
                        LogCustomPropertyMask="512"
                        LogCustomPropertyName="URI Query"
                        LogCustomPropertyNodeID="13"
          >
</IIsCustomLogModule>
<IIsCustomLogModule          Location ="/LM/Logging/Custom Logging/Extended Properties/URI Stem"
                        LogCustomPropertyDataType="6"
```

```
                    LogCustomPropertyHeader="cs-uri-stem"
                    LogCustomPropertyMask="256"
                    LogCustomPropertyName="URI Stem"
                    LogCustomPropertyNodeID="12"
          >
</IIsCustomLogModule>
<IIsCustomLogModule         Location ="/LM/Logging/Custom Logging/Extended Properties/User Agent"
                    LogCustomPropertyDataType="6"
                    LogCustomPropertyHeader="cs(User-Agent)"
                    LogCustomPropertyMask="65536"
                    LogCustomPropertyName="User Agent"
                    LogCustomPropertyNodeID="21"
          >
</IIsCustomLogModule>
<IIsCustomLogModule         Location ="/LM/Logging/Custom Logging/Extended Properties/User Name"
                    LogCustomPropertyDataType="6"
                    LogCustomPropertyHeader="cs-username"
                    LogCustomPropertyMask="8"
                    LogCustomPropertyName="User Name"
                    LogCustomPropertyNodeID="6"
          >
</IIsCustomLogModule>
<IIsCustomLogModule         Location ="/LM/Logging/Custom Logging/Extended Properties/Win32 Status"
                    LogCustomPropertyDataType="3"
                    LogCustomPropertyHeader="sc-win32-status"
                    LogCustomPropertyMask="2048"
                    LogCustomPropertyName="Win32 Status"
                    LogCustomPropertyNodeID="15"
          >
</IIsCustomLogModule>
<IIsCustomLogModule         Location ="/LM/Logging/Custom Logging/Time"
                    LogCustomPropertyDataType="6"
                    LogCustomPropertyHeader="time"
                    LogCustomPropertyID="4013"
                    LogCustomPropertyMask="2"
                    LogCustomPropertyName="Time"
                    LogCustomPropertyNodeID="2"
          >
</IIsCustomLogModule>
<IIsLogModule       Location ="/LM/Logging/Microsoft IIS Log File Format"
                    LogModuleId="{FF160657-DE82-11CF-BC0A-00AA006111E0}"
                    LogModuleUiId="{31DCAB87-BB3E-11d0-9299-00C04FB6678B}"
          >
</IIsLogModule>
<IIsLogModule       Location ="/LM/Logging/NCSA Common Log File Format"
                    LogModuleId="{FF16065F-DE82-11CF-BC0A-00AA006111E0}"
                    LogModuleUiId="{31DCAB85-BB3E-11d0-9299-00C04FB6678B}"
          >
</IIsLogModule>
<IIsLogModule       Location ="/LM/Logging/ODBC Logging"
                    LogModuleId="{FF16065B-DE82-11CF-BC0A-00AA006111E0}"
                    LogModuleUiId="{31DCAB86-BB3E-11d0-9299-00C04FB6678B}"
          >
</IIsLogModule>
<IIsLogModule       Location ="/LM/Logging/W3C Extended Log File Format"
                    LogModuleId="{FF160663-DE82-11CF-BC0A-00AA006111E0}"
                    LogModuleUiId="{31DCAB88-BB3E-11d0-9299-00C04FB6678B}"
          >
</IIsLogModule>
<IIsMimeMap         Location ="/LM/MimeMap"
                    MimeMap=".asx,video/x-ms-asf
                              .xml,text/xml
                              .tsv,text/tab-separated-values
                              .ra,audio/x-pn-realaudio
                              .sv4crc,application/x-sv4crc
                              .spc,application/x-pkcs7-certificates
                              .pmc,application/x-perfmon
```

.lit,application/x-ms-reader
.crd,application/x-mscardfile
.isp,application/x-internet-signup
.wmlsc,application/vnd.wap.wmlscriptc
.vst,application/vnd.visio
.xlam,application/vnd.ms-excel.addin.macroEnabled.12
.ttf,application/octet-stream
.pfm,application/octet-stream
.csv,application/octet-stream
.aaf,application/octet-stream
.one,application/onenote
.hta,application/hta
.atom,application/atom+xml
.323,text/h323
.mhtml,message/rfc822
.midi,audio/mid
.p7r,application/x-pkcs7-certreqresp
.mny,application/x-msmoney
.clp,application/x-msclip
.vsd,application/vnd.visio
.lpk,application/octet-stream
.bin,application/octet-stream
.onetoc,application/onenote
.x,application/directx
.wvx,video/x-ms-wvx
.vcf,text/x-vcard
.htc,text/x-component
.htt,text/webviewhtml
.h,text/plain
.mht,message/rfc822
.mid,audio/mid
.p7b,application/x-pkcs7-certificates
.gz,application/x-gzip
.dvi,application/x-dvi
.cpio,application/x-cpio
.vdx,application/vnd.ms-visio.viewer
.sldm,application/vnd.ms-powerpoint.slide.macroEnabled.12
.xlm,application/vnd.ms-excel
.fdf,application/vnd.fdf
.setreg,application/set-registration-initiation
.eps,application/postscript
.p7s,application/pkcs7-signature
.toc,application/octet-stream
.mdp,application/octet-stream
.ics,application/octet-stream
.chm,application/octet-stream
.asi,application/octet-stream
.afm,application/octet-stream
.evy,application/envoy
.wmp,video/x-ms-wmp
.qt,video/quicktime
.mpv2,video/mpeg
.xslt,text/xml
.etx,text/x-setext
.cod,image/cis-cod
.snd,audio/basic
.au,audio/basic
.man,application/x-troff-man
.qtl,application/x-quicktimeplayer
.pmw,application/x-perfmon
.class,application/x-java-applet
.iii,application/x-iphone
.csh,application/x-csh
.z,application/x-compress
.vtx,application/vnd.visio
.vsw,application/vnd.visio
.wps,application/vnd.ms-works

.potx,application/vnd.openxmlformats-officedocument.presentationml.template
.ps,application/postscript
.p7c,application/pkcs7-mime
.thn,application/octet-stream
.mso,application/octet-stream
.dot,application/msword
.doc,application/msword
.sgml,text/sgml
.nws,message/rfc822
.pbm,image/x-portable-bitmap
.ief,image/ief
.wav,audio/wav
.texi,application/x-texinfo
.mvb,application/x-msmediaview
.hdf,application/x-hdf
.vsx,application/vnd.visio
.dotm,application/vnd.ms-word.template.macroEnabled.12
.docm,application/vnd.ms-word.document.macroEnabled.12
.pptx,application/vnd.openxmlformats-officedocument.presentationml.presentation
.psm,application/octet-stream
.java,application/octet-stream
.eot,application/octet-stream
.jar,application/java-archive
.mpeg,video/mpeg
.xsf,text/xml
.map,text/plain
.uls,text/iuls
.rf,image/vnd.rn-realflash
.m3u,audio/x-mpegurl
.wma,audio/x-ms-wma
.aifc,audio/aiff
.mdb,application/x-msaccess
.mvc,application/x-miva-compiled
.stl,application/vnd.ms-pki.stl
.ppsx,application/vnd.openxmlformats-officedocument.presentationml.slideshow
.xlsb,application/vnd.ms-excel.sheet.binary.macroEnabled.12
.setpay,application/set-payment-initiation
.prm,application/octet-stream
.mix,application/octet-stream
.lzh,application/octet-stream
.hhk,application/octet-stream
.onepkg,application/onenote
.xaf,x-world/x-vrml
.flr,x-world/x-vrml
.IVF,video/x-ivf
.cnf,text/plain
.asm,text/plain
.tiff,image/tiff
.wax,audio/x-ms-wax
.ms,application/x-troff-ms
.tcl,application/x-tcl
.shar,application/x-shar
.sh,application/x-sh
.nc,application/x-netcdf
.hlp,application/winhlp
.oda,application/oda
.pfb,application/octet-stream
.fla,application/octet-stream
.wm,video/x-ms-wm
.rgb,image/x-rgb
.ppm,image/x-portable-pixmap
.ram,audio/x-pn-realaudio
.sit,application/x-stuffit
.dir,application/x-director
.mpp,application/vnd.ms-project
.xla,application/vnd.ms-excel
.ssm,application/streamingmedia

.axs,application/olescript
.ods,application/oleobject
.psp,application/octet-stream
.jpb,application/octet-stream
.wrz,x-world/x-vrml
.m1v,video/mpeg
.mno,text/xml
.cmx,image/x-cmx
.jpeg,image/jpeg
.dib,image/bmp
.rmi,audio/mid
.aiff,audio/aiff
.wmd,application/x-ms-wmd
.wri,application/x-mswrite
.pub,application/x-mspublisher
.ins,application/x-internet-signup
.wks,application/vnd.ms-works
.xls,application/vnd.ms-excel
.ai,application/postscript
.crl,application/pkix-crl
.qxd,application/octet-stream
.dwp,application/octet-stream
.xof,x-world/x-vrml
.wmv,video/x-ms-wmv
.nsc,video/x-ms-asf
.mpa,video/mpeg
.pnm,image/x-portable-anymap
.rpm,audio/x-pn-realaudio-plugin
.aif,audio/x-aiff
.me,application/x-troff-me
.pml,application/x-perfmon
.trm,application/x-msterminal
.m13,application/x-msmediaview
.js,application/x-javascript
.dxr,application/x-director
.potm,application/vnd.ms-powerpoint.template.macroEnabled.12
.xltx,application/vnd.openxmlformats-officedocument.spreadsheetml.template
.xlt,application/vnd.ms-excel
.xlc,application/vnd.ms-excel
.p10,application/pkcs10
.smi,application/octet-stream
.sea,application/octet-stream
.hqx,application/mac-binhex40
.spl,application/futuresplash
.movie,video/x-sgi-movie
.lsf,video/x-la-asf
.txt,text/plain
.jfif,image/pjpeg
.jpe,image/jpeg
.zip,application/x-zip-compressed
.wmf,application/x-msmetafile
.m14,application/x-msmediaview
.latex,application/x-latex
.wcm,application/vnd.ms-works
.pptm,application/vnd.ms-powerpoint.presentation.macroEnabled.12
.xlsx,application/vnd.openxmlformats-officedocument.spreadsheetml.sheet
.hhp,application/octet-stream
.aca,application/octet-stream
.accdb,application/msaccess
.jcz,application/liquidmotion
.wrl,x-world/x-vrml
.wmx,video/x-ms-wmx
.asr,video/x-ms-asf
.lsx,video/x-la-asf
.xsl,text/xml
.html,text/html
.tif,image/tiff

.der,application/x-x509-ca-cert
.pfx,application/x-pkcs12
.p12,application/x-pkcs12
.ppsm,application/vnd.ms-powerpoint.slideshow.macroEnabled.12
.cur,application/octet-stream
.accdt,application/msaccess
.hdml,text/x-hdml
.htm,text/html
.xbm,image/x-xbitmap
.jpg,image/jpeg
.texinfo,application/x-texinfo
.ppam,application/vnd.ms-powerpoint.addin.macroEnabled.12
.xlw,application/vnd.ms-excel
.rm,application/vnd.rn-realmedia
.pdf,application/pdf
.rar,application/octet-stream
.psd,application/octet-stream
.inf,application/octet-stream
.emz,application/octet-stream
.dsp,application/octet-stream
.onea,application/onenote
.jck,application/liquidmotion
.mpe,video/mpeg
.mp2,video/mpeg
.sct,text/scriptlet
.ras,image/x-cmu-raster
.swf,application/x-shockwave-flash
.wmz,application/x-ms-wmz
.gtar,application/x-gtar
.dcr,application/x-director
.sldx,application/vnd.openxmlformats-officedocument.presentationml.slide
.pps,application/vnd.ms-pps
.p7m,application/pkcs7-mime
.xsn,application/octet-stream
.ocx,application/octet-stream
.accde,application/msaccess
.mov,video/quicktime
.wmls,text/vnd.wap.wmlscript
.cpp,text/plain
.c,text/plain
.bas,text/plain
.css,text/css
.art,image/x-jg
.mp3,audio/mpeg
.t,application/x-troff
.roff,application/x-troff
.tar,application/x-tar
.hhc,application/x-oleobject
.scd,application/x-msschedule
.pko,application/vnd.ms-pki.pko
.sst,application/vnd.ms-pki.certstore
.ppt,application/vnd.ms-powerpoint
.xtp,application/octet-stream
.u32,application/octet-stream
.pcx,application/octet-stream
.msi,application/octet-stream
.exe,application/octet-stream
.asd,application/octet-stream
.onetoc2,application/onenote
.fif,application/fractals
.mpg,video/mpeg
.vml,text/xml
.xdr,text/plain
.vcs,text/plain
.hxt,text/html
.eml,message/rfc822
.xpm,image/x-xpixmap

```
                        .ico,image/x-icon
                        .gif,image/gif
                        .dwf,drawing/x-dwf
                        .src,application/x-wais-source
                        .tr,application/x-troff
                        .pmr,application/x-perfmon
                        .pma,application/x-perfmon
                        .dll,application/x-msdownload
                        .bcpio,application/x-bcpio
                        .wmlc,application/vnd.wap.wmlc
                        .wdb,application/vnd.ms-works
                        .dotx,application/vnd.openxmlformats-officedocument.wordprocessingml.template
                        .docx,application/vnd.openxmlformats-officedocument.wordprocessingml.document
                        .pot,application/vnd.ms-powerpoint
                        .xltm,application/vnd.ms-excel.template.macroEnabled.12
                        .rtf,application/rtf
                        .prf,application/pics-rules
                        .snp,application/octet-stream
                        .cab,application/octet-stream
                        .avi,video/x-msvideo
                        .asf,video/x-ms-asf
                        .dtd,text/xml
                        .wml,text/vnd.wap.wml
                        .vbs,text/vbscript
                        .rtx,text/richtext
                        .dlm,text/dlm
                        .xwd,image/x-xwindowdump
                        .pgm,image/x-portable-graymap
                        .bmp,image/bmp
                        .crt,application/x-x509-ca-cert
                        .ustar,application/x-ustar
                        .tex,application/x-tex
                        .sv4cpio,application/x-sv4cpio
                        .tgz,application/x-compressed
                        .cdf,application/x-cdf
                        .vss,application/vnd.visio
                        .cat,application/vnd.ms-pki.seccat
                        .thmx,application/vnd.ms-officetheme
                        .xlsm,application/vnd.ms-excel.sheet.macroEnabled.12
                        .prx,application/octet-stream
                        .pcz,application/octet-stream
                        .onetmp,application/onenote
                        .acx,application/internet-property-stream
                        .wsdl,text/xml
                        .disco,text/xml
                        .xsd,text/xml
                        .wbmp,image/vnd.wap.wbmp
                        .png,image/png
                        .pnz,image/png
                        .smd,audio/x-smd
                        .smz,audio/x-smd
                        .smx,audio/x-smd
                        .mmf,application/x-smaf
                        .application,application/x-ms-application
                        .manifest,application/x-ms-manifest
                        .deploy,application/octet-stream"
        >
</IIsMimeMap>
<IIsWebService      Location ="/LM/W3SVC"
                    AllowKeepAlive="TRUE"
                    AnonymousUserName="IUSR_NEWTON"

        AnonymousUserPass="4963446270000000220000004000000205de133e6ca3f65b9fb72e0690db380b338eb166026d8710
f809d8675ea48f276ba000001000000573def67395b4db1722ce765c68ba9cf4aa9b7ccd2a89fef87a28195bf1ff636697251e7adf719a7e
89028b9761c37778c9987627e2fdf451efc90777f44b128"
                    AppAllowClientDebug="FALSE"
                    AppAllowDebugging="FALSE"
```

AppPoolId="DefaultAppPool"
AppPoolId="DefaultAppPool"
ApplicationDependencies="Active Server Pages;ASP
Internet Data Connector;HTTPODBC
Server Side Includes;SSINC
WebDAV;WEBDAV
ASP.NET v2.0.50727 (32-bit);ASP.NET v2.0.50727 (32-bit)"
AspAllowOutOfProcComponents="TRUE"
AspAllowSessionState="TRUE"
AspAppServiceFlags="0"
AspBufferingLimit="4194304"
AspBufferingOn="TRUE"
AspCalcLineNumber="TRUE"
AspCodepage="0"
AspDiskTemplateCacheDirectory="%windir%\system32\inetsrv\ASP Compiled Templates"
AspEnableApplicationRestart="TRUE"
AspEnableAspHtmlFallback="FALSE"
AspEnableChunkedEncoding="TRUE"
AspEnableParentPaths="FALSE"
AspEnableTypelibCache="TRUE"
AspErrorsToNTLog="FALSE"
AspExceptionCatchEnable="TRUE"
AspExecuteInMTA="0"
AspKeepSessionIDSecure="0"
AspLCID="2048"
AspLogErrorRequests="TRUE"
AspMaxDiskTemplateCacheFiles="2000"
AspMaxRequestEntityAllowed="204800"
AspProcessorThreadMax="25"
AspQueueConnectionTestTime="3"
AspQueueTimeout="4294967295"
AspRequestQueueMax="3000"
AspRunOnEndAnonymously="TRUE"
AspScriptEngineCacheMax="250"
AspScriptErrorMessage="An error occurred on the server when processing the URL. Please contact the system administrator."
AspScriptErrorSentToBrowser="TRUE"
AspScriptFileCacheSize="500"
AspScriptLanguage="VBScript"
AspScriptTimeout="90"
AspSessionMax="4294967295"
AspSessionTimeout="20"
AspTrackThreadingModel="FALSE"
AuthChangeURL="/iisadmpwd/achg.asp"
AuthExpiredURL="/iisadmpwd/aexp.asp"
AuthExpiredUnsecureURL="/iisadmpwd/aexp3.asp"
AuthFlags="AuthAnonymous"
AuthNotifyPwdExpURL="/iisadmpwd/anot.asp"
AuthNotifyPwdExpUnsecureURL="/iisadmpwd/anot3.asp"
CGITimeout="300"
CacheISAPI="TRUE"
CentralBinaryLoggingEnabled="FALSE"
ConnectionTimeout="300"
ContentIndexed="TRUE"
DefaultDoc="Default.aspx,Default.htm,Default.asp,index.htm"
DirBrowseFlags="DirBrowseShowDate | DirBrowseShowTime | DirBrowseShowSize | DirBrowseShowExtension | DirBrowseShowLongDate | EnableDefaultDoc"
DownlevelAdminInstance="1"
HttpCustomHeaders="X-Powered-By: ASP.NET"
HttpErrors="400,*,FILE,C:\WINDOWS\help\iisHelp\common\400.htm
401,1,FILE,C:\WINDOWS\help\iisHelp\common\401-1.htm
401,2,FILE,C:\WINDOWS\help\iisHelp\common\401-2.htm
401,3,FILE,C:\WINDOWS\help\iisHelp\common\401-3.htm
401,4,FILE,C:\WINDOWS\help\iisHelp\common\401-4.htm
401,5,FILE,C:\WINDOWS\help\iisHelp\common\401-5.htm
401,7,FILE,C:\WINDOWS\help\iisHelp\common\401-1.htm
403,1,FILE,C:\WINDOWS\help\iisHelp\common\403-1.htm
403,2,FILE,C:\WINDOWS\help\iisHelp\common\403-2.htm

```
                    403,3,FILE,C:\WINDOWS\help\iisHelp\common\403-3.htm
                    403,4,FILE,C:\WINDOWS\help\iisHelp\common\403-4.htm
                    403,5,FILE,C:\WINDOWS\help\iisHelp\common\403-5.htm
                    403,6,FILE,C:\WINDOWS\help\iisHelp\common\403-6.htm
                    403,7,FILE,C:\WINDOWS\help\iisHelp\common\403-7.htm
                    403,8,FILE,C:\WINDOWS\help\iisHelp\common\403-8.htm
                    403,9,FILE,C:\WINDOWS\help\iisHelp\common\403-9.htm
                    403,10,FILE,C:\WINDOWS\help\iisHelp\common\403-10.htm
                    403,11,FILE,C:\WINDOWS\help\iisHelp\common\403-11.htm
                    403,12,FILE,C:\WINDOWS\help\iisHelp\common\403-12.htm
                    403,13,FILE,C:\WINDOWS\help\iisHelp\common\403-13.htm
                    403,15,FILE,C:\WINDOWS\help\iisHelp\common\403-15.htm
                    403,16,FILE,C:\WINDOWS\help\iisHelp\common\403-16.htm
                    403,17,FILE,C:\WINDOWS\help\iisHelp\common\403-17.htm
                    403,18,FILE,C:\WINDOWS\help\iisHelp\common\403.htm
                    403,19,FILE,C:\WINDOWS\help\iisHelp\common\403.htm
                    403,20,FILE,C:\WINDOWS\help\iisHelp\common\403-20.htm
                    404,*,FILE,C:\WINDOWS\help\iisHelp\common\404b.htm
                    404,2,FILE,C:\WINDOWS\help\iisHelp\common\404b.htm
                    404,3,FILE,C:\WINDOWS\help\iisHelp\common\404b.htm
                    405,*,FILE,C:\WINDOWS\help\iisHelp\common\405.htm
                    406,*,FILE,C:\WINDOWS\help\iisHelp\common\406.htm
                    407,*,FILE,C:\WINDOWS\help\iisHelp\common\407.htm
                    412,*,FILE,C:\WINDOWS\help\iisHelp\common\412.htm
                    414,*,FILE,C:\WINDOWS\help\iisHelp\common\414.htm
                    415,*,FILE,C:\WINDOWS\help\iisHelp\common\415.htm
                    500,12,FILE,C:\WINDOWS\help\iisHelp\common\500-12.htm
                    500,13,FILE,C:\WINDOWS\help\iisHelp\common\500-13.htm
                    500,15,FILE,C:\WINDOWS\help\iisHelp\common\500-15.htm
                    500,16,FILE,C:\WINDOWS\help\iisHelp\common\500.htm
                    500,17,FILE,C:\WINDOWS\help\iisHelp\common\500.htm
                    500,18,FILE,C:\WINDOWS\help\iisHelp\common\500.htm
                    500,19,FILE,C:\WINDOWS\help\iisHelp\common\500.htm"
            IIs5IsolationModeEnabled="FALSE"
            InProcessIsapiApps="C:\WINDOWS\system32\inetsrv\httpext.dll
                    C:\WINDOWS\system32\inetsrv\httpodbc.dll
                    C:\WINDOWS\system32\inetsrv\ssinc.dll
                    C:\WINDOWS\system32\msw3prt.dll
                    C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll"
            LogExtFileFlags="LogExtFileDate | LogExtFileTime | LogExtFileClientIp | LogExtFileUserName |
LogExtFileSiteName | LogExtFileServerIp | LogExtFileMethod | LogExtFileUriStem | LogExtFileUriQuery | LogExtFileHttpStatus |
LogExtFileWin32Status | LogExtFileServerPort | LogExtFileUserAgent | LogExtFileHttpSubStatus"
            LogFileDirectory="C:\WINDOWS\system32\LogFiles"
            LogFilePeriod="1"
            LogFileTruncateSize="20971520"
            LogInUTF8="FALSE"
            LogOdbcDataSource="HTTPLOG"

        LogOdbcPassword="4963446260000000120000004000000205de133c5ca7a6591fb2ae0740dbc80873810008000000f50
40f08900c51ee3fd6c0363cad3cb7aa641de4115a553ee23d53826a911bf5c23cb65adf255f31ea60b6395ad016afde13b72c99d9d05ca1
7756f9d3b896b0"
            LogOdbcTableName="InternetLog"
            LogOdbcUserName="InternetAdmin"
            LogPluginClsid="{FF160663-DE82-11CF-BC0A-00AA006111E0}"
            LogType="0"
            MaxConnections="4294967295"
            MaxGlobalBandwidth="4294967295"
            MinFileBytesPerSec="240"
            PasswordChangeFlags="AuthChangeDisable | AuthAdvNotifyDisable"
            ScriptMaps=".asp,C:\WINDOWS\system32\inetsrv\asp.dll,5,GET,HEAD,POST,TRACE
                    .cer,C:\WINDOWS\system32\inetsrv\asp.dll,5,GET,HEAD,POST,TRACE
                    .cdx,C:\WINDOWS\system32\inetsrv\asp.dll,5,GET,HEAD,POST,TRACE
                    .asa,C:\WINDOWS\system32\inetsrv\asp.dll,5,GET,HEAD,POST,TRACE
                    .idc,C:\WINDOWS\system32\inetsrv\httpodbc.dll,5,GET,POST
                    .shtm,C:\WINDOWS\system32\inetsrv\ssinc.dll,5,GET,POST
                    .shtml,C:\WINDOWS\system32\inetsrv\ssinc.dll,5,GET,POST
                    .stm,C:\WINDOWS\system32\inetsrv\ssinc.dll,5,GET,POST
```

.asax,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
.ascx,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
.ashx,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,1,GET,HEAD,POST,DEBUG

.asmx,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,1,GET,HEAD,POST,DEBUG
.aspx,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,1,GET,HEAD,POST,DEBUG
.axd,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,1,GET,HEAD,POST,DEBUG

.vsdisco,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,1,GET,HEAD,POST,DEBUG
.rem,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,1,GET,HEAD,POST,DEBUG

.soap,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,1,GET,HEAD,POST,DEBUG

.config,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
.cs,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

.csproj,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
.vb,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

.vbproj,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

.webinfo,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
.licx,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
.resx,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

.resources,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

.master,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
.skin,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

.compiled,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

.browser,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
.mdb,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
.jsl,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

.vjsproj,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

.sitemap,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

.msgx,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,1,GET,HEAD,POST,DEBUG
.ad,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
.dd,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
.ldd,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
.sd,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
.cd,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

.adprototype,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

.lddprototype,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
.sdm,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

.sdmDocument,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
.ldb,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
.mdf,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
.ldf,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
.java,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

.exclude,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

.refresh,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
.svc,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,1,GET,HEAD,POST,DEBUG

.xoml,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,1,GET,HEAD,POST,DEBUG

.rules,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,1,GET,HEAD,POST,DEBUG"
WAMUserName="IWAM_NEWTON"

```
        WAMUserPass="4963446270000000220000004000000205de133d5ca4665aafb10e0720d9f80e038a9166d26c8711a80c3
8654ea4cf276ba00000100000005707fded39fa923ae0a062a107f49886f58bfd4c3aa92fefd0e916636207174b7927125987924260d8e2
94417289148d18cd9b1bad6fc9f3b845eca6bfc8701"
                WebSvcExtRestrictionList="0,C:\WINDOWS\system32\inetsrv\httpodbc.dll,0,HTTPODBC,Internet Data
Connector
                        0,C:\WINDOWS\system32\inetsrv\ssinc.dll,0,SSINC,Server Side Includes
                        0,C:\WINDOWS\system32\inetsrv\asp.dll,0,ASP,Active Server Pages
                        0,*.exe
                        0,C:\WINDOWS\system32\inetsrv\httpext.dll,0,WEBDAV,WebDAV
                        1,*.dll
                        0,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,0,ASP.NET v2.0.50727 (32-
bit),ASP.NET v2.0.50727 (32-bit)"
        >
        <Custom
                Name="MD_ISM_ACCESS_CHECK"
                ID="6269"
                Value="65535"
                Type="DWORD"
                UserType="IIS_MD_UT_FILE"
                Attributes="NO_ATTRIBUTES"
        />
        <Custom
                Name="UnknownName_2166"
                ID="2166"
                Value="1"
                Type="DWORD"
                UserType="IIS_MD_UT_SERVER"
                Attributes="NO_ATTRIBUTES"
        />
        <Custom
                Name="UnknownName_9202"
                ID="9202"
                Value="4294967295"
                Type="DWORD"
                UserType="IIS_MD_UT_SERVER"
                Attributes="NO_ATTRIBUTES"
        />
</IIsWebService>
<IIsWebServer   Location ="/LM/W3SVC/87368199"
                AuthFlags="0"
                ConnectionTimeout="450"
                LogPluginClsid="{FF160663-DE82-11CF-BC0A-00AA006111E0}"
                MaxConnections="20000"
                ServerAutoStart="TRUE"
                ServerBindings=":80:"
                ServerComment="tpcc"
        >
</IIsWebServer>
<IIsFilters     Location ="/LM/W3SVC/87368199/filters"

        AdminACL="49634462f0000000a400000040000000205de133b7ca0ce57dfb46e08b0ddb808738d81614268a7132809b862
0ea23f276ba880ee12981453129199c23169b8620ae004989f6a3ec8a76b3ad4d17fdce48690b2d61882b883b974b1e58622ab798727d
945d1140316f0475339787940de1d6ac70051b314bc03c6a3c5f896eed391b2ebe434d4505350c78b2c1e8192f70f42d2544110651934
20bc1e90e3363d36f8da2fae52e7663379b7da17e1014a5bb711000000000e8fc550ebc8727b243ab4cb24a539ea3a3df2cca372eab2c7
28165478b1c51d3c813aa43f17139a9c4d8144875e1c39ac1585d41f2f453cee1b93dde795b49cb"
        >
</IIsFilters>
<IIsWebVirtualDir   Location ="/LM/W3SVC/87368199/root"
                AccessFlags="AccessExecute | AccessRead | AccessScript"
                AnonymousUserName="NEWTON\Administrator"

        AnonymousUserPass="49634462680000001a00000040000000205de133fbca4f65ccfb76e02b0deb80e338bd166c26e67130
800f08900c51eea6e0a481a7f83fab2752ba2636d44d5181caf9967dc2a0ee60efad0bc22f4089f2c974d700e72b76127439da12e410f73
0a1b544f24d4f45c0b70962dd01d90e"
                AppFriendlyName="Default Application"
                AppIsolated="2"
```

```
                AppRoot="/LM/W3SVC/87368199/Root"
                AuthFlags="AuthAnonymous"
                ContentIndexed="FALSE"
                DirBrowseFlags="DirBrowseShowDate | DirBrowseShowTime | DirBrowseShowSize | DirBrowseShowExtension |
DirBrowseShowLongDate | EnableDefaultDoc"
                DontLog="TRUE"
                Path="c:\tpcc"

        UNCPassword="4963446250000000060000004000000205de133b6ca0000983bcd6edad5a09444605bf85853beea2aee8b
1812e7e6a7a72d4b4e3df83c1148d9652930869b0d222fd5d37d35f9f505d3f3cbd9db23f049faf18b45b344b1"
        >
</IIsWebVirtualDir>
<IIsWebDirectory   Location ="/LM/W3SVC/87368199/root/aspnet_client"
                AccessFlags="AccessRead"
                DirBrowseFlags="0"
        >
</IIsWebDirectory>
<IIsWebVirtualDir   Location ="/LM/W3SVC/87368199/root/tpcc"
                AccessFlags="AccessExecute | AccessRead | AccessScript"
                AppFriendlyName="tpcc"
                AppIsolated="2"
                AppRoot="/LM/W3SVC/87368199/Root/tpcc"
                DirBrowseFlags="DirBrowseShowDate | DirBrowseShowTime | DirBrowseShowSize | DirBrowseShowExtension |
DirBrowseShowLongDate | EnableDefaultDoc"
                Path="c:\tpcc"
                ScriptMaps="*,F:\tpcc\kit\tpccisapicom.dll,4,All
                        .asax,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
                        .ascx,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
                        .ashx,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,1,GET,HEAD,POST,DEBUG

        .asmx,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,1,GET,HEAD,POST,DEBUG
                        .aspx,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,1,GET,HEAD,POST,DEBUG
                        .axd,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,1,GET,HEAD,POST,DEBUG

        .vsdisco,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,1,GET,HEAD,POST,DEBUG
                        .rem,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,1,GET,HEAD,POST,DEBUG

        .soap,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,1,GET,HEAD,POST,DEBUG

        .config,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
                        .cs,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

        .csproj,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
                        .vb,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

        .vbproj,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

        .webinfo,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
                        .licx,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
                        .resx,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

        .resources,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

        .master,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
                        .skin,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

        .compiled,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

        .browser,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
                        .mdb,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
                        .jsl,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

        .vjsproj,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

        .sitemap,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

        .msgx,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,1,GET,HEAD,POST,DEBUG
```

```
                        .ad,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
                        .dd,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
                        .ldd,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
                        .sd,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
                        .cd,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

        .adprototype,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

        .lddprototype,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
                        .sdm,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

        .sdmDocument,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
                        .ldb,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
                        .mdf,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
                        .ldf,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG
                        .java,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

        .exclude,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG

        .refresh,C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll,5,GET,HEAD,POST,DEBUG"
        >
</IIsWebVirtualDir>
<IIsApplicationPools        Location ="/LM/W3SVC/AppPools"

        AdminACL="49634462f0000000a40000040000000205de133b7ca0ce57dfb46e08b0ddb808738d81614268a7132809b862
0ea23f276ba880eea2981453129199c23169b8620ae004989f6a3ec8176b3ad4d17fdce48690b2d61882b883b974b1e53622ab798727d
945d1140316f0475339787940de1d6ac70051b314bc03c6a3c5f896eed391b2ebe434d4505350c78b2c1e8192f70f42d2544110651934
20bc1e90e3363d36f8da2fae52e7663379b7da17e1014a5bb71100000000000b83fa859309f1cab46d2a3e0787004f475f75d1b8878bf2
7a9d14dd6d972c16b86bf64956bc3e321f004b9de75f326641f3afc08a6523b8bddad482933946b"
                AppPoolIdentityType="2"
                AppPoolQueueLength="1000"
                CPULimit="0"
                CPUResetInterval="5"
                DisallowOverlappingRotation="FALSE"
                DisallowRotationOnConfigChange="FALSE"
                Enable32BitAppOnWin64="TRUE"
                IdleTimeout="20"
                LoadBalancerCapabilities="2"
                LogEventOnRecycle="AppPoolRecycleTime | AppPoolRecycleMemory | AppPoolRecyclePrivateMemory"
                MaxProcesses="1"
                OrphanWorkerProcess="FALSE"
                PeriodicRestartMemory="0"
                PeriodicRestartPrivateMemory="0"
                PeriodicRestartRequests="0"
                PeriodicRestartTime="1740"
                PingInterval="30"
                PingResponseTime="90"
                PingingEnabled="TRUE"
                RapidFailProtection="TRUE"
                RapidFailProtectionInterval="5"
                RapidFailProtectionMaxCrashes="5"
                SMPAffinitized="FALSE"
                SMPProcessorAffinityMask="4294967295"
                ShutdownTimeLimit="90"
                StartupTimeLimit="90"
        >
</IIsApplicationPools>
<IIsApplicationPool Location ="/LM/W3SVC/AppPools/DefaultAppPool"
        >
</IIsApplicationPool>
<IIsFilters         Location ="/LM/W3SVC/Filters"

        AdminACL="49634462f0000000a40000040000000205de133b7ca0ce57dfb46e08b0ddb808738d81614268a7132809b862
0ea23f276ba880ee12981453129199c23169b8620ae004989f6a3ec8a76b3ad4d17fdce48690b2d61882b883b974b1e58622ab798727d
945d1140316f0475339787940de1d6ac70051b314bc03c6a3c5f896eed391b2ebe434d4505350c78b2c1e8192f70f42d2544110651934
20bc1e90e3363d36f8da2fae52e7663379b7da17e1014a5bb711000000000e8fc550ebc8727b243ab4cb24a539ea3a3df2cca372eab2c7
28165478b1c51d3c813aa43f17139a9c4d8144875e1c39ac1585d41f2f453cee1b93dde795b49cb"
```

```
                FilterLoadOrder="ASP.NET_2.0.50727.0"
        >
</IIsFilters>
<IIsFilter Location ="/LM/W3SVC/Filters/ASP.NET_2.0.50727.0"
                FilterDescription="ASP.NET Cookieless Session Filter"
                FilterEnableCache="TRUE"
                FilterFlags="NotifyPreProcHeaders | NotifyUrlMap | NotifyOrderLow"
                FilterPath="C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_filter.dll"
                FilterState="1"
        >
</IIsFilter>
<IIsFilter Location ="/LM/W3SVC/Filters/Compression"
        >
</IIsFilter>
<IIsCompressionScheme   Location ="/LM/W3SVC/Filters/Compression/deflate"
                HcCompressionDll="%windir%\system32\inetsrv\gzip.dll"
                HcCreateFlags="0"
                HcDoDynamicCompression="TRUE"
                HcDoOnDemandCompression="TRUE"
                HcDoStaticCompression="FALSE"
                HcDynamicCompressionLevel="0"
                HcFileExtensions="htm
                        html
                        txt"
                HcOnDemandCompLevel="10"
                HcPriority="1"
                HcScriptFileExtensions="asp
                        dll
                        exe"
        >
</IIsCompressionScheme>
<IIsCompressionScheme   Location ="/LM/W3SVC/Filters/Compression/gzip"
                HcCompressionDll="%windir%\system32\inetsrv\gzip.dll"
                HcCreateFlags="1"
                HcDoDynamicCompression="TRUE"
                HcDoOnDemandCompression="TRUE"
                HcDoStaticCompression="TRUE"
                HcDynamicCompressionLevel="0"
                HcFileExtensions="htm
                        html
                        txt"
                HcOnDemandCompLevel="10"
                HcPriority="1"
                HcScriptFileExtensions="asp
                        dll
                        exe"
        >
</IIsCompressionScheme>
<IIsCompressionSchemes  Location ="/LM/W3SVC/Filters/Compression/Parameters"
                HcCacheControlHeader="max-age=86400"
                HcCompressionBufferSize="8192"
                HcCompressionDirectory="%windir%\IIS Temporary Compressed Files"
                HcDoDiskSpaceLimiting="FALSE"
                HcDoDynamicCompression="FALSE"
                HcDoOnDemandCompression="TRUE"
                HcDoStaticCompression="FALSE"
                HcExpiresHeader="Wed, 01 Jan 1997 12:00:00 GMT"
                HcFilesDeletedPerDiskFree="256"
                HcIoBufferSize="8192"
                HcMaxDiskSpaceUsage="100000000"
                HcMaxQueueLength="1000"
                HcMinFileSizeForComp="1"
                HcNoCompressionForHttp10="TRUE"
                HcNoCompressionForProxies="TRUE"
                HcNoCompressionForRange="FALSE"
                HcSendCacheHeaders="FALSE"
        >
```

```
</IIsCompressionSchemes>
<IIsWebInfo         Location ="/LM/W3SVC/Info"

            AdminACL="49634462a0000000580000004000000205de133b7ca0ce5c9fb46e05f0ddb808738d81614268a713280d78625
ea23f276ba840ec1298545312a199c23169b8613ae0049a9f4b7ec0074b3ad4c16fdcb68690b28558a2b883a956f1ed3622ab2b9777d9
47d13403419a79bbbca93e3788fa694f43e9f81459064fc0d06742c04c29f7040fc9c382994a9a0220e94d0bb8c137123c8fa94baf83082
092694d18ce55fcc5e0b003b75"
                    CustomErrorDescriptions="400,0,Bad Request,,0
                                401,1,Unauthorized,Logon failed,1
                                401,2,Unauthorized,Logon failed due to server configuration,1
                                401,3,Unauthorized,Unauthorized due to ACL on resource,1
                                401,4,Unauthorized,Authorization failed by filter,1
                                401,5,Unauthorized,Authorization failed by ISAPI/CGI app,1
                                401,7,Unauthorized,Denied due to URL Authorization policy,0
                                403,1,Forbidden,Execute access denied,0
                                403,2,Forbidden,Read access denied,0
                                403,3,Forbidden,Write access denied,0
                                403,4,Forbidden,SSL required,0
                                403,5,Forbidden,SSL128 required,0
                                403,6,Forbidden,IP address rejected,0
                                403,7,Forbidden,Client certificate required,0
                                403,8,Forbidden,Site access denied,0
                                403,9,Forbidden,Too many users,0
                                403,10,Forbidden,Invalid Configuration,0
                                403,11,Forbidden,Password Change,0
                                403,12,Forbidden,Mapper Denied Access,0
                                403,13,Forbidden,Client certificate revoked,0
                                403,14,Forbidden,Directory Listing Denied,0
                                403,15,Forbidden,Client Access Licenses Exceeded,0
                                403,16,Forbidden,Client certificate untrusted or ill-formed,0
                                403,17,Forbidden,Client certificate has expired or is not yet valid,0
                                403,18,Forbidden,Cannot execute request from this application pool,0
                                403,19,Forbidden,CGI Access denied,0
                                403,20,Forbidden,Passport Login failed,0
                                404,0,Not Found,,0
                                404,2,Not Found,Denied due to Lockdown Policy,0
                                404,3,Not Found,Denied due to MIMEMAP Policy,0
                                405,0,Method Not Allowed,,0
                                406,0,Not Acceptable,,0
                                407,0,Proxy Authentication Required,,1
                                412,0,Precondition Failed,,0
                                414,0,Request-URI Too Long,,0
                                415,0,Unsupported Media Type,,0
                                500,0,Internal Server Error,,0
                                500,12,Internal Server Error,Application restarting,0
                                500,13,Internal Server Error,Server too busy,0
                                500,15,Internal Server Error,Direct requests for GLOBAL.ASA forbidden,0
                                500,16,Internal Server Error,UNC Access Error,0
                                500,17,Internal Server Error,URL Authorization store not found,0
                                500,18,Internal Server Error,URL Authorization store cannot be opened,0
                                500,19,Internal Server Error,Bad file metadata,0
                                500,100,Internal Server Error,ASP error,0
                                501,0,Not Implemented,,0
                                502,0,Bad Gateway,,1"
                    LogModuleList="NCSA Common Log File Format,Microsoft IIS Log File Format,W3C Extended Log File
Format,ODBC Logging"
                    MD_SERVER_CAPABILITIES="129983"
                    MD_SERVER_PLATFORM="1"
                    MajorIIsVersionNumber="6"
                    MinorIIsVersionNumber="0"
          >
</IIsWebInfo>
<IIsConfigObject    Location ="/LM/W3SVC/Info/Templates"
          >
</IIsConfigObject>
<IIsWebServer       Location ="/LM/W3SVC/Info/Templates/Public Web Site"
                    ServerComment="Allows all users to browse static and dynamic content."
```

```
            >
</IIsWebServer>
<IIsWebVirtualDir   Location ="/LM/W3SVC/Info/Templates/Public Web Site/Root"
                    AccessFlags="AccessRead | AccessScript"
                    AuthFlags="AuthAnonymous"
                    IPSecurity=""
            >
</IIsWebVirtualDir>
<IIsWebServer       Location ="/LM/W3SVC/Info/Templates/Secure Web Site"
                    ServerComment="Allows users with a Windows account to view static and dynamic content."
            >
</IIsWebServer>
<IIsWebVirtualDir   Location ="/LM/W3SVC/Info/Templates/Secure Web Site/Root"
                    AccessFlags="AccessRead | AccessScript"
                    AuthFlags="AuthBasic | AuthNTLM | AuthMD5"
                    IPSecurity=""
            >
</IIsWebVirtualDir>
</MBProperty>
</configuration>
""
""
""
"Disks:"

Microsoft DiskPart version 5.2.3790.3959
Copyright (C) 1999-2001 Microsoft Corporation.
On computer: NEWTON

  Disk ###  Status      Size     Free     Dyn  Gpt
  --------  ----------  -------  -------  ---  ---
  Disk 0    Online        33 GB  8001 KB
  Disk 1    Online        68 GB      0 B
  Disk 2    Online       681 GB      0 B
  Disk 3    Online      1016 GB      0 B

  Volume ###  Ltr  Label        Fs     Type        Size     Status     Info
  ----------  ---  -----------  -----  ----------  -------  ---------  --------
  Volume 0     M   New Volume   NTFS   Partition    681 GB  Healthy
  Volume 1     J   New Volume   NTFS   Partition   1016 GB  Healthy
  Volume 2     C   OS           NTFS   Partition     12 GB  Healthy    System
  Volume 3     F   DATAPART1    NTFS   Partition     21 GB  Healthy
  Volume 4     E                       DVD-ROM         0 B  Healthy
  Volume 5     D   New Volume   NTFS   Partition     68 GB  Healthy
List of Controllers in the system

Controllers
ID                              : 0
Status                          : Ok
Name                            : PERC 5/i Integrated
Slot ID                         : Embedded
State                           : Ready
Firmware Version                : 5.2.1-0067
Minimum Required Firmware Version : Not Applicable
Driver Version                  : 2.14.00.64
Minimum Required Driver Version   : Not Applicable
Number of Connectors            : 2
Rebuild Rate                    : 30%
BGI Rate                        : 30%
Check Consistency Rate          : 0%
Reconstruct Rate                : 30%
Alarm State                     : Not Applicable
Cluster Mode                    : Not Applicable
SCSI Initiator ID               : Not Applicable
Cache Memory Size               : 256 MB
Patrol Read Mode                : Disabled
Patrol Read State               : Stopped
```

Patrol Read Rate : 30%
Patrol Read Iterations : 1

ID : 1
Status : Ok
Name : PERC 6/E Adapter
Slot ID : PCI Slot 2
State : Ready
Firmware Version : 6.0.2-0002
Minimum Required Firmware Version : Not Applicable
Driver Version : 2.14.00.64
Minimum Required Driver Version : Not Applicable
Number of Connectors : 2
Rebuild Rate : 0%
BGI Rate : 0%
Check Consistency Rate : 0%
Reconstruct Rate : 0%
Alarm State : Not Applicable
Cluster Mode : Not Applicable
SCSI Initiator ID : Not Applicable
Cache Memory Size : 512 MB
Patrol Read Mode : Disabled
Patrol Read State : Stopped
Patrol Read Rate : 0%
Patrol Read Iterations : 0
List of Physical Disks on Controller PERC 5/i Integrated (Embedded)

Controller PERC 5/i Integrated (Embedded)
ID : 0:0:0
Status : Ok
Name : Physical Disk 0:0:0
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 33.38 GB (35836133376 bytes)
Used RAID Disk Space : 33.38 GB (35836133376 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : MBC2036RC
Revision : D505
Serial No. : BXA3P83008KE
Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 02
Manufacture Week : 10
Manufacture Year : 2008
SAS Address : 500000E01A54FB22

ID : 0:0:1
Status : Ok
Name : Physical Disk 0:0:1
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 33.38 GB (35836133376 bytes)
Used RAID Disk Space : 33.38 GB (35836133376 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : MBC2036RC
Revision : D505
Serial No. : BXA3P83008KD
Negotiated Speed : Not Available
Capable Speed : Not Available

Manufacture Day : 02
Manufacture Week : 10
Manufacture Year : 2008
SAS Address : 500000E01A54CC12

ID : 0:0:2
Status : Ok
Name : Physical Disk 0:0:2
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 67.75 GB (72746008576 bytes)
Used RAID Disk Space : 67.75 GB (72746008576 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST973451SS
Revision : SM04
Serial No. : 3PD1AD83
Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 08
Manufacture Week : 05
Manufacture Year : 2005
SAS Address : 5000C50007C37DC9

ID : 0:0:3
Status : Ok
Name : Physical Disk 0:0:3
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 136.13 GB (146163105792 bytes)
Used RAID Disk Space : 136.13 GB (146163105792 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST9146802SS
Revision : S229
Serial No. : 3NM5QQZA
Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 08
Manufacture Week : 18
Manufacture Year : 2005
SAS Address : 5000C5000B37DEF5

ID : 1:0:4
Status : Ok
Name : Physical Disk 1:0:4
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 136.13 GB (146163105792 bytes)
Used RAID Disk Space : 136.13 GB (146163105792 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST9146802SS
Revision : S229
Serial No. : 3NM5N9A6
Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 08

Manufacture Week : 18
Manufacture Year : 2005
SAS Address : 5000C5000B37C8F5

ID : 1:0:5
Status : Ok
Name : Physical Disk 1:0:5
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 136.13 GB (146163105792 bytes)
Used RAID Disk Space : 136.13 GB (146163105792 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST9146802SS
Revision : S229
Serial No. : 3NM5NW6V
Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 08
Manufacture Week : 18
Manufacture Year : 2005
SAS Address : 5000C5000B378B2D

ID : 1:0:6
Status : Ok
Name : Physical Disk 1:0:6
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 136.13 GB (146163105792 bytes)
Used RAID Disk Space : 136.13 GB (146163105792 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST9146802SS
Revision : S229
Serial No. : 3NM5NAQ7
Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 08
Manufacture Week : 18
Manufacture Year : 2005
SAS Address : 5000C5000B379D09

ID : 1:0:7
Status : Ok
Name : Physical Disk 1:0:7
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 136.13 GB (146163105792 bytes)
Used RAID Disk Space : 136.13 GB (146163105792 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST9146802SS
Revision : S229
Serial No. : 3NM5PKX2
Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 08
Manufacture Week : 18

Manufacture Year : 2005
SAS Address : 5000C5000B382C99
List of Virtual Disks on Controller PERC 5/i Integrated (Embedded)

Controller PERC 5/i Integrated (Embedded)
ID : 0
Status : Ok
Name : Virtual Disk 0
State : Ready
Progress : Not Applicable
Layout : RAID-1
Size : 33.38 GB (35836133376 bytes)
Device Name : Windows Disk 0
Type : SAS
Read Policy : No Read Ahead
Write Policy : Write Through
Cache Policy : Not Applicable
Stripe Element Size : 64 KB
Disk Cache Policy : Disabled

ID : 1
Status : Ok
Name : DDRIVE
State : Ready
Progress : Not Applicable
Layout : RAID-0
Size : 67.75 GB (72746008576 bytes)
Device Name : Windows Disk 1
Type : SAS
Read Policy : No Read Ahead
Write Policy : Write Through
Cache Policy : Not Applicable
Stripe Element Size : 64 KB
Disk Cache Policy : Disabled

ID : 2
Status : Ok
Name : MISC
State : Ready
Progress : Not Applicable
Layout : RAID-0
Size : 680.63 GB (730815528960 bytes)
Device Name : Windows Disk 2
Type : SAS
Read Policy : No Read Ahead
Write Policy : Write Through
Cache Policy : Not Applicable
Stripe Element Size : 64 KB
Disk Cache Policy : Disabled
List of Physical Disks on Controller PERC 6/E Adapter (Slot 2)

Controller PERC 6/E Adapter (Slot 2)
ID : 1:0:0
Status : Ok
Name : Physical Disk 1:0:0
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 67.75 GB (72746008576 bytes)
Used RAID Disk Space : 67.75 GB (72746008576 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST373455SS
Revision : S527
Serial No. : 3LQ34CCY

Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 08
Manufacture Week : 16
Manufacture Year : 2005
SAS Address : 5000C500090B3B06

ID : 1:0:1
Status : Ok
Name : Physical Disk 1:0:1
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 67.75 GB (72746008576 bytes)
Used RAID Disk Space : 67.75 GB (72746008576 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST373455SS
Revision : S527
Serial No. : 3LQ35H87
Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 08
Manufacture Week : 16
Manufacture Year : 2005
SAS Address : 5000C500090B2ADE

ID : 1:0:2
Status : Ok
Name : Physical Disk 1:0:2
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 67.75 GB (72746008576 bytes)
Used RAID Disk Space : 67.75 GB (72746008576 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST373455SS
Revision : S527
Serial No. : 3LQ35RNR
Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 08
Manufacture Week : 16
Manufacture Year : 2005
SAS Address : 5000C500090A9A42

ID : 1:0:3
Status : Ok
Name : Physical Disk 1:0:3
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 67.75 GB (72746008576 bytes)
Used RAID Disk Space : 67.75 GB (72746008576 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST373455SS
Revision : S527
Serial No. : 3LQ360MP
Negotiated Speed : Not Available

Capable Speed : Not Available
Manufacture Day : 08
Manufacture Week : 16
Manufacture Year : 2005
SAS Address : 5000C500090B7EBE

ID : 1:0:4
Status : Ok
Name : Physical Disk 1:0:4
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 67.75 GB (72746008576 bytes)
Used RAID Disk Space : 67.75 GB (72746008576 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST373455SS
Revision : S527
Serial No. : 3LQ34VJF
Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 08
Manufacture Week : 16
Manufacture Year : 2005
SAS Address : 5000C500090AA2F2

ID : 1:0:5
Status : Ok
Name : Physical Disk 1:0:5
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 67.75 GB (72746008576 bytes)
Used RAID Disk Space : 67.75 GB (72746008576 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST373455SS
Revision : S527
Serial No. : 3LQ35NWT
Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 08
Manufacture Week : 16
Manufacture Year : 2005
SAS Address : 5000C500090AC8D2

ID : 1:0:6
Status : Ok
Name : Physical Disk 1:0:6
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 67.75 GB (72746008576 bytes)
Used RAID Disk Space : 67.75 GB (72746008576 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST373455SS
Revision : S527
Serial No. : 3LQ35NNF
Negotiated Speed : Not Available
Capable Speed : Not Available

Manufacture Day : 08
Manufacture Week : 16
Manufacture Year : 2005
SAS Address : 5000C500090AA7CA

ID : 1:0:7
Status : Ok
Name : Physical Disk 1:0:7
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 67.75 GB (72746008576 bytes)
Used RAID Disk Space : 67.75 GB (72746008576 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST373455SS
Revision : S527
Serial No. : 3LQ34NMX
Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 08
Manufacture Week : 16
Manufacture Year : 2005
SAS Address : 5000C500090BA4D6

ID : 1:0:8
Status : Ok
Name : Physical Disk 1:0:8
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 67.75 GB (72746008576 bytes)
Used RAID Disk Space : 67.75 GB (72746008576 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST373455SS
Revision : S527
Serial No. : 3LQ35MHR
Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 08
Manufacture Week : 16
Manufacture Year : 2005
SAS Address : 5000C500090AB1EA

ID : 1:0:9
Status : Ok
Name : Physical Disk 1:0:9
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 67.75 GB (72746008576 bytes)
Used RAID Disk Space : 67.75 GB (72746008576 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST373455SS
Revision : S527
Serial No. : 3LQ35JCJ
Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 08

Manufacture Week : 16
Manufacture Year : 2005
SAS Address : 5000C500090B1A6A

ID : 1:0:10
Status : Ok
Name : Physical Disk 1:0:10
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 67.75 GB (72746008576 bytes)
Used RAID Disk Space : 67.75 GB (72746008576 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST373455SS
Revision : S527
Serial No. : 3LQ3609Z
Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 08
Manufacture Week : 16
Manufacture Year : 2005
SAS Address : 5000C500090BA4E6

ID : 1:0:11
Status : Ok
Name : Physical Disk 1:0:11
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 67.75 GB (72746008576 bytes)
Used RAID Disk Space : 67.75 GB (72746008576 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST373455SS
Revision : S527
Serial No. : 3LQ34B2B
Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 08
Manufacture Week : 16
Manufacture Year : 2005
SAS Address : 5000C500090B4EDA

ID : 1:0:12
Status : Ok
Name : Physical Disk 1:0:12
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 67.75 GB (72746008576 bytes)
Used RAID Disk Space : 67.75 GB (72746008576 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST373455SS
Revision : S527
Serial No. : 3LQ3610Q
Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 08
Manufacture Week : 16

Manufacture Year : 2005
SAS Address : 5000C500090B84DE

ID : 1:0:13
Status : Ok
Name : Physical Disk 1:0:13
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 67.75 GB (72746008576 bytes)
Used RAID Disk Space : 67.75 GB (72746008576 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST373455SS
Revision : S527
Serial No. : 3LQ35YV9
Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 08
Manufacture Week : 16
Manufacture Year : 2005
SAS Address : 5000C500090BA4FA

ID : 1:0:14
Status : Ok
Name : Physical Disk 1:0:14
State : Online
Failure Predicted : No
Progress : Not Applicable
Type : SAS
Capacity : 67.75 GB (72746008576 bytes)
Used RAID Disk Space : 67.75 GB (72746008576 bytes)
Available RAID Disk Space : 0.00 GB (0 bytes)
Hot Spare : No
Vendor ID : DELL
Product ID : ST373455SS
Revision : S527
Serial No. : 3LQ3DAV1
Negotiated Speed : Not Available
Capable Speed : Not Available
Manufacture Day : 08
Manufacture Week : 23
Manufacture Year : 2005
SAS Address : 5000C5000A87F0C6
Virtual Disk 0 on Controller PERC 6/E Adapter (Slot 2)

Controller PERC 6/E Adapter (Slot 2)
ID : 0
Status : Ok
Name : JDRIVE
State : Ready
Progress : Not Applicable
Layout : RAID-0
Size : 1,016.25 GB (1091190128640 bytes)
Device Name : Windows Disk 3
Type : SAS
Read Policy : No Read Ahead
Write Policy : Write Through
Cache Policy : Not Applicable
Stripe Element Size : 64 KB
Disk Cache Policy : Enabled

Adapter #0

==========

```
                Versions
               ================
Product Name    : PERC 5/i Integrated
Serial No       : 12345
FW Package Build: 5.2.1-0067

                Mfg. Data
               ================
Mfg. Date       : 00/00/00
Rework Date     : 00/00/00
Revision No     : @À¬ ôA ¬
Battery FRU     : N/A

               Image Versions In Flash:
               ================
Boot Block Version : R.2.3.12
BIOS Version       : MT28-8
MPT Version        : MPTFW-00.10.61.00-IT
FW Version         : 1.03.40-0316
WebBIOS Version    : 1.03-04
Ctrl-R Version     : 1.04-019A

               Pending Images In Flash
               ================
None

               PCI Info
               ================
Vendor Id       : 1028
Device Id       : 0015
SubVendorId     : 1028
SubDeviceId     : 1f03

Host Interface  : PCIE

Number of Frontend Port: 0
Device Interface  : PCIE

Number of Backend Port: 8
Port  :  Address
0       500000e01a54fb22
1       500000e01a54cc12
2       5000c50007c37dc9
3       5000c5000b37def5
4       5000c5000b37c8f5
5       5000c5000b378b2d
6       5000c5000b379d09
7       5000c5000b382c99

               HW Configuration
               ================
SAS Address     : 5001e4f023719000
BBU             : Present
Alarm           : Absent
NVRAM           : Present
Serial Debugger : Present
Memory          : Present
Flash           : Present
Memory Size     : 256MB

               Settings
               ================
Current Time                     : 19:44:0 7/15, 2008
Predictive Fail Poll Interval    : 10000sec
Interrupt Throttle Active Count  : 16
Interrupt Throttle Completion    : 50us
Rebuild Rate                     : 30%
```

```
PR Rate                         : 30%
Resynch Rate                    : 30%
Check Consistency Rate          : 0%
Reconstruction Rate             : 30%
Cache Flush Interval            : 60s
Max Drives to Spinup at One Time : 0
Delay Among Spinup Groups       : 120s
Physical Drive Coercion Mode    : 128MB
Cluster Mode                    : Disabled
Alarm                           : Disabled
Auto Rebuild                    : Disabled
Battery Warning                 : Enabled
Ecc Bucket Size                 : 15
Ecc Bucket Leak Rate            : 1440 Minutes
Restore HotSpare on Insertion   : Disabled
Expose Enclosure Devices        : Disabled
Maintain PD Fail History        : Disabled
Host Request Reordering         : Enabled
Auto Detect BackPlane Enabled   : SGPIO/i2c SEP
Load Balance Mode               : Auto

            Capabilities
            ================
RAID Level Supported            : RAID0, RAID1, RAID5, RAID10, RAID50
Supported Drives                : SAS, SATA

Allowed Mixing:
Mix In Enclosure Allowed

            Status
            ================
ECC Bucket Count                : 0

            Limitations
            ================
Max Arms Per VD         : 32
Max Spans Per VD        : 8
Max Arrays              : 128
Max Number of VDs       : 64
Max Parallel Commands   : 1008
Max SGE Count           : 80
Max Data Transfer Size  : 8192 sectors
Max Strips PerIO        : 84
Min Stripe Size         : 8kB
Max Stripe Size         : 128kB

            Device Present
            ================
Virtual Drives    : 3
  Degraded        : 0
  Offline         : 0
Physical Devices  : 9
  Disks           : 8
  Critical Disks  : 0
  Failed Disks    : 0

            Supported Adapter Operations
            ================
Rebuild Rate                    : Yes
CC Rate                         : Yes
BGI Rate                        : Yes
Reconstruct Rate                : Yes
Patrol Read Rate                : Yes
Alarm Control                   : Yes
Cluster Support                 : No
BBU                             : Yes
Spanning                        : Yes
```

```
Dedicated Hot Spare             : Yes
Revertible Hot Spares           : No
Foreign Config Import           : Yes
Self Diagnostic                 : Yes
Allow Mixed Redundancy on Array : No
Global Hot Spares               : Yes
Deny SCSI Passthrough           : No
Deny SMP Passthrough            : No
Deny STP Passthrough            : No

            Supported VD Operations
            ================
Read Policy          : Yes
Write Policy         : Yes
IO Policy            : Yes
Access Policy        : Yes
Disk Cache Policy    : Yes
Reconstruction       : Yes
Deny Locate          : No
Deny CC              : No

            Supported PD Operations
            ================
Force Online             : Yes
Force Offline            : Yes
Force Rebuild            : Yes
Deny Force Failed        : No
Deny Force Good/Bad      : No
Deny Missing Replace     : No
Deny Clear               : No
Deny Locate              : No
Disable Copyback         : No
Enable Copyback on SMART : No

            Error Counters
            ================
Memory Correctable Errors   : 0
Memory Uncorrectable Errors : 0

            Cluster Information
            ================
Cluster Permitted    : No
Cluster Active       : No

            Default Settings
            ================
Phy Polarity                : 0
Phy PolaritySplit           : 0
Background Rate             : 30
Stripe Size                 : 64kB
Flush Time                  : 4 seconds
Write Policy                : WB
Read Policy                 : None
Cache When BBU Bad          : Disabled
Cached IO                   : No
SMART Mode                  : Mode 6
Alarm Disable               : No
Coercion Mode               : 128MB
ZCR Config                  : IDSEL
Dirty LED Shows Drive Activity   : No
BIOS Continue on Error      : No
Spin Down Mode              : None
Allowed Device Type         : SAS/SATA Mix
Allow Mix In Enclosure      : Yes
Allow Mix In VD             : No
Allow SATA In Cluster       : No
Max Chained Enclosures      : 1
```

```
Disable Ctrl-R                  : No
Enable Web BIOS                 : No
Direct PD Mapping               : No
BIOS Enumerate VDs              : No
Restore Hot Spare on Insertion  : No
Expose Enclosure Devices        : No
Maintain PD Fail History        : No
Disable Puncturing              : No
Zero Based Enclosure Enumeration : No
PreBoot CLI Enabled             : Yes
LED Show Drive Activity         : No
Cluster Disable                 : Yes
SAS Disable                     : No
Auto Detect BackPlane Enable    : SGPIO/i2c SEP
Adapter #1

==============================================================================
                Versions
               ================
Product Name    : PERC 6/E Adapter
Serial No       : 1122334455667788
FW Package Build: 6.0.2-0002

                Mfg. Data
               ================
Mfg. Date       : 06/08/07
Rework Date     : 06/08/07
Revision No     :
Battery FRU     : N/A

               Image Versions In Flash:
               ================
FW Version         : 1.11.52-0396
BIOS Version       : NT13-2
WebBIOS Version    : 1.1-32-e_11-Rel
Ctrl-R Version     : 1.01-010B
Boot Block Version : 1.00.00.01-0008

               Pending Images In Flash
               ================
None

               PCI Info
               ================
Vendor Id       : 1000
Device Id       : 0060
SubVendorId     : 1028
SubDeviceId     : 1f0a

Host Interface  : PCIE

Number of Frontend Port: 0
Device Interface  : PCIE

Number of Backend Port: 8
Port  :  Address
0        5001e4f241fb7200
1        0000000000000000
2        0000000000000000
3        0000000000000000
4        5001e4f241f7b800
5        0000000000000000
6        0000000000000000
7        0000000000000000

               HW Configuration
               ================
```

```
SAS Address     : 5001ec90d4ee7100
BBU             : Present
Alarm           : Absent
NVRAM           : Present
Serial Debugger : Present
Memory          : Present
Flash           : Present
Memory Size     : 512MB

                Settings
                ================
Current Time                     : 19:44:7 7/15, 2008
Predictive Fail Poll Interval    : 300sec
Interrupt Throttle Active Count  : 16
Interrupt Throttle Completion    : 50us
Rebuild Rate                     : 0%
PR Rate                          : 0%
Resynch Rate                     : 0%
Check Consistency Rate           : 0%
Reconstruction Rate              : 0%
Cache Flush Interval             : 60s
Max Drives to Spinup at One Time : 2
Delay Among Spinup Groups        : 12s
Physical Drive Coercion Mode     : 128MB
Cluster Mode                     : Disabled
Alarm                            : Disabled
Auto Rebuild                     : Enabled
Battery Warning                  : Enabled
Ecc Bucket Size                  : 15
Ecc Bucket Leak Rate             : 1440 Minutes
Restore HotSpare on Insertion    : Disabled
Expose Enclosure Devices         : Disabled
Maintain PD Fail History         : Disabled
Host Request Reordering          : Enabled
Auto Detect BackPlane Enabled    : SGPIO/i2c SEP
Load Balance Mode                : Auto

                Capabilities
                ================
RAID Level Supported             : RAID0, RAID1, RAID5, RAID6, RAID10, RAID50, RAID60
Supported Drives                 : SAS, SATA

Allowed Mixing:
Mix In Enclosure Allowed

                Status
                ================
ECC Bucket Count                 : 0

                Limitations
                ================
Max Arms Per VD          : 32
Max Spans Per VD         : 8
Max Arrays               : 128
Max Number of VDs        : 64
Max Parallel Commands    : 1008
Max SGE Count            : 80
Max Data Transfer Size   : 8192 sectors
Max Strips PerIO         : 42
Min Stripe Size          : 8kB
Max Stripe Size          : 1024kB

                Device Present
                ================
Virtual Drives    : 1
  Degraded        : 0
  Offline         : 0
```

```
Physical Devices  : 16
  Disks           : 15
  Critical Disks  : 0
  Failed Disks    : 0

                Supported Adapter Operations
                ================
Rebuild Rate                  : Yes
CC Rate                       : Yes
BGI Rate                      : Yes
Reconstruct Rate              : Yes
Patrol Read Rate              : Yes
Alarm Control                 : Yes
Cluster Support               : No
BBU                           : Yes
Spanning                      : Yes
Dedicated Hot Spare           : Yes
Revertible Hot Spares         : No
Foreign Config Import         : Yes
Self Diagnostic               : Yes
Allow Mixed Redundancy on Array : No
Global Hot Spares             : Yes
Deny SCSI Passthrough         : No
Deny SMP Passthrough          : No
Deny STP Passthrough          : No

                Supported VD Operations
                ================
Read Policy          : Yes
Write Policy         : Yes
IO Policy            : Yes
Access Policy        : Yes
Disk Cache Policy    : Yes
Reconstruction       : Yes
Deny Locate          : No
Deny CC              : No

                Supported PD Operations
                ================
Force Online             : Yes
Force Offline            : Yes
Force Rebuild            : Yes
Deny Force Failed        : No
Deny Force Good/Bad      : No
Deny Missing Replace     : No
Deny Clear               : No
Deny Locate              : No
Disable Copyback         : No
Enable Copyback on SMART  : No

                Error Counters
                ================
Memory Correctable Errors   : 0
Memory Uncorrectable Errors : 0

                Cluster Information
                ================
Cluster Permitted     : No
Cluster Active        : No

                Default Settings
                ================
Phy Polarity                  : 0
Phy PolaritySplit             : 0
Background Rate               : 30
Stripe Size                   : 64kB
Flush Time                    : 4 seconds
```

```
Write Policy                    : WB
Read Policy                     : None
Cache When BBU Bad              : Disabled
Cached IO                       : No
SMART Mode                      : Mode 6
Alarm Disable                   : No
Coercion Mode                   : 128MB
ZCR Config                      : Unknown
Dirty LED Shows Drive Activity  : No
BIOS Continue on Error          : No
Spin Down Mode                  : None
Allowed Device Type             : SAS/SATA Mix
Allow Mix In Enclosure          : Yes
Allow Mix In VD                 : No
Allow SATA In Cluster           : No
Max Chained Enclosures          : 3
Disable Ctrl-R                  : No
Enable Web BIOS                 : No
Direct PD Mapping               : No
BIOS Enumerate VDs              : Yes
Restore Hot Spare on Insertion  : No
Expose Enclosure Devices        : No
Maintain PD Fail History        : No
Disable Puncturing              : No
Zero Based Enclosure Enumeration : Yes
PreBoot CLI Enabled             : No
LED Show Drive Activity         : No
Cluster Disable                 : Yes
SAS Disable                     : No
Auto Detect BackPlane Enable    : SGPIO/i2c SEP

Exit Code: 0x00
""
""
""
""
"System summary:"
System Summary

------------------
Software Profile
------------------
Systems Management
Name                    : Dell OpenManage Server Administrator
Version                 : 5.4.0
Description             : Systems Management Software
Contains                : Common Storage Module 2.4.0
                        : Data Engine 5.8.0
                        : Hardware Application Programming Interface 5.8.0
                        : Instrumentation Service 5.8.0
                        : Instrumentation Service Integration Layer 3.4.0
                        : OpenManage Inventory Collector 2.8.0
                        : OpenManage Tools 3.4.0
                        : Remote Access Controller 5 Data Populator 5.8.0
                        : Remote Access Controller 5 Managed Node 5.3.3
                        : Secure Port Server 3.4.0
                        : Server Administrator Framework 3.4.0
                        : Storage Management 2.4.0
                        : Sun Java Runtime Environment 1.6.0

Operating System
Name                    : Microsoft Windows Server 2003 R2, Standard x64 Edition
Version                 : Version 5.2 (Build 3790 : Service Pack 2) (x64)
System Time             : Tue Jul 15 19:44:08 2008
System Bootup Time      : Tue Jul 15 12:10:40 2008

--------
```

System
--------
System
Host Name : NEWTON
System Location : Please set the value

---
Main System Chassis
---
Chassis Information
Chassis Model : PowerEdge 2950
Chassis Service Tag : CXM4BG1
Chassis Lock : Present
Chassis Asset Tag :

Processor 1
Processor Brand : Intel(R) Xeon(R) CPU E5420 @ 2.50GHz
Processor Version : Model 23 Stepping 6
Voltage : 1400 mV

Memory
Total Installed Capacity : 16384 MB
Memory Available to the OS : 16384 MB
Total Maximum Capacity : 32768 MB
Memory Array Count : 1

Memory Array 1
Location : System Board or Motherboard
Use : System Memory
Installed Capacity : 16384 MB
Maximum Capacity : 32768 MB
Slots Available : 8
Slots Used : 8
ECC Type : Multibit ECC

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

Slot PCI2
Adapter : PERC 6/E Adapter
Type : PCI E
Data Bus Width : 8x or x8
Speed : [Not Obtained, see card documentation]
Slot Length : Long
Voltage Supply : 3.3 Volts

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

BIOS Information
Manufacturer : Dell Inc.
Version : 2.2.6
Release Date : 02/05/2008

Firmware Information
Name : Baseboard Management Controller
Version : 2.05

```
Firmware Information
Name                    : Remote Access Controller firmware
Version                 : 1.33

----------
Network Data
----------
Network Interface 0
IP Address              : 10.25.107.85
Subnet Mask             : 255.255.254.0
Default Gateway         : 10.25.106.1
MAC Address             : 00-1E-C9-48-1D-36

Network Interface 1
IP Address              : 192.168.5.5
Subnet Mask             : 255.255.0.0
Default Gateway         : 192.168.5.5
MAC Address             : 00-1E-C9-48-1D-38

--------------------
Storage Enclosures
--------------------
Storage Enclosures
Name                    : Backplane
Service Tag             : 838025R

Storage Enclosures
Name                    : Backplane
Service Tag             : 838025R

Storage Enclosures
Name                    : MD1000
Product ID              : MD1000
Service Tag             : FTM0BG1
Asset Tag               :
""
""
""
""
"Environment variables:"
ALLUSERSPROFILE=C:\Documents and Settings\All Users
APPDATA=C:\Documents and Settings\Administrator\Application Data
CLIENTNAME=DBPREC650
ClusterLog=C:\WINDOWS\Cluster\cluster.log
CommonProgramFiles=C:\Program Files\Common Files
CommonProgramFiles(x86)=C:\Program Files (x86)\Common Files
COMPUTERNAME=NEWTON
ComSpec=C:\WINDOWS\system32\cmd.exe
dir_name=j:\tpcc\sa11_4k_1650w
FP_NO_HOST_CHECK=NO
HOMEDRIVE=C:
HOMEPATH=\Documents and Settings\Administrator
INTEL_LICENSE_FILE=C:\Program Files (x86)\Common Files\Intel\Licenses
LOGONSERVER=\\NEWTON
NUMBER_OF_PROCESSORS=4
OS=Windows_NT
Path=C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;F:\sa\sa11\bin64;F:\sa\sa11\bin32;C:\dell\RAC5;C:\d
ell\oma\bin
PATHEXT=.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH
PROCESSOR_ARCHITECTURE=AMD64
PROCESSOR_IDENTIFIER=EM64T Family 6 Model 23 Stepping 6, GenuineIntel
PROCESSOR_LEVEL=6
PROCESSOR_REVISION=1706
ProgramFiles=C:\Program Files
ProgramFiles(x86)=C:\Program Files (x86)
PROMPT=$P$G
```

SATMP=j:\temp
SESSIONNAME=RDP-Tcp#1
SQLANY11=F:\sa\sa11
SQLANYSAMP11=C:\Documents and Settings\All Users\Documents\SQL Anywhere 11\Samples
sqlconnect=uid=dba;pwd=sql;eng=tpcc_newton
SystemDrive=C:
SystemRoot=C:\WINDOWS
TEMP=C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\1
TMP=C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\1
USERDOMAIN=NEWTON
USERNAME=Administrator
USERPROFILE=C:\Documents and Settings\Administrator
VS90COMNTOOLS=D:\vstudio2008\Common7\Tools\
windir=C:\WINDOWS

# APPENDIX F: SOURCE CODE

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

// ********************************************************************
// Copyright 2002 iAnywhere Solutions, Inc.  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\dlldata.c*/
/*********************************************/

/*********************************************************
   DllData file -- generated by MIDL compiler

        DO NOT ALTER THIS FILE

   This file is regenerated by MIDL on every IDL file compile.

   To completely reconstruct this file, delete it and rerun MIDL
   on all the IDL files in this DLL, specifying this file for the
   /dlldata command line option

*********************************************************/


#include <rpcproxy.h>

#ifdef __cplusplus
extern "C"   {
#endif

EXTERN_PROXY_FILE( tpcccom )


PROXYFILE_LIST_START
/* Start of list */
  REFERENCE_PROXY_FILE( tpcccom ),
/* End of list */
PROXYFILE_LIST_END


DLLDATA_ROUTINES( aProxyFileList, GET_DLL_CLSID )

#ifdef __cplusplus
}  /*extern "C" */
#endif

/* end of generated dlldata file */
```

```
/*c:\original\kit\makesource.py*/
/*********************************************/

import sys
import glob

type = sys.argv[1]
if type == 'source':
    dirs = ['c:\\original\\kit', 'c:\\original\\kit\\acid']
    extensions = ['cpp', 'c', 'hpp', 'h', 'py']
```

```
elif type == 'sql':
    dirs = ['c:\\original\\kit\\watcom', 'c:\\original\\kit\\tsql']
    extensions = ['sql']
else:
    print 'Unknown type'
    sys.exit(1)

files = []
for dir in dirs:
    files += glob.glob(dir+'\\*.*')

ret = ''
for fname in files:
    splits = fname.split('.')
    if len(splits) == 2:
            ext = splits[1]
            if ext in extensions:
                f = open(fname, 'rt')
                ret += '\n\n\n\n\n/*' + fname + '*/\n/************************************************/\n\n'
                while 1:
                        s = f.readline()
                        if s == '':
                            break
                        ret += s
                f.close()

print ret




/*c:\original\kit\odbc.cpp*/
/************************************************/

// *********************************************************************
// Copyright 2002-2008 iAnywhere Solutions, Inc.  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

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();
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

    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::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 2002 iAnywhere Solutions, Inc.  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 2002-2008 iAnywhere Solutions, Inc.  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 <windows.h>
#endif
#include <math.h>
#include <process.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 )
#elif defined( WINNT )
    #define _Sleep( x ) Sleep( x )
#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;

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
}

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

    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( 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 ) {
                Transactions->printRunStats( runstatfile, NWarehouses * 10 );
                fclose( runstatfile );
            }
```

```
    } // don't write output logs
    Transactions->printRunStats( stdout, NWarehouses * 10 );
}

#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 ] ) );
   _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] );
   _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( unsigned argc, char *argv[] )
/*********************************/
{
    unsigned nTxn = 1000;
    while( --argc > 0 ) {
            char * arg = *++argv;
            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, "    -d    (debugging)\n" );
            fprintf( stderr, "    -e <trans_name>    (exclude transaction)\n" );
            fprintf( stderr, "        trans_name: [new_order,payment,order_status,delivery,stock_level]\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 <threads>\n" );
            fprintf( stderr, "    -v <delivery threads>\n" );
            fprintf( stderr, "    -w <warehouses>\n" );
            fprintf( stderr, "    -wl <warehouse limit>\n" );
            fprintf( stderr, "    -x <trans_name>    (execute only transaction)\n" );
            fprintf( stderr, "        trans_name: [new_order,payment,order_status,delivery,stock_level]\n" );
            fprintf( stderr, "    -z <display_type>\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 2002-2008 iAnywhere Solutions, Inc.  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\tpcccom.h*/
/*********************************************/
```

```
/* this ALWAYS GENERATED file contains the definitions for the interfaces */


 /* File created by MIDL compiler version 7.00.0499 */
/* at Fri Jul 04 15:21:33 2008
 */
/* Compiler settings for tpcccom.idl:
    Oicf, W1, Zp8, env=Win32 (32b run)
    protocol : dce , ms_ext, c_ext, robust
    error checks: allocation ref bounds_check enum stub_data
    VC __declspec() decoration level:
         __declspec(uuid()), __declspec(selectany), __declspec(novtable)
         DECLSPEC_UUID(), MIDL_INTERFACE()
*/
//@@MIDL_FILE_HEADING(  )

#pragma warning( disable: 4049 )  /* more than 64k source lines */


/* verify that the <rpcndr.h> version is high enough to compile this file*/
#ifndef __REQUIRED_RPCNDR_H_VERSION__
#define __REQUIRED_RPCNDR_H_VERSION__ 475
#endif

#include "rpc.h"
#include "rpcndr.h"

#ifndef __RPCNDR_H_VERSION__
#error this stub requires an updated version of <rpcndr.h>
#endif // __RPCNDR_H_VERSION__

#ifndef COM_NO_WINDOWS_H
#include "windows.h"
#include "ole2.h"
#endif /*COM_NO_WINDOWS_H*/

#ifndef __tpcccom_h__
#define __tpcccom_h__

#if defined(_MSC_VER) && (_MSC_VER >= 1020)
#pragma once
#endif

/* Forward Declarations */

#ifndef __ITPCCTran_FWD_DEFINED__
#define __ITPCCTran_FWD_DEFINED__
typedef interface ITPCCTran ITPCCTran;
#endif     /* __ITPCCTran_FWD_DEFINED__ */


#ifndef __TPCCTran_FWD_DEFINED__
#define __TPCCTran_FWD_DEFINED__

#ifdef __cplusplus
typedef class TPCCTran TPCCTran;
#else
typedef struct TPCCTran TPCCTran;
#endif /* __cplusplus */

#endif     /* __TPCCTran_FWD_DEFINED__ */


/* header files for imported files */
```

```
#include "oaidl.h"
#include "ocidl.h"

#ifdef __cplusplus
extern "C"{
#endif


#ifndef __ITPCCTran_INTERFACE_DEFINED__
#define __ITPCCTran_INTERFACE_DEFINED__

/* interface ITPCCTran */
/* [unique][helpstring][uuid][object] */


EXTERN_C const IID IID_ITPCCTran;

#if defined(__cplusplus) && !defined(CINTERFACE)

    MIDL_INTERFACE("53678C9E-CEAD-4D44-85DA-E1FE89D32C40")
    ITPCCTran : public IUnknown
    {
    public:
        virtual HRESULT STDMETHODCALLTYPE initialization_complete( void) = 0;

        virtual HRESULT STDMETHODCALLTYPE new_order(
            /* [in] */ int data_len,
            /* [size_is][out][in] */ unsigned char *param) = 0;

        virtual HRESULT STDMETHODCALLTYPE payment(
            /* [in] */ int data_len,
            /* [size_is][out][in] */ unsigned char *param) = 0;

        virtual HRESULT STDMETHODCALLTYPE order_status(
            /* [in] */ int data_len,
            /* [size_is][out][in] */ unsigned char *param) = 0;

        virtual HRESULT STDMETHODCALLTYPE delivery(
            /* [in] */ int data_len,
            /* [size_is][out][in] */ unsigned char *param) = 0;

        virtual HRESULT STDMETHODCALLTYPE stock_level(
            /* [in] */ int data_len,
            /* [size_is][out][in] */ unsigned char *param) = 0;

        virtual HRESULT STDMETHODCALLTYPE get_maximum(
            /* [in] */ int data_len,
            /* [size_is][out] */ unsigned char *param) = 0;

    };

#else      /* C style interface */

    typedef struct ITPCCTranVtbl
    {
        BEGIN_INTERFACE

        HRESULT ( STDMETHODCALLTYPE *QueryInterface )(
            ITPCCTran * This,
            /* [in] */ REFIID riid,
            /* [iid_is][out] */
            __RPC__deref_out  void **ppvObject);

        ULONG ( STDMETHODCALLTYPE *AddRef )(
            ITPCCTran * This);

        ULONG ( STDMETHODCALLTYPE *Release )(
```

```
        ITPCCTran * This);

    HRESULT ( STDMETHODCALLTYPE *initialization_complete )(
        ITPCCTran * This);

    HRESULT ( STDMETHODCALLTYPE *new_order )(
        ITPCCTran * This,
        /* [in] */ int data_len,
        /* [size_is][out][in] */ unsigned char *param);

    HRESULT ( STDMETHODCALLTYPE *payment )(
        ITPCCTran * This,
        /* [in] */ int data_len,
        /* [size_is][out][in] */ unsigned char *param);

    HRESULT ( STDMETHODCALLTYPE *order_status )(
        ITPCCTran * This,
        /* [in] */ int data_len,
        /* [size_is][out][in] */ unsigned char *param);

    HRESULT ( STDMETHODCALLTYPE *delivery )(
        ITPCCTran * This,
        /* [in] */ int data_len,
        /* [size_is][out][in] */ unsigned char *param);

    HRESULT ( STDMETHODCALLTYPE *stock_level )(
        ITPCCTran * This,
        /* [in] */ int data_len,
        /* [size_is][out][in] */ unsigned char *param);

    HRESULT ( STDMETHODCALLTYPE *get_maximum )(
        ITPCCTran * This,
        /* [in] */ int data_len,
        /* [size_is][out] */ unsigned char *param);

    END_INTERFACE
} ITPCCTranVtbl;

interface ITPCCTran
{
    CONST_VTBL struct ITPCCTranVtbl *lpVtbl;
};


#ifdef COBJMACROS


#define ITPCCTran_QueryInterface(This,riid,ppvObject)       \
    ( (This)->lpVtbl -> QueryInterface(This,riid,ppvObject) )

#define ITPCCTran_AddRef(This)          \
    ( (This)->lpVtbl -> AddRef(This) )

#define ITPCCTran_Release(This)         \
    ( (This)->lpVtbl -> Release(This) )


#define ITPCCTran_initialization_complete(This)    \
    ( (This)->lpVtbl -> initialization_complete(This) )

#define ITPCCTran_new_order(This,data_len,param)          \
    ( (This)->lpVtbl -> new_order(This,data_len,param) )

#define ITPCCTran_payment(This,data_len,param)            \
    ( (This)->lpVtbl -> payment(This,data_len,param) )
```

```
#define ITPCCTran_order_status(This,data_len,param)        \
    ( (This)->lpVtbl -> order_status(This,data_len,param) )

#define ITPCCTran_delivery(This,data_len,param) \
    ( (This)->lpVtbl -> delivery(This,data_len,param) )

#define ITPCCTran_stock_level(This,data_len,param)          \
    ( (This)->lpVtbl -> stock_level(This,data_len,param) )

#define ITPCCTran_get_maximum(This,data_len,param)      \
    ( (This)->lpVtbl -> get_maximum(This,data_len,param) )

#endif /* COBJMACROS */


#endif     /* C style interface */



#endif     /* __ITPCCTran_INTERFACE_DEFINED__ */


#ifndef __TPCCCOMLib_LIBRARY_DEFINED__
#define __TPCCCOMLib_LIBRARY_DEFINED__

/* library TPCCCOMLib */
/* [helpstring][version][uuid] */


EXTERN_C const IID LIBID_TPCCCOMLib;

EXTERN_C const CLSID CLSID_TPCCTran;

#ifdef __cplusplus

class DECLSPEC_UUID("FE7BD69A-501F-42FC-9921-7C6D95C660EA")
TPCCTran;
#endif
#endif /* __TPCCCOMLib_LIBRARY_DEFINED__ */

/* Additional Prototypes for ALL interfaces */

/* end of Additional Prototypes */

#ifdef __cplusplus
}
#endif

#endif




/*c:\original\kit\tpcccom_i.c*/
/*********************************************/


/* this ALWAYS GENERATED file contains the IIDs and CLSIDs */

/* link this file in with the server and any clients */
```

```
 /* File created by MIDL compiler version 7.00.0499 */
/* at Fri Jul 04 15:21:33 2008
 */
/* Compiler settings for tpcccom.idl:
    Oicf, W1, Zp8, env=Win32 (32b run)
    protocol : dce , ms_ext, c_ext, robust
    error checks: allocation ref bounds_check enum stub_data
    VC __declspec() decoration level:
         __declspec(uuid()), __declspec(selectany), __declspec(novtable)
         DECLSPEC_UUID(), MIDL_INTERFACE()
*/
//@@MIDL_FILE_HEADING(  )

#pragma warning( disable: 4049 )  /* more than 64k source lines */


#ifdef __cplusplus
extern "C"{
#endif


#include <rpc.h>
#include <rpcndr.h>

#ifdef _MIDL_USE_GUIDDEF_

#ifndef INITGUID
#define INITGUID
#include <guiddef.h>
#undef INITGUID
#else
#include <guiddef.h>
#endif

#define MIDL_DEFINE_GUID(type,name,l,w1,w2,b1,b2,b3,b4,b5,b6,b7,b8) \
        DEFINE_GUID(name,l,w1,w2,b1,b2,b3,b4,b5,b6,b7,b8)

#else // !_MIDL_USE_GUIDDEF_

#ifndef __IID_DEFINED__
#define __IID_DEFINED__

typedef struct _IID
{
    unsigned long x;
    unsigned short s1;
    unsigned short s2;
    unsigned char  c[8];
} IID;

#endif // __IID_DEFINED__

#ifndef CLSID_DEFINED
#define CLSID_DEFINED
typedef IID CLSID;
#endif // CLSID_DEFINED

#define MIDL_DEFINE_GUID(type,name,l,w1,w2,b1,b2,b3,b4,b5,b6,b7,b8) \
        const type name = {l,w1,w2,{b1,b2,b3,b4,b5,b6,b7,b8}}

#endif !_MIDL_USE_GUIDDEF_

MIDL_DEFINE_GUID(IID, IID_ITPCCTran,0x53678C9E,0xCEAD,0x4D44,0x85,0xDA,0xE1,0xFE,0x89,0xD3,0x2C,0x40);


MIDL_DEFINE_GUID(IID, LIBID_TPCCCOMLib,0x7FC7DA14,0x5758,0x4E75,0xBE,0x2B,0xF1,0x6D,0xAE,0x61,0xBA,0x79);
```

```
MIDL_DEFINE_GUID(CLSID, CLSID_TPCCTran,0xFE7BD69A,0x501F,0x42FC,0x99,0x21,0x7C,0x6D,0x95,0xC6,0x60,0xEA);

#undef MIDL_DEFINE_GUID

#ifdef __cplusplus
}
#endif







/*c:\original\kit\tpcccom_p.c*/
/*********************************************/



/* this ALWAYS GENERATED file contains the proxy stub code */


 /* File created by MIDL compiler version 7.00.0499 */
/* at Fri Jul 04 15:21:33 2008
 */
/* Compiler settings for tpcccom.idl:
    Oicf, W1, Zp8, env=Win32 (32b run)
    protocol : dce , ms_ext, c_ext, robust
    error checks: allocation ref bounds_check enum stub_data
    VC __declspec() decoration level:
         __declspec(uuid()), __declspec(selectany), __declspec(novtable)
         DECLSPEC_UUID(), MIDL_INTERFACE()
*/
//@@MIDL_FILE_HEADING(  )

#if !defined(_M_IA64) && !defined(_M_AMD64)


#pragma warning( disable: 4049 )  /* more than 64k source lines */
#if _MSC_VER >= 1200
#pragma warning(push)
#endif

#pragma warning( disable: 4211 )  /* redefine extern to static */
#pragma warning( disable: 4232 )  /* dllimport identity*/
#pragma warning( disable: 4024 )  /* array to pointer mapping*/
#pragma warning( disable: 4152 )  /* function/data pointer conversion in expression */
#pragma warning( disable: 4100 ) /* unreferenced arguments in x86 call */

#pragma optimize("", off )

#define USE_STUBLESS_PROXY


/* verify that the <rpcproxy.h> version is high enough to compile this file*/
#ifndef __REDQ_RPCPROXY_H_VERSION__
#define __REQUIRED_RPCPROXY_H_VERSION__ 475
#endif


#include "rpcproxy.h"
#ifndef __RPCPROXY_H_VERSION__
#error this stub requires an updated version of <rpcproxy.h>
#endif // __RPCPROXY_H_VERSION__
```

```
#include "tpcccom.h"

#define TYPE_FORMAT_STRING_SIZE   19
#define PROC_FORMAT_STRING_SIZE   283
#define EXPR_FORMAT_STRING_SIZE   1
#define TRANSMIT_AS_TABLE_SIZE    0
#define WIRE_MARSHAL_TABLE_SIZE   0

typedef struct _tpcccom_MIDL_TYPE_FORMAT_STRING
    {
    short          Pad;
    unsigned char  Format[ TYPE_FORMAT_STRING_SIZE ];
    } tpcccom_MIDL_TYPE_FORMAT_STRING;

typedef struct _tpcccom_MIDL_PROC_FORMAT_STRING
    {
    short          Pad;
    unsigned char  Format[ PROC_FORMAT_STRING_SIZE ];
    } tpcccom_MIDL_PROC_FORMAT_STRING;

typedef struct _tpcccom_MIDL_EXPR_FORMAT_STRING
    {
    long          Pad;
    unsigned char  Format[ EXPR_FORMAT_STRING_SIZE ];
    } tpcccom_MIDL_EXPR_FORMAT_STRING;


static RPC_SYNTAX_IDENTIFIER  _RpcTransferSyntax =
{{0x8A885D04,0x1CEB,0x11C9,{0x9F,0xE8,0x08,0x00,0x2B,0x10,0x48,0x60}},{2,0}};


extern const tpcccom_MIDL_TYPE_FORMAT_STRING tpcccom__MIDL_TypeFormatString;
extern const tpcccom_MIDL_PROC_FORMAT_STRING tpcccom__MIDL_ProcFormatString;
extern const tpcccom_MIDL_EXPR_FORMAT_STRING tpcccom__MIDL_ExprFormatString;


extern const MIDL_STUB_DESC Object_StubDesc;


extern const MIDL_SERVER_INFO ITPCCTran_ServerInfo;
extern const MIDL_STUBLESS_PROXY_INFO ITPCCTran_ProxyInfo;



#if !defined(__RPC_WIN32__)
#error  Invalid build platform for this stub.
#endif

#if !(TARGET_IS_NT50_OR_LATER)
#error You need a Windows 2000 or later to run this stub because it uses these features:
#error   /robust command line switch.
#error However, your C/C++ compilation flags indicate you intend to run this app on earlier systems.
#error This app will fail with the RPC_X_WRONG_STUB_VERSION error.
#endif


static const tpcccom_MIDL_PROC_FORMAT_STRING tpcccom__MIDL_ProcFormatString =
    {
        0,
        {

            /* Procedure initialization_complete */

                                0x33,               /* FC_AUTO_HANDLE */
                                0x6c,               /* Old Flags:  object, Oi2 */
```

```
/*  2 */   NdrFcLong( 0x0 ),   /* 0 */
/*  6 */   NdrFcShort( 0x3 ),  /* 3 */
/*  8 */   NdrFcShort( 0x8 ),  /* x86 Stack size/offset = 8 */
/* 10 */   NdrFcShort( 0x0 ),  /* 0 */
/* 12 */   NdrFcShort( 0x8 ),  /* 8 */
/* 14 */   0x44,               /* Oi2 Flags:  has return, has ext, */
                               0x1,                /* 1 */
/* 16 */   0x8,                /* 8 */
                               0x1,                /* Ext Flags:  new corr desc, */
/* 18 */   NdrFcShort( 0x0 ),  /* 0 */
/* 20 */   NdrFcShort( 0x0 ),  /* 0 */
/* 22 */   NdrFcShort( 0x0 ),  /* 0 */

           /* Return value */

/* 24 */   NdrFcShort( 0x70 ), /* Flags:  out, return, base type, */
/* 26 */   NdrFcShort( 0x4 ),  /* x86 Stack size/offset = 4 */
/* 28 */   0x8,                /* FC_LONG */
                               0x0,                /* 0 */

           /* Procedure new_order */

/* 30 */   0x33,               /* FC_AUTO_HANDLE */
                               0x6c,               /* Old Flags:  object, Oi2 */
/* 32 */   NdrFcLong( 0x0 ),   /* 0 */
/* 36 */   NdrFcShort( 0x4 ),  /* 4 */
/* 38 */   NdrFcShort( 0x10 ), /* x86 Stack size/offset = 16 */
/* 40 */   NdrFcShort( 0x8 ),  /* 8 */
/* 42 */   NdrFcShort( 0x8 ),  /* 8 */
/* 44 */   0x47,               /* Oi2 Flags:  srv must size, clt must size, has return, has ext, */
                               0x3,                /* 3 */
/* 46 */   0x8,                /* 8 */
                               0x7,                /* Ext Flags:  new corr desc, clt corr check, srv corr check, */
/* 48 */   NdrFcShort( 0x1 ),  /* 1 */
/* 50 */   NdrFcShort( 0x1 ),  /* 1 */
/* 52 */   NdrFcShort( 0x0 ),  /* 0 */

           /* Parameter data_len */

/* 54 */   NdrFcShort( 0x48 ), /* Flags:  in, base type, */
/* 56 */   NdrFcShort( 0x4 ),  /* x86 Stack size/offset = 4 */
/* 58 */   0x8,                /* FC_LONG */
                               0x0,                /* 0 */

           /* Parameter param */

/* 60 */   NdrFcShort( 0x11b ),        /* Flags:  must size, must free, in, out, simple ref, */
/* 62 */   NdrFcShort( 0x8 ),  /* x86 Stack size/offset = 8 */
/* 64 */   NdrFcShort( 0x6 ),  /* Type Offset=6 */

           /* Return value */

/* 66 */   NdrFcShort( 0x70 ), /* Flags:  out, return, base type, */
/* 68 */   NdrFcShort( 0xc ),  /* x86 Stack size/offset = 12 */
/* 70 */   0x8,                /* FC_LONG */
                               0x0,                /* 0 */

           /* Procedure payment */

/* 72 */   0x33,               /* FC_AUTO_HANDLE */
                               0x6c,               /* Old Flags:  object, Oi2 */
/* 74 */   NdrFcLong( 0x0 ),   /* 0 */
/* 78 */   NdrFcShort( 0x5 ),  /* 5 */
/* 80 */   NdrFcShort( 0x10 ), /* x86 Stack size/offset = 16 */
/* 82 */   NdrFcShort( 0x8 ),  /* 8 */
/* 84 */   NdrFcShort( 0x8 ),  /* 8 */
/* 86 */   0x47,               /* Oi2 Flags:  srv must size, clt must size, has return, has ext, */
```

```
                                0x3,                    /* 3 */
/* 88 */   0x8,                 /* 8 */
                                0x7,                    /* Ext Flags:  new corr desc, clt corr check, srv corr check, */
/* 90 */   NdrFcShort( 0x1 ),   /* 1 */
/* 92 */   NdrFcShort( 0x1 ),   /* 1 */
/* 94 */   NdrFcShort( 0x0 ),   /* 0 */

        /* Parameter data_len */

/* 96 */   NdrFcShort( 0x48 ), /* Flags:  in, base type, */
/* 98 */   NdrFcShort( 0x4 ),   /* x86 Stack size/offset = 4 */
/* 100 */ 0x8,                  /* FC_LONG */
                                0x0,                    /* 0 */

        /* Parameter param */

/* 102 */ NdrFcShort( 0x11b ),          /* Flags:  must size, must free, in, out, simple ref, */
/* 104 */ NdrFcShort( 0x8 ),   /* x86 Stack size/offset = 8 */
/* 106 */ NdrFcShort( 0x6 ),   /* Type Offset=6 */

        /* Return value */

/* 108 */ NdrFcShort( 0x70 ), /* Flags:  out, return, base type, */
/* 110 */ NdrFcShort( 0xc ),   /* x86 Stack size/offset = 12 */
/* 112 */ 0x8,                  /* FC_LONG */
                                0x0,                    /* 0 */

        /* Procedure order_status */

/* 114 */ 0x33,                 /* FC_AUTO_HANDLE */
                                0x6c,                   /* Old Flags:  object, Oi2 */
/* 116 */ NdrFcLong( 0x0 ),    /* 0 */
/* 120 */ NdrFcShort( 0x6 ),   /* 6 */
/* 122 */ NdrFcShort( 0x10 ), /* x86 Stack size/offset = 16 */
/* 124 */ NdrFcShort( 0x8 ),   /* 8 */
/* 126 */ NdrFcShort( 0x8 ),   /* 8 */
/* 128 */ 0x47,                 /* Oi2 Flags:  srv must size, clt must size, has return, has ext, */
                                0x3,                    /* 3 */
/* 130 */ 0x8,                  /* 8 */
                                0x7,                    /* Ext Flags:  new corr desc, clt corr check, srv corr check, */
/* 132 */ NdrFcShort( 0x1 ),   /* 1 */
/* 134 */ NdrFcShort( 0x1 ),   /* 1 */
/* 136 */ NdrFcShort( 0x0 ),   /* 0 */

        /* Parameter data_len */

/* 138 */ NdrFcShort( 0x48 ), /* Flags:  in, base type, */
/* 140 */ NdrFcShort( 0x4 ),   /* x86 Stack size/offset = 4 */
/* 142 */ 0x8,                  /* FC_LONG */
                                0x0,                    /* 0 */

        /* Parameter param */

/* 144 */ NdrFcShort( 0x11b ),          /* Flags:  must size, must free, in, out, simple ref, */
/* 146 */ NdrFcShort( 0x8 ),   /* x86 Stack size/offset = 8 */
/* 148 */ NdrFcShort( 0x6 ),   /* Type Offset=6 */

        /* Return value */

/* 150 */ NdrFcShort( 0x70 ), /* Flags:  out, return, base type, */
/* 152 */ NdrFcShort( 0xc ),   /* x86 Stack size/offset = 12 */
/* 154 */ 0x8,                  /* FC_LONG */
                                0x0,                    /* 0 */

        /* Procedure delivery */

/* 156 */ 0x33,                 /* FC_AUTO_HANDLE */
```

```
                            0x6c,               /* Old Flags:  object, Oi2 */
/* 158 */ NdrFcLong( 0x0 ),  /* 0 */
/* 162 */ NdrFcShort( 0x7 ), /* 7 */
/* 164 */ NdrFcShort( 0x10 ),/* x86 Stack size/offset = 16 */
/* 166 */ NdrFcShort( 0x8 ), /* 8 */
/* 168 */ NdrFcShort( 0x8 ), /* 8 */
/* 170 */ 0x47,              /* Oi2 Flags:  srv must size, clt must size, has return, has ext, */
                            0x3,                /* 3 */
/* 172 */ 0x8,               /* 8 */
                            0x7,                /* Ext Flags:  new corr desc, clt corr check, srv corr check, */
/* 174 */ NdrFcShort( 0x1 ), /* 1 */
/* 176 */ NdrFcShort( 0x1 ), /* 1 */
/* 178 */ NdrFcShort( 0x0 ), /* 0 */

        /* Parameter data_len */

/* 180 */ NdrFcShort( 0x48 ),/* Flags:  in, base type, */
/* 182 */ NdrFcShort( 0x4 ), /* x86 Stack size/offset = 4 */
/* 184 */ 0x8,               /* FC_LONG */
                            0x0,                /* 0 */

        /* Parameter param */

/* 186 */ NdrFcShort( 0x11b ),         /* Flags:  must size, must free, in, out, simple ref, */
/* 188 */ NdrFcShort( 0x8 ), /* x86 Stack size/offset = 8 */
/* 190 */ NdrFcShort( 0x6 ), /* Type Offset=6 */

        /* Return value */

/* 192 */ NdrFcShort( 0x70 ),/* Flags:  out, return, base type, */
/* 194 */ NdrFcShort( 0xc ), /* x86 Stack size/offset = 12 */
/* 196 */ 0x8,               /* FC_LONG */
                            0x0,                /* 0 */

        /* Procedure stock_level */

/* 198 */ 0x33,              /* FC_AUTO_HANDLE */
                            0x6c,               /* Old Flags:  object, Oi2 */
/* 200 */ NdrFcLong( 0x0 ),  /* 0 */
/* 204 */ NdrFcShort( 0x8 ), /* 8 */
/* 206 */ NdrFcShort( 0x10 ),/* x86 Stack size/offset = 16 */
/* 208 */ NdrFcShort( 0x8 ), /* 8 */
/* 210 */ NdrFcShort( 0x8 ), /* 8 */
/* 212 */ 0x47,              /* Oi2 Flags:  srv must size, clt must size, has return, has ext, */
                            0x3,                /* 3 */
/* 214 */ 0x8,               /* 8 */
                            0x7,                /* Ext Flags:  new corr desc, clt corr check, srv corr check, */
/* 216 */ NdrFcShort( 0x1 ), /* 1 */
/* 218 */ NdrFcShort( 0x1 ), /* 1 */
/* 220 */ NdrFcShort( 0x0 ), /* 0 */

        /* Parameter data_len */

/* 222 */ NdrFcShort( 0x48 ),/* Flags:  in, base type, */
/* 224 */ NdrFcShort( 0x4 ), /* x86 Stack size/offset = 4 */
/* 226 */ 0x8,               /* FC_LONG */
                            0x0,                /* 0 */

        /* Parameter param */

/* 228 */ NdrFcShort( 0x11b ),         /* Flags:  must size, must free, in, out, simple ref, */
/* 230 */ NdrFcShort( 0x8 ), /* x86 Stack size/offset = 8 */
/* 232 */ NdrFcShort( 0x6 ), /* Type Offset=6 */

        /* Return value */

/* 234 */ NdrFcShort( 0x70 ),/* Flags:  out, return, base type, */
```

```
/* 236 */ NdrFcShort( 0xc ),   /* x86 Stack size/offset = 12 */
/* 238 */ 0x8,                 /* FC_LONG */
                               0x0,                 /* 0 */

          /* Procedure get_maximum */

/* 240 */ 0x33,                /* FC_AUTO_HANDLE */
                               0x6c,                /* Old Flags:  object, Oi2 */
/* 242 */ NdrFcLong( 0x0 ),    /* 0 */
/* 246 */ NdrFcShort( 0x9 ),   /* 9 */
/* 248 */ NdrFcShort( 0x10 ), /* x86 Stack size/offset = 16 */
/* 250 */ NdrFcShort( 0x8 ),   /* 8 */
/* 252 */ NdrFcShort( 0x8 ),   /* 8 */
/* 254 */ 0x45,                /* Oi2 Flags:  srv must size, has return, has ext, */
                               0x3,                 /* 3 */
/* 256 */ 0x8,                 /* 8 */
                               0x3,                 /* Ext Flags:  new corr desc, clt corr check, */
/* 258 */ NdrFcShort( 0x1 ),   /* 1 */
/* 260 */ NdrFcShort( 0x0 ),   /* 0 */
/* 262 */ NdrFcShort( 0x0 ),   /* 0 */

          /* Parameter data_len */

/* 264 */ NdrFcShort( 0x48 ), /* Flags:  in, base type, */
/* 266 */ NdrFcShort( 0x4 ),   /* x86 Stack size/offset = 4 */
/* 268 */ 0x8,                 /* FC_LONG */
                               0x0,                 /* 0 */

          /* Parameter param */

/* 270 */ NdrFcShort( 0x113 ),          /* Flags:  must size, must free, out, simple ref, */
/* 272 */ NdrFcShort( 0x8 ),   /* x86 Stack size/offset = 8 */
/* 274 */ NdrFcShort( 0x6 ),   /* Type Offset=6 */

          /* Return value */

/* 276 */ NdrFcShort( 0x70 ), /* Flags:  out, return, base type, */
/* 278 */ NdrFcShort( 0xc ),   /* x86 Stack size/offset = 12 */
/* 280 */ 0x8,                 /* FC_LONG */
                               0x0,                 /* 0 */

                               0x0
      }
   };

static const tpcccom_MIDL_TYPE_FORMAT_STRING tpcccom__MIDL_TypeFormatString =
   {
      0,
      {
                               NdrFcShort( 0x0 ),   /* 0 */
/*  2 */
                               0x11, 0x0,           /* FC_RP */
/*  4 */   NdrFcShort( 0x2 ),  /* Offset= 2 (6) */
/*  6 */
                               0x1b,                /* FC_CARRAY */
                               0x0,                 /* 0 */
/*  8 */   NdrFcShort( 0x1 ),  /* 1 */
/* 10 */   0x28,               /* Corr desc:  parameter, FC_LONG */
                               0x0,                 /*  */
/* 12 */   NdrFcShort( 0x4 ),  /* x86 Stack size/offset = 4 */
/* 14 */   NdrFcShort( 0x1 ),  /* Corr flags:  early, */
/* 16 */   0x2,                /* FC_CHAR */
                               0x5b,                /* FC_END */

                               0x0
      }
   };
```

```
/* Object interface: IUnknown, ver. 0.0,
   GUID={0x00000000,0x0000,0x0000,{0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46}} */


/* Object interface: ITPCCTran, ver. 0.0,
   GUID={0x53678C9E,0xCEAD,0x4D44,{0x85,0xDA,0xE1,0xFE,0x89,0xD3,0x2C,0x40}} */

#pragma code_seg(".orpc")
static const unsigned short ITPCCTran_FormatStringOffsetTable[] =
    {
    0,
    30,
    72,
    114,
    156,
    198,
    240
    };

static const MIDL_STUBLESS_PROXY_INFO ITPCCTran_ProxyInfo =
    {
    &Object_StubDesc,
    tpcccom__MIDL_ProcFormatString.Format,
    &ITPCCTran_FormatStringOffsetTable[-3],
    0,
    0,
    0
    };


static const MIDL_SERVER_INFO ITPCCTran_ServerInfo =
    {
    &Object_StubDesc,
    0,
    tpcccom__MIDL_ProcFormatString.Format,
    &ITPCCTran_FormatStringOffsetTable[-3],
    0,
    0,
    0,
    0};
CINTERFACE_PROXY_VTABLE(10) _ITPCCTranProxyVtbl =
{
    &ITPCCTran_ProxyInfo,
    &IID_ITPCCTran,
    IUnknown_QueryInterface_Proxy,
    IUnknown_AddRef_Proxy,
    IUnknown_Release_Proxy ,
    (void *) (INT_PTR) -1 /* ITPCCTran::initialization_complete */ ,
    (void *) (INT_PTR) -1 /* ITPCCTran::new_order */ ,
    (void *) (INT_PTR) -1 /* ITPCCTran::payment */ ,
    (void *) (INT_PTR) -1 /* ITPCCTran::order_status */ ,
    (void *) (INT_PTR) -1 /* ITPCCTran::delivery */ ,
    (void *) (INT_PTR) -1 /* ITPCCTran::stock_level */ ,
    (void *) (INT_PTR) -1 /* ITPCCTran::get_maximum */
};

const CInterfaceStubVtbl _ITPCCTranStubVtbl =
{
    &IID_ITPCCTran,
    &ITPCCTran_ServerInfo,
    10,
    0, /* pure interpreted */
    CStdStubBuffer_METHODS
};
```

```
static const MIDL_STUB_DESC Object_StubDesc =
    {
    0,
    NdrOleAllocate,
    NdrOleFree,
    0,
    0,
    0,
    0,
    0,
    tpcccom__MIDL_TypeFormatString.Format,
    1, /* -error bounds_check flag */
    0x50002, /* Ndr library version */
    0,
    0x70001f3, /* MIDL Version 7.0.499 */
    0,
    0,
    0,  /* notify & notify_flag routine table */
    0x1, /* MIDL flag */
    0, /* cs routines */
    0,   /* proxy/server info */
    0
    };

const CInterfaceProxyVtbl * _tpcccom_ProxyVtblList[] =
{
    ( CInterfaceProxyVtbl *) &_ITPCCTranProxyVtbl,
    0
};

const CInterfaceStubVtbl * _tpcccom_StubVtblList[] =
{
    ( CInterfaceStubVtbl *) &_ITPCCTranStubVtbl,
    0
};

PCInterfaceName const _tpcccom_InterfaceNamesList[] =
{
    "ITPCCTran",
    0
};


#define _tpcccom_CHECK_IID(n)          IID_GENERIC_CHECK_IID( _tpcccom, pIID, n)

int __stdcall _tpcccom_IID_Lookup( const IID * pIID, int * pIndex )
{

    if(!_tpcccom_CHECK_IID(0))
        {
        *pIndex = 0;
        return 1;
        }

    return 0;
}

const ExtendedProxyFileInfo tpcccom_ProxyFileInfo =
{
    (PCInterfaceProxyVtblList *) & _tpcccom_ProxyVtblList,
    (PCInterfaceStubVtblList *) & _tpcccom_StubVtblList,
    (const PCInterfaceName * ) & _tpcccom_InterfaceNamesList,
    0, // no delegation
    & _tpcccom_IID_Lookup,
    1,
    2,
    0, /* table of [async_uuid] interfaces */
```

```
    0, /* Filler1 */
    0, /* Filler2 */
    0  /* Filler3 */
};
#pragma optimize("", on )
#if _MSC_VER >= 1200
#pragma warning(pop)
#endif


#endif /* !defined(_M_IA64) && !defined(_M_AMD64)*/




/*c:\original\kit\tpccisapi.cpp*/
/************************************************/

// ********************************************************************
// Copyright 2002-2008 iAnywhere Solutions, Inc.  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\Sybase\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",
};

// 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\Sybase\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\\Sybase\\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, 50000 );
}

BOOL WINAPI DllMain( HINSTANCE      hinstDll,
                     DWORD          dwReason,
                     LPVOID         lpvContext )
/*****************************************/
{
    int i;
    char buffer[200];
    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"
                                "\tUsing Worker Threads: %s, # Worker Threads: %d\n"
                                "\tMax Deliveries: %d, # Delivery Threads: %d\n"
                                "\tMax Terminals: %d\n",
                                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"
                         "\tNum Terminals: %d, Num Deliveries: %d\n",
                         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 )
/*********************************************/
{
    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 );
            free( deliveries );
    } 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",

// 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",

// 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",

// 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"
};

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\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\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;
    unsigned long           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++;
    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\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",
                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;
   unsigned long                  key;

   InterlockedIncrement( &numWorkerThreadsRunning );
#ifdef SERVER_COM
   CoInitializeEx( NULL, COINIT_MULTITHREADED );
#endif
   for( ; ; ) {
         GetQueuedCompletionStatus( workerCompletionPort,
                                    &req_type,
                                    &key,
                                    (LPOVERLAPPED *)&pECB,
                                    INFINITE );
         if( pECB == NULL ) break;
         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 );
   }
   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
         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.c*/
/************************************************/

#define _SQL_OS_WINNT
// ************************************************
// ********* GENERATED FILE - DO NOT EDIT! *********
// ************************************************

#define _SQL_SQLPP_VERSION_MAJOR 11
#define _SQL_SQLPP_VERSION_MINOR 0
#define _SQL_SQLPP_VERSION_MAINT 0
#define _SQL_SQLPP_DBLIB_VERSION 11
// ************************************************************************
// Copyright 2002 iAnywhere Solutions, Inc.  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
#endif

#include <time.h>
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include "random.hpp"
#include "sqldef.h"
//#include <itoa.h>
#if defined( UNIX )
    #include "clibext.h"
#endif

#define _unused( x ) ( x = x )
#define MAXITEMS      100000
#define CUST_PER_DIST 3000
#define DIST_PER_WARE 10
#define ORD_PER_DIST  3000

#define    LOADER_NURAND_C            123
```

```
/* EXEC SQL INCLUDE SQLCA; */
#include "sqlca.h"
#include "sqldef.h"
#ifdef __cplusplus
    extern "C" {
#endif
extern SQLCA sqlca;
extern SQLCA *sqlcaptr;
extern short int _ESQL_Version11_;
extern short int _ESQL_OS_WINNT_;
#ifdef __cplusplus
    }
#endif
#include "sqlda.h"
static char __SQLV_tpccld_1[] = "DBA";
static char __SQLV_tpccld_2[] = "sql";
static char __SQLV_tpccld_4[] = "INSERT INTO\n item (i_id, i_im_id, i_name, i_price, i_data)\n values (:i_id, :i_im_id, :i_name,
:i_price, :i_data)";
static char __SQLV_tpccld_6[] = "INSERT INTO\n warehouse (w_id, w_name,\n w_street_1, w_street_2, w_city, w_state, w_zip,\n
w_tax, w_ytd)\n values (:w_id, :w_name,\n :w_street_1, :w_street_2, :w_city, :w_state,\n :w_zip, :w_tax, :w_ytd)";
static char __SQLV_tpccld_8[] = "INSERT INTO\n stock (s_i_id, s_w_id, s_quantity,\n s_dist_01, s_dist_02, s_dist_03, s_dist_04,
s_dist_05,\n s_dist_06, s_dist_07, s_dist_08, s_dist_09, s_dist_10,\n s_ytd, s_order_cnt, s_remote_cnt, s_data)\n values (:s_i_id,
:s_w_id, :s_quantity,\n :s_dist_01, :s_dist_02, :s_dist_03, :s_dist_04, :s_dist_05,\n :s_dist_06, :s_dist_07, :s_dist_08, :s_dist_09,
:s_dist_10,\n 0, 0, 0, :s_data)";
static char __SQLV_tpccld_10[] = "INSERT INTO\n district (d_id, d_w_id, d_name,\n d_street_1, d_street_2, d_city, d_state,
d_zip,\n d_tax, d_ytd, d_next_o_id)\n values (:d_id, :d_w_id, :d_name,\n :d_street_1, :d_street_2, :d_city, :d_state, :d_zip,\n
:d_tax, :d_ytd, :d_next_o_id)";
static char __SQLV_tpccld_12[] = "INSERT INTO\n customer (c_id, c_d_id, c_w_id,\n c_first, c_middle, c_last,\n c_street_1,
c_street_2, c_city, c_state, c_zip,\n c_phone, c_since, c_credit,\n c_credit_lim, c_discount, c_balance, c_data1, c_data2,\n
c_ytd_payment, c_payment_cnt, c_delivery_cnt)\n values (:c_id, :c_d_id, :c_w_id,\n :c_first, :c_middle, :c_last,\n :c_street_1,
:c_street_2, :c_city, :c_state, :c_zip,\n :c_phone, :timestamp, :c_credit,\n :c_credit_lim, :c_discount, :c_balance, :c_data1,
:c_data2,\n 10.0, 1, 0) ";
static char __SQLV_tpccld_14[] = "INSERT INTO\n customer (c_id, c_d_id, c_w_id,\n c_first, c_middle, c_last,\n c_street_1,
c_street_2, c_city, c_state, c_zip,\n c_phone, c_since, c_credit,\n c_credit_lim, c_discount, c_balance, c_data,\n c_ytd_payment,
c_payment_cnt, c_delivery_cnt)\n values (:c_id, :c_d_id, :c_w_id,\n :c_first, :c_middle, :c_last,\n :c_street_1, :c_street_2, :c_city,
:c_state, :c_zip,\n :c_phone, :timestamp, :c_credit,\n :c_credit_lim, :c_discount, :c_balance, :c_data,\n 10.0, 1, 0) ";
static char __SQLV_tpccld_16[] = "INSERT INTO\n history (h_c_id, h_c_d_id, h_c_w_id,\n h_d_id, h_w_id, h_date, h_amount,
h_data)\n values (:c_id, :c_d_id, :c_w_id,\n :c_d_id, :c_w_id, :timestamp, :h_amount, :h_data)";
static char __SQLV_tpccld_18[] = "INSERT INTO\n orders (o_id, o_d_id, o_w_id, o_c_id,\n o_entry_d, o_carrier_id, o_ol_cnt,
o_all_local)\n values (:o_id, :o_d_id, :o_w_id, :o_c_id,\n :timestamp, NULL, :o_ol_cnt, 1)";
static char __SQLV_tpccld_20[] = "INSERT INTO\n new_order (no_o_id, no_d_id, no_w_id)\n values (:o_id, :o_d_id, :o_w_id)";
static char __SQLV_tpccld_22[] = "INSERT INTO\n orders (o_id, o_d_id, o_w_id, o_c_id,\n o_entry_d, o_carrier_id, o_ol_cnt,
o_all_local)\n values (:o_id, :o_d_id, :o_w_id, :o_c_id,\n :timestamp, :o_carrier_id, :o_ol_cnt, 1)";
static char __SQLV_tpccld_24[] = "INSERT INTO\n order_line (ol_o_id, ol_d_id, ol_w_id, ol_number,\n ol_i_id, ol_supply_w_id,
ol_quantity, ol_amount,\n ol_dist_info, ol_delivery_d)\n values (:o_id, :o_d_id, :o_w_id, :ol,\n :ol_i_id, :ol_supply_w_id,
:ol_quantity, :ol_amount,\n :ol_dist_info, NULL)";
static char __SQLV_tpccld_26[] = "INSERT INTO\n order_line (ol_o_id, ol_d_id, ol_w_id, ol_number,\n ol_i_id, ol_supply_w_id,
ol_quantity, ol_amount,\n ol_dist_info, ol_delivery_d)\n values (:o_id, :o_d_id, :o_w_id, :ol,\n :ol_i_id, :ol_supply_w_id,
:ol_quantity, :ol_amount,\n :ol_dist_info, :timestamp)";


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);
long            NURand(int iConst, long x, long y, long C);
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;

/*=================================================================+
 |      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; */


#if defined( UNIX )
    pthread_key_create( &thread_random_data,  NULL );
#endif

  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"; */
            {
```

```
            dbpp_connect_40( (sqlcaptr), __SQLV_tpccld_1, __SQLV_tpccld_2, SQLNULL, SQLNULL, SQLNULL );
if( (sqlcaptr)->sqlcode < 0 ) goto Error_SqlCall;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }

    }
    // sgenrand( time( 0 ) );
    sgenrand( 123 ); // 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; */
            {
            dbpp_commit( (sqlcaptr), 0 );
if( (sqlcaptr)->sqlcode < 0 ) goto Error_SqlCall;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }

            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); */
              {
              struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[5];}
__SQLV_tpccld_3    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(5-1)*sizeof(SQLDA_VARIABLE),    5, 5 };
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[0].sqltype = (unsigned short)DT_INT;
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[0].sqldata = (void *)(&(i_id));
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[0].sqlind = (short int *)((SQLNULL));
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[1].sqltype = (unsigned short)DT_INT;
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[1].sqldata = (void *)(&(i_im_id));
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[1].sqlind = (short int *)((SQLNULL));
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[2].sqltype = (unsigned short)DT_STRING;
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[2].sqllen = 25L;
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[2].sqldata = (void *)((i_name));
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[2].sqlind = (short int *)((SQLNULL));
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[3].sqltype = (unsigned short)DT_FLOAT;
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[3].sqldata = (void *)(&(i_price));
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[3].sqlind = (short int *)((SQLNULL));
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[4].sqltype = (unsigned short)DT_STRING;
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[4].sqllen = 51L;
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[4].sqldata = (void *)((i_data));
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[4].sqlind = (short int *)((SQLNULL));
```

```
            db_verify_version_11();
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_4, ((SQLDA *)&__SQLV_tpccld_3), (SQLDA *)0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }

      }
      if ( !(i_id % 100) ) {
               if( inserting ) {
                  printf(".");
                  /* EXEC SQL COMMIT WORK; */
               {
               dbpp_commit( (sqlcaptr), 0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
               }

                }
         if ( !(i_id % 5000) ) printf(" %ld\r",i_id);
      }
    }
    if( inserting ) {
               /* EXEC SQL COMMIT WORK; */
               {
               dbpp_commit( (sqlcaptr), 0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
               }

    } 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); */
            {
            struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[9];}
__SQLV_tpccld_5    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(9-1)*sizeof(SQLDA_VARIABLE),    9, 9 };
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[0].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[0].sqldata = (void *)(&(w_id));
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[0].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[1].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[1].sqllen = 11L;
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[1].sqldata = (void *)((w_name));
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[1].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[2].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[2].sqllen = 21L;
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[2].sqldata = (void *)((w_street_1));
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[2].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[3].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[3].sqllen = 21L;
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[3].sqldata = (void *)((w_street_2));
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[3].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[4].sqltype = (unsigned short)DT_STRING;
```

```
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[4].sqllen = 21L;
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[4].sqldata = (void *)((w_city));
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[4].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[5].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[5].sqllen = 3L;
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[5].sqldata = (void *)((w_state));
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[5].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[6].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[6].sqllen = 10L;
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[6].sqldata = (void *)((w_zip));
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[6].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[7].sqltype = (unsigned short)DT_FLOAT;
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[7].sqldata = (void *)(&(w_tax));
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[7].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[8].sqltype = (unsigned short)DT_FLOAT;
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[8].sqldata = (void *)(&(w_ytd));
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[8].sqlind = (short int *)((SQLNULL));
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_6, ((SQLDA *)&__SQLV_tpccld_5), (SQLDA *)0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }

     }
     /** Make Rows associated with Warehouse **/
     Stock(w_id);
     District(w_id);
   }
   if( inserting ) {
            /* EXEC SQL COMMIT WORK; */
            {
            dbpp_commit( (sqlcaptr), 0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }

   } 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; */
            {
            dbpp_commit( (sqlcaptr), 0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }
  /* 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; */
            {
            dbpp_commit( (sqlcaptr), 0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }
  /* 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); */
            {
            struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[14];}
__SQLV_tpccld_7    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(14-1)*sizeof(SQLDA_VARIABLE),    14, 14 };
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[0].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[0].sqldata = (void *)(&(s_i_id));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[0].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[1].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[1].sqldata = (void *)(&(s_w_id));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[1].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[2].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[2].sqldata = (void *)(&(s_quantity));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[2].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[3].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[3].sqllen = 25L;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[3].sqldata = (void *)((s_dist_01));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[3].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[4].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[4].sqllen = 25L;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[4].sqldata = (void *)((s_dist_02));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[4].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[5].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[5].sqllen = 25L;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[5].sqldata = (void *)((s_dist_03));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[5].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[6].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[6].sqllen = 25L;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[6].sqldata = (void *)((s_dist_04));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[6].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[7].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[7].sqllen = 25L;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[7].sqldata = (void *)((s_dist_05));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[7].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[8].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[8].sqllen = 25L;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[8].sqldata = (void *)((s_dist_06));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[8].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[9].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[9].sqllen = 25L;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[9].sqldata = (void *)((s_dist_07));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[9].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[10].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[10].sqllen = 25L;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[10].sqldata = (void *)((s_dist_08));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[10].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[11].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[11].sqllen = 25L;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[11].sqldata = (void *)((s_dist_09));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[11].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[12].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[12].sqllen = 25L;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[12].sqldata = (void *)((s_dist_10));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[12].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[13].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[13].sqllen = 51L;
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[13].sqldata = (void *)((s_data));
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[13].sqlind = (short int *)((SQLNULL));
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_8, ((SQLDA *)&__SQLV_tpccld_7), (SQLDA *)0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
```

```
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }

      }
      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; */
               {
               dbpp_commit( (sqlcaptr), 0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
               }

                }
      }
    }
    if( inserting ) {
               /* EXEC SQL COMMIT WORK; */
               {
               dbpp_commit( (sqlcaptr), 0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
               }
  /* 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); */
           {
           struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[11];}
__SQLV_tpccld_9    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(11-1)*sizeof(SQLDA_VARIABLE),    11, 11 };
           ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[0].sqltype = (unsigned short)DT_INT;
           ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[0].sqldata = (void *)(&(d_id));
           ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[0].sqlind = (short int *)((SQLNULL));
           ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[1].sqltype = (unsigned short)DT_INT;
           ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[1].sqldata = (void *)(&(d_w_id));
           ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[1].sqlind = (short int *)((SQLNULL));
           ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[2].sqltype = (unsigned short)DT_STRING;
           ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[2].sqllen = 11L;
           ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[2].sqldata = (void *)((d_name));
           ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[2].sqlind = (short int *)((SQLNULL));
           ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[3].sqltype = (unsigned short)DT_STRING;
           ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[3].sqllen = 21L;
           ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[3].sqldata = (void *)((d_street_1));
```

```
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[3].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[4].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[4].sqllen = 21L;
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[4].sqldata = (void *)((d_street_2));
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[4].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[5].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[5].sqllen = 21L;
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[5].sqldata = (void *)((d_city));
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[5].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[6].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[6].sqllen = 3L;
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[6].sqldata = (void *)((d_state));
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[6].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[7].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[7].sqllen = 10L;
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[7].sqldata = (void *)((d_zip));
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[7].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[8].sqltype = (unsigned short)DT_FLOAT;
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[8].sqldata = (void *)(&(d_tax));
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[8].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[9].sqltype = (unsigned short)DT_FLOAT;
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[9].sqldata = (void *)(&(d_ytd));
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[9].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[10].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[10].sqldata = (void *)(&(d_next_o_id));
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[10].sqlind = (short int *)((SQLNULL));
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_10, ((SQLDA *)&__SQLV_tpccld_9), (SQLDA *)0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }

     }

   }
   if( inserting ) {
            /* EXEC SQL COMMIT WORK; */
            {
            dbpp_commit( (sqlcaptr), 0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }

   }

   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(255,0,999,LOADER_NURAND_C),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) ; */
            {
            struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[19];}
__SQLV_tpccld_11    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(19-1)*sizeof(SQLDA_VARIABLE),    19, 19 };
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[0].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[0].sqldata = (void *)(&(c_id));
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[0].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[1].sqltype = (unsigned short)DT_INT;
```

```
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[1].sqldata = (void *)(&(c_d_id));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[1].sqlind = (short int *)((SQLNULL));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[2].sqltype = (unsigned short)DT_INT;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[2].sqldata = (void *)(&(c_w_id));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[2].sqlind = (short int *)((SQLNULL));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[3].sqltype = (unsigned short)DT_STRING;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[3].sqllen = 17L;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[3].sqldata = (void *)((c_first));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[3].sqlind = (short int *)((SQLNULL));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[4].sqltype = (unsigned short)DT_STRING;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[4].sqllen = 3L;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[4].sqldata = (void *)((c_middle));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[4].sqlind = (short int *)((SQLNULL));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[5].sqltype = (unsigned short)DT_STRING;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[5].sqllen = 17L;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[5].sqldata = (void *)((c_last));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[5].sqlind = (short int *)((SQLNULL));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[6].sqltype = (unsigned short)DT_STRING;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[6].sqllen = 21L;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[6].sqldata = (void *)((c_street_1));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[6].sqlind = (short int *)((SQLNULL));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[7].sqltype = (unsigned short)DT_STRING;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[7].sqllen = 21L;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[7].sqldata = (void *)((c_street_2));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[7].sqlind = (short int *)((SQLNULL));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[8].sqltype = (unsigned short)DT_STRING;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[8].sqllen = 21L;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[8].sqldata = (void *)((c_city));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[8].sqlind = (short int *)((SQLNULL));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[9].sqltype = (unsigned short)DT_STRING;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[9].sqllen = 3L;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[9].sqldata = (void *)((c_state));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[9].sqlind = (short int *)((SQLNULL));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[10].sqltype = (unsigned short)DT_STRING;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[10].sqllen = 10L;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[10].sqldata = (void *)((c_zip));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[10].sqlind = (short int *)((SQLNULL));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[11].sqltype = (unsigned short)DT_STRING;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[11].sqllen = 17L;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[11].sqldata = (void *)((c_phone));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[11].sqlind = (short int *)((SQLNULL));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[12].sqltype = (unsigned short)DT_STRING;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[12].sqllen = 31L;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[12].sqldata = (void *)((timestamp));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[12].sqlind = (short int *)((SQLNULL));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[13].sqltype = (unsigned short)DT_STRING;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[13].sqllen = 3L;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[13].sqldata = (void *)((c_credit));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[13].sqlind = (short int *)((SQLNULL));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[14].sqltype = (unsigned short)DT_INT;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[14].sqldata = (void *)(&(c_credit_lim));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[14].sqlind = (short int *)((SQLNULL));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[15].sqltype = (unsigned short)DT_FLOAT;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[15].sqldata = (void *)(&(c_discount));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[15].sqlind = (short int *)((SQLNULL));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[16].sqltype = (unsigned short)DT_FLOAT;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[16].sqldata = (void *)(&(c_balance));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[16].sqlind = (short int *)((SQLNULL));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[17].sqltype = (unsigned short)DT_STRING;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[17].sqllen = 251L;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[17].sqldata = (void *)((c_data1));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[17].sqlind = (short int *)((SQLNULL));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[18].sqltype = (unsigned short)DT_STRING;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[18].sqllen = 251L;
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[18].sqldata = (void *)((c_data2));
((SQLDA *)&__SQLV_tpccld_11)->sqlvar[18].sqlind = (short int *)((SQLNULL));
dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_12, ((SQLDA *)&__SQLV_tpccld_11), (SQLDA *)0 );
```

```
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }

              } 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) ; */
            {
            struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[18];}
__SQLV_tpccld_13    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(18-1)*sizeof(SQLDA_VARIABLE),    18, 18 };
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[0].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[0].sqldata = (void *)(&(c_id));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[0].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[1].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[1].sqldata = (void *)(&(c_d_id));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[1].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[2].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[2].sqldata = (void *)(&(c_w_id));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[2].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[3].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[3].sqllen = 17L;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[3].sqldata = (void *)((c_first));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[3].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[4].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[4].sqllen = 3L;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[4].sqldata = (void *)((c_middle));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[4].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[5].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[5].sqllen = 17L;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[5].sqldata = (void *)((c_last));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[5].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[6].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[6].sqllen = 21L;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[6].sqldata = (void *)((c_street_1));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[6].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[7].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[7].sqllen = 21L;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[7].sqldata = (void *)((c_street_2));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[7].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[8].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[8].sqllen = 21L;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[8].sqldata = (void *)((c_city));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[8].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[9].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[9].sqllen = 3L;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[9].sqldata = (void *)((c_state));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[9].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[10].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[10].sqllen = 10L;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[10].sqldata = (void *)((c_zip));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[10].sqlind = (short int *)((SQLNULL));
```

```
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[11].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[11].sqllen = 17L;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[11].sqldata = (void *)((c_phone));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[11].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[12].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[12].sqllen = 31L;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[12].sqldata = (void *)((timestamp));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[12].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[13].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[13].sqllen = 3L;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[13].sqldata = (void *)((c_credit));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[13].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[14].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[14].sqldata = (void *)(&(c_credit_lim));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[14].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[15].sqltype = (unsigned short)DT_FLOAT;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[15].sqldata = (void *)(&(c_discount));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[15].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[16].sqltype = (unsigned short)DT_FLOAT;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[16].sqldata = (void *)(&(c_balance));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[16].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[17].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[17].sqllen = 501L;
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[17].sqldata = (void *)((c_data));
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[17].sqlind = (short int *)((SQLNULL));
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_14, ((SQLDA *)&__SQLV_tpccld_13), (SQLDA *)0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }

             }
    }

    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); */
            {
            struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[6];}
__SQLV_tpccld_15    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(6-1)*sizeof(SQLDA_VARIABLE),    6, 6 };
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[0].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[0].sqldata = (void *)(&(c_id));
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[0].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[1].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[1].sqldata = (void *)(&(c_d_id));
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[1].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[2].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[2].sqldata = (void *)(&(c_w_id));
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[2].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[3].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[3].sqllen = 31L;
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[3].sqldata = (void *)((timestamp));
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[3].sqlind = (short int *)((SQLNULL));
```

```
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[4].sqltype = (unsigned short)DT_FLOAT;
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[4].sqldata = (void *)(&(h_amount));
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[4].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[5].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[5].sqllen = 25L;
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[5].sqldata = (void *)((h_data));
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[5].sqlind = (short int *)((SQLNULL));
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_16, ((SQLDA *)&__SQLV_tpccld_15), (SQLDA *)0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }

      }

      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; */
            {
            dbpp_commit( (sqlcaptr), 0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }

               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); */
           {
           struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[6];}
__SQLV_tpccld_17    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(6-1)*sizeof(SQLDA_VARIABLE),    6, 6 };
           ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[0].sqltype = (unsigned short)DT_INT;
           ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[0].sqldata = (void *)(&(o_id));
           ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[0].sqlind = (short int *)((SQLNULL));
           ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[1].sqltype = (unsigned short)DT_INT;
           ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[1].sqldata = (void *)(&(o_d_id));
           ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[1].sqlind = (short int *)((SQLNULL));
           ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[2].sqltype = (unsigned short)DT_INT;
           ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[2].sqldata = (void *)(&(o_w_id));
           ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[2].sqlind = (short int *)((SQLNULL));
           ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[3].sqltype = (unsigned short)DT_INT;
           ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[3].sqldata = (void *)(&(o_c_id));
           ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[3].sqlind = (short int *)((SQLNULL));
           ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[4].sqltype = (unsigned short)DT_STRING;
           ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[4].sqllen = 31L;
           ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[4].sqldata = (void *)((timestamp));
           ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[4].sqlind = (short int *)((SQLNULL));
           ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[5].sqltype = (unsigned short)DT_INT;
           ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[5].sqldata = (void *)(&(o_ol_cnt));
```

```
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[5].sqlind = (short int *)((SQLNULL));
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_18, ((SQLDA *)&__SQLV_tpccld_17), (SQLDA *)0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }

       /* EXEC SQL INSERT INTO
 new_order (no_o_id, no_d_id, no_w_id)
 values (:o_id, :o_d_id, :o_w_id); */
            {
            struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[3];}
__SQLV_tpccld_19    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(3-1)*sizeof(SQLDA_VARIABLE),    3, 3 };
            ((SQLDA *)&__SQLV_tpccld_19)->sqlvar[0].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_19)->sqlvar[0].sqldata = (void *)(&(o_id));
            ((SQLDA *)&__SQLV_tpccld_19)->sqlvar[0].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_19)->sqlvar[1].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_19)->sqlvar[1].sqldata = (void *)(&(o_d_id));
            ((SQLDA *)&__SQLV_tpccld_19)->sqlvar[1].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_19)->sqlvar[2].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_19)->sqlvar[2].sqldata = (void *)(&(o_w_id));
            ((SQLDA *)&__SQLV_tpccld_19)->sqlvar[2].sqlind = (short int *)((SQLNULL));
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_20, ((SQLDA *)&__SQLV_tpccld_19), (SQLDA *)0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }

            }
      } 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); */
            {
            struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[7];}
__SQLV_tpccld_21    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(7-1)*sizeof(SQLDA_VARIABLE),    7, 7 };
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[0].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[0].sqldata = (void *)(&(o_id));
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[0].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[1].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[1].sqldata = (void *)(&(o_d_id));
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[1].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[2].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[2].sqldata = (void *)(&(o_w_id));
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[2].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[3].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[3].sqldata = (void *)(&(o_c_id));
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[3].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[4].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[4].sqllen = 31L;
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[4].sqldata = (void *)((timestamp));
```

```
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[4].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[5].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[5].sqldata = (void *)(&(o_carrier_id));
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[5].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[6].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[6].sqldata = (void *)(&(o_ol_cnt));
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[6].sqlind = (short int *)((SQLNULL));
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_22, ((SQLDA *)&__SQLV_tpccld_21), (SQLDA *)0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }

            }
      }


      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); */
            {
            struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[9];}
__SQLV_tpccld_23    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(9-1)*sizeof(SQLDA_VARIABLE),    9, 9 };
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[0].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[0].sqldata = (void *)(&(o_id));
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[0].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[1].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[1].sqldata = (void *)(&(o_d_id));
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[1].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[2].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[2].sqldata = (void *)(&(o_w_id));
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[2].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[3].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[3].sqldata = (void *)(&(ol));
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[3].sqlind = (short int *)((SQLNULL));
```

```
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[4].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[4].sqldata = (void *)(&(ol_i_id));
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[4].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[5].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[5].sqldata = (void *)(&(ol_supply_w_id));
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[5].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[6].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[6].sqldata = (void *)(&(ol_quantity));
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[6].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[7].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[7].sqldata = (void *)(&(ol_amount));
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[7].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[8].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[8].sqllen = 25L;
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[8].sqldata = (void *)((ol_dist_info));
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[8].sqlind = (short int *)((SQLNULL));
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_24, ((SQLDA *)&__SQLV_tpccld_23), (SQLDA *)0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }

              }
        } 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); */
            {
            struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[10];}
__SQLV_tpccld_25    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(10-1)*sizeof(SQLDA_VARIABLE),    10, 10 };
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[0].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[0].sqldata = (void *)(&(o_id));
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[0].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[1].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[1].sqldata = (void *)(&(o_d_id));
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[1].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[2].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[2].sqldata = (void *)(&(o_w_id));
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[2].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[3].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[3].sqldata = (void *)(&(ol));
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[3].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[4].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[4].sqldata = (void *)(&(ol_i_id));
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[4].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[5].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[5].sqldata = (void *)(&(ol_supply_w_id));
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[5].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[6].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[6].sqldata = (void *)(&(ol_quantity));
```

```
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[6].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[7].sqltype = (unsigned short)DT_INT;
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[7].sqldata = (void *)(&(ol_amount));
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[7].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[8].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[8].sqllen = 25L;
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[8].sqldata = (void *)((ol_dist_info));
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[8].sqlind = (short int *)((SQLNULL));
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[9].sqltype = (unsigned short)DT_STRING;
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[9].sqllen = 31L;
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[9].sqldata = (void *)((timestamp));
            ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[9].sqlind = (short int *)((SQLNULL));
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_26, ((SQLDA *)&__SQLV_tpccld_25), (SQLDA *)0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }

              }
            }

       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; */
            {
            dbpp_commit( (sqlcaptr), 0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }

                if ( !(o_id % 1000) ) printf(" %ld\r",o_id);
            }
     }
   }
   if( inserting ) {
            /* EXEC SQL COMMIT WORK; */
            {
            dbpp_commit( (sqlcaptr), 0 );
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }

   }

   // 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; */
            {
            dbpp_rollback( (sqlcaptr), 0 );
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
            }


    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';
    itoa(RandomNumber(0, 99999999), tmp, 10);
    memcpy(str, tmp, strlen(tmp));
    itoa(RandomNumber(0, 99999999), tmp, 10);
    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");
    itoa(RandomNumber(0, 9999), tmp, 10);
    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() / 2147483647.0);
}


//=============================================================================
// Function : RandomNumber
//
// Description:
//=============================================================================
long RandomNumber(long lower, long upper)
{
   long rand_num;
   if ( upper == lower )
      return lower;
   upper++;
   if ( upper <= lower )
      rand_num = upper;
   else
      rand_num = lower + genrand() % (upper - lower);
#ifdef DEBUG
   printf("[%ld]DBG: RandomNumber between %ld & %ld ==> %ld\n",
      (int) GetCurrentThreadId(), lower, upper,
      rand_num);
#endif
   return rand_num;
}


//=============================================================================
// Function : NURand
//
// Description:
//=============================================================================
long NURand(int iConst, long x, long y, long C)
{
   long rand_num;
#ifdef DEBUG
   printf("[%ld]DBG: Entering NURand()...\n", (int) GetCurrentThreadId());
#endif
   rand_num = (((RandomNumber(0,iConst) | RandomNumber(x,y)) + C) % (y-x+1))+x;
#ifdef DEBUG
   printf("[%ld]DBG: NURand: num = %d\n", (int) GetCurrentThreadId(), rand_num);
#endif
   return rand_num;
}




/*c:\original\kit\tpccld.cpp*/
/*************************************************/

#line 1 "tpccld.sqc"
#define _SQL_OS_WINNT
// ************************************************
// ********* GENERATED FILE - DO NOT EDIT! *********
// ************************************************

#define _SQL_SQLPP_VERSION_MAJOR 11
#define _SQL_SQLPP_VERSION_MINOR 0
#define _SQL_SQLPP_VERSION_MAINT 0
```

```
#define _SQL_SQLPP_DBLIB_VERSION 11
#line 1 "tpccld.sqc"
// ********************************************************************
// Copyright 2002-2008 iAnywhere Solutions, Inc.  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

#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; */
#line 43 "tpccld.sqc"
#include "sqlca.h"
#include "sqldef.h"
#ifdef __cplusplus
    extern "C" {
#endif
extern SQLCA sqlca;
extern SQLCA *sqlcaptr;
extern short int _ESQL_Version11_;
extern short int _ESQL_OS_WINNT_;
#ifdef __cplusplus
    }
#endif
#include "sqlda.h"
#line 43 "tpccld.sqc"
static char __SQLV_tpccld_1[] = "DBA";
static char __SQLV_tpccld_2[] = "sql";
static char __SQLV_tpccld_4[] = "INSERT INTO\n item (i_id, i_im_id, i_name, i_price, i_data)\n values (:i_id, :i_im_id, :i_name,
:i_price, :i_data)";
static char __SQLV_tpccld_6[] = "INSERT INTO\n warehouse (w_id, w_name,\n w_street_1, w_street_2, w_city, w_state, w_zip,\n
w_tax, w_ytd)\n values (:w_id, :w_name,\n :w_street_1, :w_street_2, :w_city, :w_state,\n :w_zip, :w_tax, :w_ytd)";
```

```
static char __SQLV_tpccld_8[] = "INSERT INTO\n stock (s_i_id, s_w_id, s_quantity,\n s_dist_01, s_dist_02, s_dist_03, s_dist_04,
s_dist_05,\n s_dist_06, s_dist_07, s_dist_08, s_dist_09, s_dist_10,\n s_ytd, s_order_cnt, s_remote_cnt, s_data)\n values (:s_i_id,
:s_w_id, :s_quantity,\n :s_dist_01, :s_dist_02, :s_dist_03, :s_dist_04, :s_dist_05,\n :s_dist_06, :s_dist_07, :s_dist_08, :s_dist_09,
:s_dist_10,\n 0, 0, 0, :s_data)";
static char __SQLV_tpccld_10[] = "INSERT INTO\n district (d_id, d_w_id, d_name,\n d_street_1, d_street_2, d_city, d_state,
d_zip,\n d_tax, d_ytd, d_next_o_id)\n values (:d_id, :d_w_id, :d_name,\n :d_street_1, :d_street_2, :d_city, :d_state, :d_zip,\n
:d_tax, :d_ytd, :d_next_o_id)";
static char __SQLV_tpccld_12[] = "INSERT INTO\n customer (c_id, c_d_id, c_w_id,\n c_first, c_middle, c_last,\n c_street_1,
c_street_2, c_city, c_state, c_zip,\n c_phone, c_since, c_credit,\n c_credit_lim, c_discount, c_balance, c_data1, c_data2,\n
c_ytd_payment, c_payment_cnt, c_delivery_cnt)\n values (:c_id, :c_d_id, :c_w_id,\n :c_first, :c_middle, :c_last,\n :c_street_1,
:c_street_2, :c_city, :c_state, :c_zip,\n :c_phone, :timestamp, :c_credit,\n :c_credit_lim, :c_discount, :c_balance, :c_data1,
:c_data2,\n 10.0, 1, 0) ";
static char __SQLV_tpccld_14[] = "INSERT INTO\n customer (c_id, c_d_id, c_w_id,\n c_first, c_middle, c_last,\n c_street_1,
c_street_2, c_city, c_state, c_zip,\n c_phone, c_since, c_credit,\n c_credit_lim, c_discount, c_balance, c_data,\n c_ytd_payment,
c_payment_cnt, c_delivery_cnt)\n values (:c_id, :c_d_id, :c_w_id,\n :c_first, :c_middle, :c_last,\n :c_street_1, :c_street_2, :c_city,
:c_state, :c_zip,\n :c_phone, :timestamp, :c_credit,\n :c_credit_lim, :c_discount, :c_balance, :c_data,\n 10.0, 1, 0) ";
static char __SQLV_tpccld_16[] = "INSERT INTO\n history (h_c_id, h_c_d_id, h_c_w_id,\n h_d_id, h_w_id, h_date, h_amount,
h_data)\n values (:c_id, :c_d_id, :c_w_id,\n :c_d_id, :c_w_id, :timestamp, :h_amount, :h_data)";
static char __SQLV_tpccld_18[] = "INSERT INTO\n orders (o_id, o_d_id, o_w_id, o_c_id,\n o_entry_d, o_carrier_id, o_ol_cnt,
o_all_local)\n values (:o_id, :o_d_id, :o_w_id, :o_c_id,\n :timestamp, NULL, :o_ol_cnt, 1)";
static char __SQLV_tpccld_20[] = "INSERT INTO\n new_order (no_o_id, no_d_id, no_w_id)\n values (:o_id, :o_d_id, :o_w_id)";
static char __SQLV_tpccld_22[] = "INSERT INTO\n orders (o_id, o_d_id, o_w_id, o_c_id,\n o_entry_d, o_carrier_id, o_ol_cnt,
o_all_local)\n values (:o_id, :o_d_id, :o_w_id, :o_c_id,\n :timestamp, :o_carrier_id, :o_ol_cnt, 1)";
static char __SQLV_tpccld_24[] = "INSERT INTO\n order_line (ol_o_id, ol_d_id, ol_w_id, ol_number,\n ol_i_id, ol_supply_w_id,
ol_quantity, ol_amount,\n ol_dist_info, ol_delivery_d)\n values (:o_id, :o_d_id, :o_w_id, :ol,\n :ol_i_id, :ol_supply_w_id,
:ol_quantity, :ol_amount,\n :ol_dist_info, NULL)";
static char __SQLV_tpccld_26[] = "INSERT INTO\n order_line (ol_o_id, ol_d_id, ol_w_id, ol_number,\n ol_i_id, ol_supply_w_id,
ol_quantity, ol_amount,\n ol_dist_info, ol_delivery_d)\n values (:o_id, :o_d_id, :o_w_id, :ol,\n :ol_i_id, :ol_supply_w_id,
:ol_quantity, :ol_amount,\n :ol_dist_info, :timestamp)";
#line 43 "tpccld.sqc"


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; */
#line 76 "tpccld.sqc"
#line 77 "tpccld.sqc"
 char timestamp[31];
```

```
 long count_ware;

 /* EXEC SQL END DECLARE SECTION; */
#line 79 "tpccld.sqc"
#line 79 "tpccld.sqc"

    _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; */
#line 110 "tpccld.sqc"
#line 110 "tpccld.sqc"


  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"; */
#line 186 "tpccld.sqc"
#line 185 "tpccld.sqc"
            {
#line 186 "tpccld.sqc"
            dbpp_connect_40( (sqlcaptr), __SQLV_tpccld_1, __SQLV_tpccld_2, SQLNULL, SQLNULL, SQLNULL );
#line 186 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto Error_SqlCall;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 186 "tpccld.sqc"
            }
#line 186 "tpccld.sqc"

   }
```

```
    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; */
#line 202 "tpccld.sqc"
#line 201 "tpccld.sqc"
            {
#line 202 "tpccld.sqc"
            dbpp_commit( (sqlcaptr), 0 );
#line 202 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto Error_SqlCall;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 202 "tpccld.sqc"
            }
#line 202 "tpccld.sqc"

            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; */
#line 282 "tpccld.sqc"
#line 283 "tpccld.sqc"
 long i_id;
 long i_im_id;
 char i_name[25];
 float i_price;
 char i_data[51];

  /* EXEC SQL END DECLARE SECTION; */
#line 288 "tpccld.sqc"
#line 288 "tpccld.sqc"

        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; */
```

```
#line 298 "tpccld.sqc"
#line 298 "tpccld.sqc"

    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); */
#line 342 "tpccld.sqc"
#line 341 "tpccld.sqc"
              {
#line 342 "tpccld.sqc"
              struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[5];}
__SQLV_tpccld_3    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(5-1)*sizeof(SQLDA_VARIABLE),    5, 5 };
#line 342 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[0].sqltype = (unsigned short)DT_INT;
#line 342 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[0].sqldata = (void *)(&(i_id));
#line 342 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[0].sqlind = (short int *)((SQLNULL));
#line 342 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[1].sqltype = (unsigned short)DT_INT;
#line 342 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[1].sqldata = (void *)(&(i_im_id));
#line 342 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[1].sqlind = (short int *)((SQLNULL));
#line 342 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[2].sqltype = (unsigned short)DT_STRING;
```

```
#line 342 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[2].sqllen = 25L;
#line 342 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[2].sqldata = (void *)((i_name));
#line 342 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[2].sqlind = (short int *)((SQLNULL));
#line 342 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[3].sqltype = (unsigned short)DT_FLOAT;
#line 342 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[3].sqldata = (void *)(&(i_price));
#line 342 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[3].sqlind = (short int *)((SQLNULL));
#line 342 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[4].sqltype = (unsigned short)DT_STRING;
#line 342 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[4].sqllen = 51L;
#line 342 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[4].sqldata = (void *)((i_data));
#line 342 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_3)->sqlvar[4].sqlind = (short int *)((SQLNULL));
#line 342 "tpccld.sqc"
            db_verify_version_11();
#line 342 "tpccld.sqc"
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_4, ((SQLDA *)&__SQLV_tpccld_3), (SQLDA *)0 );
#line 342 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 342 "tpccld.sqc"
            }
#line 342 "tpccld.sqc"

     }
     if ( !(i_id % 100) ) {
             if( inserting ) {
                printf(".");
                /* EXEC SQL COMMIT WORK; */
#line 347 "tpccld.sqc"
#line 346 "tpccld.sqc"
            {
#line 347 "tpccld.sqc"
            dbpp_commit( (sqlcaptr), 0 );
#line 347 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 347 "tpccld.sqc"
            }
#line 347 "tpccld.sqc"

             }
        if ( !(i_id % 5000) ) printf(" %ld\r",i_id);
     }
   }
   if( inserting ) {
            /* EXEC SQL COMMIT WORK; */
#line 353 "tpccld.sqc"
#line 352 "tpccld.sqc"
            {
```

```
#line 353 "tpccld.sqc"
            dbpp_commit( (sqlcaptr), 0 );
#line 353 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 353 "tpccld.sqc"
            }
#line 353 "tpccld.sqc"

    } 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; */
#line 373 "tpccld.sqc"
#line 374 "tpccld.sqc"
 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; */
#line 383 "tpccld.sqc"
#line 383 "tpccld.sqc"

            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; */
#line 392 "tpccld.sqc"
#line 392 "tpccld.sqc"

    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,%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); */
#line 421 "tpccld.sqc"
#line 420 "tpccld.sqc"
              {
#line 421 "tpccld.sqc"
              struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[9];}
__SQLV_tpccld_5     = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(9-1)*sizeof(SQLDA_VARIABLE),    9, 9 };
#line 421 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[0].sqltype = (unsigned short)DT_INT;
#line 421 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[0].sqldata = (void *)(&(w_id));
#line 421 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[0].sqlind = (short int *)((SQLNULL));
#line 421 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[1].sqltype = (unsigned short)DT_STRING;
#line 421 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[1].sqllen = 11L;
#line 421 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[1].sqldata = (void *)((w_name));
#line 421 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[1].sqlind = (short int *)((SQLNULL));
#line 421 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[2].sqltype = (unsigned short)DT_STRING;
#line 421 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[2].sqllen = 21L;
#line 421 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[2].sqldata = (void *)((w_street_1));
#line 421 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[2].sqlind = (short int *)((SQLNULL));
#line 421 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[3].sqltype = (unsigned short)DT_STRING;
#line 421 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[3].sqllen = 21L;
#line 421 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[3].sqldata = (void *)((w_street_2));
#line 421 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[3].sqlind = (short int *)((SQLNULL));
#line 421 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[4].sqltype = (unsigned short)DT_STRING;
#line 421 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[4].sqllen = 21L;
#line 421 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[4].sqldata = (void *)((w_city));
#line 421 "tpccld.sqc"
```

```
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[4].sqlind = (short int *)((SQLNULL));
#line 421 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[5].sqltype = (unsigned short)DT_STRING;
#line 421 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[5].sqllen = 3L;
#line 421 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[5].sqldata = (void *)((w_state));
#line 421 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[5].sqlind = (short int *)((SQLNULL));
#line 421 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[6].sqltype = (unsigned short)DT_STRING;
#line 421 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[6].sqllen = 10L;
#line 421 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[6].sqldata = (void *)((w_zip));
#line 421 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[6].sqlind = (short int *)((SQLNULL));
#line 421 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[7].sqltype = (unsigned short)DT_FLOAT;
#line 421 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[7].sqldata = (void *)(&(w_tax));
#line 421 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[7].sqlind = (short int *)((SQLNULL));
#line 421 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[8].sqltype = (unsigned short)DT_FLOAT;
#line 421 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[8].sqldata = (void *)(&(w_ytd));
#line 421 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_5)->sqlvar[8].sqlind = (short int *)((SQLNULL));
#line 421 "tpccld.sqc"
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_6, ((SQLDA *)&__SQLV_tpccld_5), (SQLDA *)0 );
#line 421 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 421 "tpccld.sqc"
            }
#line 421 "tpccld.sqc"

      }
      /** Make Rows associated with Warehouse **/
      Stock(w_id);
      District(w_id);
    }
    if( inserting ) {
            /* EXEC SQL COMMIT WORK; */
#line 428 "tpccld.sqc"
#line 427 "tpccld.sqc"
            {
#line 428 "tpccld.sqc"
            dbpp_commit( (sqlcaptr), 0 );
#line 428 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 428 "tpccld.sqc"
            }
#line 428 "tpccld.sqc"
```

```
    } 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; */
#line 447 "tpccld.sqc"

  /* EXEC SQL#line 448 "tpccld.sqc"
 END DECLARE SECTION; */
#line 448 "tpccld.sqc"
#line 448 "tpccld.sqc"

   long    w_id;
   long    d_id;
   FILE *   fpc;
   FILE *   fph;
   /* EXEC SQL WHENEVER SQLERROR GOTO sqlerr; */
#line 453 "tpccld.sqc"
#line 453 "tpccld.sqc"


   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; */
#line 466 "tpccld.sqc"
#line 465 "tpccld.sqc"
            {
#line 466 "tpccld.sqc"
            dbpp_commit( (sqlcaptr), 0 );
#line 466 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 466 "tpccld.sqc"
            }
#line 466 "tpccld.sqc"
  /* 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; */
#line 485 "tpccld.sqc"
#line 486 "tpccld.sqc"
 long w_id;
 long d_id;

  /* EXEC SQL END DECLARE SECTION; */
#line 488 "tpccld.sqc"
#line 488 "tpccld.sqc"

   FILE *   fpo;
   FILE *   fpno;
   FILE *   fpol;

   /* EXEC SQL WHENEVER SQLERROR GOTO sqlerr; */
#line 493 "tpccld.sqc"
#line 493 "tpccld.sqc"


   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; */
#line 507 "tpccld.sqc"
#line 506 "tpccld.sqc"
            {
#line 507 "tpccld.sqc"
            dbpp_commit( (sqlcaptr), 0 );
#line 507 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 507 "tpccld.sqc"
            }
#line 507 "tpccld.sqc"
  /* 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; */
#line 527 "tpccld.sqc"
#line 528 "tpccld.sqc"
 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; */
#line 542 "tpccld.sqc"
#line 542 "tpccld.sqc"

        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; */
#line 560 "tpccld.sqc"
#line 560 "tpccld.sqc"

    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); */
#line 625 "tpccld.sqc"
#line 624 "tpccld.sqc"
             {
#line 625 "tpccld.sqc"
             struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[14];}
__SQLV_tpccld_7    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(14-1)*sizeof(SQLDA_VARIABLE),    14, 14 };
#line 625 "tpccld.sqc"
             ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[0].sqltype = (unsigned short)DT_INT;
#line 625 "tpccld.sqc"
             ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[0].sqldata = (void *)(&(s_i_id));
```

```
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[0].sqlind = (short int *)((SQLNULL));
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[1].sqltype = (unsigned short)DT_INT;
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[1].sqldata = (void *)(&(s_w_id));
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[1].sqlind = (short int *)((SQLNULL));
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[2].sqltype = (unsigned short)DT_INT;
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[2].sqldata = (void *)(&(s_quantity));
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[2].sqlind = (short int *)((SQLNULL));
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[3].sqltype = (unsigned short)DT_STRING;
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[3].sqllen = 25L;
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[3].sqldata = (void *)((s_dist_01));
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[3].sqlind = (short int *)((SQLNULL));
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[4].sqltype = (unsigned short)DT_STRING;
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[4].sqllen = 25L;
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[4].sqldata = (void *)((s_dist_02));
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[4].sqlind = (short int *)((SQLNULL));
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[5].sqltype = (unsigned short)DT_STRING;
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[5].sqllen = 25L;
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[5].sqldata = (void *)((s_dist_03));
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[5].sqlind = (short int *)((SQLNULL));
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[6].sqltype = (unsigned short)DT_STRING;
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[6].sqllen = 25L;
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[6].sqldata = (void *)((s_dist_04));
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[6].sqlind = (short int *)((SQLNULL));
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[7].sqltype = (unsigned short)DT_STRING;
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[7].sqllen = 25L;
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[7].sqldata = (void *)((s_dist_05));
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[7].sqlind = (short int *)((SQLNULL));
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[8].sqltype = (unsigned short)DT_STRING;
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[8].sqllen = 25L;
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[8].sqldata = (void *)((s_dist_06));
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[8].sqlind = (short int *)((SQLNULL));
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[9].sqltype = (unsigned short)DT_STRING;
#line 625 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[9].sqllen = 25L;
#line 625 "tpccld.sqc"
```

```
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[9].sqldata = (void *)((s_dist_07));
#line 625 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[9].sqlind = (short int *)((SQLNULL));
#line 625 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[10].sqltype = (unsigned short)DT_STRING;
#line 625 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[10].sqllen = 25L;
#line 625 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[10].sqldata = (void *)((s_dist_08));
#line 625 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[10].sqlind = (short int *)((SQLNULL));
#line 625 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[11].sqltype = (unsigned short)DT_STRING;
#line 625 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[11].sqllen = 25L;
#line 625 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[11].sqldata = (void *)((s_dist_09));
#line 625 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[11].sqlind = (short int *)((SQLNULL));
#line 625 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[12].sqltype = (unsigned short)DT_STRING;
#line 625 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[12].sqllen = 25L;
#line 625 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[12].sqldata = (void *)((s_dist_10));
#line 625 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[12].sqlind = (short int *)((SQLNULL));
#line 625 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[13].sqltype = (unsigned short)DT_STRING;
#line 625 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[13].sqllen = 51L;
#line 625 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[13].sqldata = (void *)((s_data));
#line 625 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_7)->sqlvar[13].sqlind = (short int *)((SQLNULL));
#line 625 "tpccld.sqc"
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_8, ((SQLDA *)&__SQLV_tpccld_7), (SQLDA *)0 );
#line 625 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 625 "tpccld.sqc"
            }
#line 625 "tpccld.sqc"

     }
     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; */
#line 634 "tpccld.sqc"
#line 633 "tpccld.sqc"
            {
#line 634 "tpccld.sqc"
            dbpp_commit( (sqlcaptr), 0 );
#line 634 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
```

```
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 634 "tpccld.sqc"
            }
#line 634 "tpccld.sqc"

             }
      }
    }
    if( inserting ) {
            /* EXEC SQL COMMIT WORK; */
#line 639 "tpccld.sqc"
#line 638 "tpccld.sqc"
            {
#line 639 "tpccld.sqc"
            dbpp_commit( (sqlcaptr), 0 );
#line 639 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 639 "tpccld.sqc"
            }
#line 639 "tpccld.sqc"
  /* 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; */
#line 658 "tpccld.sqc"
#line 659 "tpccld.sqc"
 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; */
#line 670 "tpccld.sqc"
#line 670 "tpccld.sqc"
```

```
            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; */
#line 679 "tpccld.sqc"
#line 679 "tpccld.sqc"


    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); */
#line 714 "tpccld.sqc"
#line 713 "tpccld.sqc"
            {
#line 714 "tpccld.sqc"
            struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[11];}
__SQLV_tpccld_9    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(11-1)*sizeof(SQLDA_VARIABLE),    11, 11 };
#line 714 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[0].sqltype = (unsigned short)DT_INT;
#line 714 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[0].sqldata = (void *)(&(d_id));
#line 714 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[0].sqlind = (short int *)((SQLNULL));
#line 714 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[1].sqltype = (unsigned short)DT_INT;
#line 714 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[1].sqldata = (void *)(&(d_w_id));
#line 714 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[1].sqlind = (short int *)((SQLNULL));
#line 714 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[2].sqltype = (unsigned short)DT_STRING;
```

```
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[2].sqllen = 11L;
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[2].sqldata = (void *)((d_name));
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[2].sqlind = (short int *)((SQLNULL));
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[3].sqltype = (unsigned short)DT_STRING;
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[3].sqllen = 21L;
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[3].sqldata = (void *)((d_street_1));
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[3].sqlind = (short int *)((SQLNULL));
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[4].sqltype = (unsigned short)DT_STRING;
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[4].sqllen = 21L;
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[4].sqldata = (void *)((d_street_2));
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[4].sqlind = (short int *)((SQLNULL));
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[5].sqltype = (unsigned short)DT_STRING;
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[5].sqllen = 21L;
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[5].sqldata = (void *)((d_city));
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[5].sqlind = (short int *)((SQLNULL));
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[6].sqltype = (unsigned short)DT_STRING;
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[6].sqllen = 3L;
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[6].sqldata = (void *)((d_state));
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[6].sqlind = (short int *)((SQLNULL));
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[7].sqltype = (unsigned short)DT_STRING;
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[7].sqllen = 10L;
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[7].sqldata = (void *)((d_zip));
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[7].sqlind = (short int *)((SQLNULL));
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[8].sqltype = (unsigned short)DT_FLOAT;
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[8].sqldata = (void *)(&(d_tax));
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[8].sqlind = (short int *)((SQLNULL));
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[9].sqltype = (unsigned short)DT_FLOAT;
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[9].sqldata = (void *)(&(d_ytd));
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[9].sqlind = (short int *)((SQLNULL));
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[10].sqltype = (unsigned short)DT_INT;
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[10].sqldata = (void *)(&(d_next_o_id));
#line 714 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_9)->sqlvar[10].sqlind = (short int *)((SQLNULL));
#line 714 "tpccld.sqc"
        dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_10, ((SQLDA *)&__SQLV_tpccld_9), (SQLDA *)0 );
#line 714 "tpccld.sqc"
```

```
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 714 "tpccld.sqc"
            }
#line 714 "tpccld.sqc"

     }

   }
   if( inserting ) {
            /* EXEC SQL COMMIT WORK; */
#line 719 "tpccld.sqc"
#line 718 "tpccld.sqc"
            {
#line 719 "tpccld.sqc"
            dbpp_commit( (sqlcaptr), 0 );
#line 719 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 719 "tpccld.sqc"
            }
#line 719 "tpccld.sqc"

   }

   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; */
#line 744 "tpccld.sqc"
#line 745 "tpccld.sqc"
 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; */
#line 766 "tpccld.sqc"
#line 766 "tpccld.sqc"

        _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; */
#line 786 "tpccld.sqc"
#line 786 "tpccld.sqc"


    // 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) ; */
#line 858 "tpccld.sqc"
#line 857 "tpccld.sqc"
             {
#line 858 "tpccld.sqc"
             struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[19];}
__SQLV_tpccld_11    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(19-1)*sizeof(SQLDA_VARIABLE),    19, 19 };
#line 858 "tpccld.sqc"
             ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[0].sqltype = (unsigned short)DT_INT;
#line 858 "tpccld.sqc"
             ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[0].sqldata = (void *)(&(c_id));
#line 858 "tpccld.sqc"
             ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[0].sqlind = (short int *)((SQLNULL));
#line 858 "tpccld.sqc"
             ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[1].sqltype = (unsigned short)DT_INT;
#line 858 "tpccld.sqc"
             ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[1].sqldata = (void *)(&(c_d_id));
#line 858 "tpccld.sqc"
             ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[1].sqlind = (short int *)((SQLNULL));
#line 858 "tpccld.sqc"
             ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[2].sqltype = (unsigned short)DT_INT;
```

```
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[2].sqldata = (void *)(&(c_w_id));
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[2].sqlind = (short int *)((SQLNULL));
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[3].sqltype = (unsigned short)DT_STRING;
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[3].sqllen = 17L;
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[3].sqldata = (void *)((c_first));
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[3].sqlind = (short int *)((SQLNULL));
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[4].sqltype = (unsigned short)DT_STRING;
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[4].sqllen = 3L;
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[4].sqldata = (void *)((c_middle));
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[4].sqlind = (short int *)((SQLNULL));
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[5].sqltype = (unsigned short)DT_STRING;
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[5].sqllen = 17L;
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[5].sqldata = (void *)((c_last));
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[5].sqlind = (short int *)((SQLNULL));
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[6].sqltype = (unsigned short)DT_STRING;
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[6].sqllen = 21L;
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[6].sqldata = (void *)((c_street_1));
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[6].sqlind = (short int *)((SQLNULL));
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[7].sqltype = (unsigned short)DT_STRING;
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[7].sqllen = 21L;
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[7].sqldata = (void *)((c_street_2));
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[7].sqlind = (short int *)((SQLNULL));
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[8].sqltype = (unsigned short)DT_STRING;
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[8].sqllen = 21L;
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[8].sqldata = (void *)((c_city));
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[8].sqlind = (short int *)((SQLNULL));
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[9].sqltype = (unsigned short)DT_STRING;
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[9].sqllen = 3L;
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[9].sqldata = (void *)((c_state));
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[9].sqlind = (short int *)((SQLNULL));
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[10].sqltype = (unsigned short)DT_STRING;
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[10].sqllen = 10L;
#line 858 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[10].sqldata = (void *)((c_zip));
#line 858 "tpccld.sqc"
```

```
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[10].sqlind = (short int *)((SQLNULL));
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[11].sqltype = (unsigned short)DT_STRING;
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[11].sqllen = 17L;
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[11].sqldata = (void *)((c_phone));
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[11].sqlind = (short int *)((SQLNULL));
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[12].sqltype = (unsigned short)DT_STRING;
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[12].sqllen = 31L;
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[12].sqldata = (void *)((timestamp));
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[12].sqlind = (short int *)((SQLNULL));
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[13].sqltype = (unsigned short)DT_STRING;
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[13].sqllen = 3L;
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[13].sqldata = (void *)((c_credit));
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[13].sqlind = (short int *)((SQLNULL));
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[14].sqltype = (unsigned short)DT_INT;
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[14].sqldata = (void *)(&(c_credit_lim));
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[14].sqlind = (short int *)((SQLNULL));
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[15].sqltype = (unsigned short)DT_FLOAT;
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[15].sqldata = (void *)(&(c_discount));
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[15].sqlind = (short int *)((SQLNULL));
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[16].sqltype = (unsigned short)DT_FLOAT;
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[16].sqldata = (void *)(&(c_balance));
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[16].sqlind = (short int *)((SQLNULL));
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[17].sqltype = (unsigned short)DT_STRING;
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[17].sqllen = 251L;
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[17].sqldata = (void *)((c_data1));
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[17].sqlind = (short int *)((SQLNULL));
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[18].sqltype = (unsigned short)DT_STRING;
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[18].sqllen = 251L;
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[18].sqldata = (void *)((c_data2));
#line 858 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_11)->sqlvar[18].sqlind = (short int *)((SQLNULL));
#line 858 "tpccld.sqc"
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_12, ((SQLDA *)&__SQLV_tpccld_11), (SQLDA *)0 );
#line 858 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
```

```
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 858 "tpccld.sqc"
            }
#line 858 "tpccld.sqc"

              } 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) ; */
#line 872 "tpccld.sqc"
#line 871 "tpccld.sqc"
            {
#line 872 "tpccld.sqc"
            struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[18];}
__SQLV_tpccld_13    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(18-1)*sizeof(SQLDA_VARIABLE),    18, 18 };
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[0].sqltype = (unsigned short)DT_INT;
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[0].sqldata = (void *)(&(c_id));
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[0].sqlind = (short int *)((SQLNULL));
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[1].sqltype = (unsigned short)DT_INT;
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[1].sqldata = (void *)(&(c_d_id));
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[1].sqlind = (short int *)((SQLNULL));
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[2].sqltype = (unsigned short)DT_INT;
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[2].sqldata = (void *)(&(c_w_id));
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[2].sqlind = (short int *)((SQLNULL));
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[3].sqltype = (unsigned short)DT_STRING;
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[3].sqllen = 17L;
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[3].sqldata = (void *)((c_first));
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[3].sqlind = (short int *)((SQLNULL));
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[4].sqltype = (unsigned short)DT_STRING;
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[4].sqllen = 3L;
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[4].sqldata = (void *)((c_middle));
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[4].sqlind = (short int *)((SQLNULL));
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[5].sqltype = (unsigned short)DT_STRING;
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[5].sqllen = 17L;
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[5].sqldata = (void *)((c_last));
#line 872 "tpccld.sqc"
```

```
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[5].sqlind = (short int *)((SQLNULL));
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[6].sqltype = (unsigned short)DT_STRING;
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[6].sqllen = 21L;
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[6].sqldata = (void *)((c_street_1));
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[6].sqlind = (short int *)((SQLNULL));
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[7].sqltype = (unsigned short)DT_STRING;
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[7].sqllen = 21L;
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[7].sqldata = (void *)((c_street_2));
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[7].sqlind = (short int *)((SQLNULL));
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[8].sqltype = (unsigned short)DT_STRING;
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[8].sqllen = 21L;
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[8].sqldata = (void *)((c_city));
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[8].sqlind = (short int *)((SQLNULL));
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[9].sqltype = (unsigned short)DT_STRING;
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[9].sqllen = 3L;
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[9].sqldata = (void *)((c_state));
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[9].sqlind = (short int *)((SQLNULL));
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[10].sqltype = (unsigned short)DT_STRING;
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[10].sqllen = 10L;
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[10].sqldata = (void *)((c_zip));
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[10].sqlind = (short int *)((SQLNULL));
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[11].sqltype = (unsigned short)DT_STRING;
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[11].sqllen = 17L;
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[11].sqldata = (void *)((c_phone));
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[11].sqlind = (short int *)((SQLNULL));
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[12].sqltype = (unsigned short)DT_STRING;
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[12].sqllen = 31L;
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[12].sqldata = (void *)((timestamp));
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[12].sqlind = (short int *)((SQLNULL));
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[13].sqltype = (unsigned short)DT_STRING;
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[13].sqllen = 3L;
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[13].sqldata = (void *)((c_credit));
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[13].sqlind = (short int *)((SQLNULL));
#line 872 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[14].sqltype = (unsigned short)DT_INT;
```

```
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[14].sqldata = (void *)(&(c_credit_lim));
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[14].sqlind = (short int *)((SQLNULL));
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[15].sqltype = (unsigned short)DT_FLOAT;
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[15].sqldata = (void *)(&(c_discount));
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[15].sqlind = (short int *)((SQLNULL));
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[16].sqltype = (unsigned short)DT_FLOAT;
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[16].sqldata = (void *)(&(c_balance));
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[16].sqlind = (short int *)((SQLNULL));
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[17].sqltype = (unsigned short)DT_STRING;
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[17].sqllen = 501L;
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[17].sqldata = (void *)((c_data));
#line 872 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_13)->sqlvar[17].sqlind = (short int *)((SQLNULL));
#line 872 "tpccld.sqc"
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_14, ((SQLDA *)&__SQLV_tpccld_13), (SQLDA *)0 );
#line 872 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 872 "tpccld.sqc"
            }
#line 872 "tpccld.sqc"

             }
    }

    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); */
#line 888 "tpccld.sqc"
#line 887 "tpccld.sqc"
            {
#line 888 "tpccld.sqc"
            struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[6];}
__SQLV_tpccld_15    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(6-1)*sizeof(SQLDA_VARIABLE),    6, 6 };
#line 888 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[0].sqltype = (unsigned short)DT_INT;
#line 888 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[0].sqldata = (void *)(&(c_id));
#line 888 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[0].sqlind = (short int *)((SQLNULL));
#line 888 "tpccld.sqc"
```

```
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[1].sqltype = (unsigned short)DT_INT;
#line 888 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[1].sqldata = (void *)(&(c_d_id));
#line 888 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[1].sqlind = (short int *)((SQLNULL));
#line 888 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[2].sqltype = (unsigned short)DT_INT;
#line 888 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[2].sqldata = (void *)(&(c_w_id));
#line 888 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[2].sqlind = (short int *)((SQLNULL));
#line 888 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[3].sqltype = (unsigned short)DT_STRING;
#line 888 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[3].sqllen = 31L;
#line 888 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[3].sqldata = (void *)((timestamp));
#line 888 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[3].sqlind = (short int *)((SQLNULL));
#line 888 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[4].sqltype = (unsigned short)DT_FLOAT;
#line 888 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[4].sqldata = (void *)(&(h_amount));
#line 888 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[4].sqlind = (short int *)((SQLNULL));
#line 888 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[5].sqltype = (unsigned short)DT_STRING;
#line 888 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[5].sqllen = 25L;
#line 888 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[5].sqldata = (void *)((h_data));
#line 888 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_15)->sqlvar[5].sqlind = (short int *)((SQLNULL));
#line 888 "tpccld.sqc"
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_16, ((SQLDA *)&__SQLV_tpccld_15), (SQLDA *)0 );
#line 888 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 888 "tpccld.sqc"
            }
#line 888 "tpccld.sqc"

    }

    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; */
#line 896 "tpccld.sqc"
#line 895 "tpccld.sqc"
            {
#line 896 "tpccld.sqc"
            dbpp_commit( (sqlcaptr), 0 );
#line 896 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
```

```
#endif
#line 896 "tpccld.sqc"
            }
#line 896 "tpccld.sqc"

                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; */
#line 926 "tpccld.sqc"
#line 927 "tpccld.sqc"
 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; */
#line 939 "tpccld.sqc"
#line 939 "tpccld.sqc"

        _def_null_term(ol_dist_info);

    /* EXEC SQL WHENEVER SQLERROR GOTO sqlerr; */
#line 942 "tpccld.sqc"
#line 942 "tpccld.sqc"


    // 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); */
#line 970 "tpccld.sqc"
#line 969 "tpccld.sqc"
            {
#line 970 "tpccld.sqc"
            struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[6];}
__SQLV_tpccld_17    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(6-1)*sizeof(SQLDA_VARIABLE),    6, 6 };
#line 970 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[0].sqltype = (unsigned short)DT_INT;
#line 970 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[0].sqldata = (void *)(&(o_id));
#line 970 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[0].sqlind = (short int *)((SQLNULL));
#line 970 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[1].sqltype = (unsigned short)DT_INT;
#line 970 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[1].sqldata = (void *)(&(o_d_id));
#line 970 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[1].sqlind = (short int *)((SQLNULL));
#line 970 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[2].sqltype = (unsigned short)DT_INT;
#line 970 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[2].sqldata = (void *)(&(o_w_id));
#line 970 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[2].sqlind = (short int *)((SQLNULL));
#line 970 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[3].sqltype = (unsigned short)DT_INT;
#line 970 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[3].sqldata = (void *)(&(o_c_id));
#line 970 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[3].sqlind = (short int *)((SQLNULL));
#line 970 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[4].sqltype = (unsigned short)DT_STRING;
#line 970 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[4].sqllen = 31L;
#line 970 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[4].sqldata = (void *)((timestamp));
#line 970 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[4].sqlind = (short int *)((SQLNULL));
#line 970 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[5].sqltype = (unsigned short)DT_INT;
#line 970 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[5].sqldata = (void *)(&(o_ol_cnt));
#line 970 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_17)->sqlvar[5].sqlind = (short int *)((SQLNULL));
#line 970 "tpccld.sqc"
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_18, ((SQLDA *)&__SQLV_tpccld_17), (SQLDA *)0 );
```

```
#line 970 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 970 "tpccld.sqc"
            }
#line 970 "tpccld.sqc"

       /* EXEC SQL INSERT INTO
 new_order (no_o_id, no_d_id, no_w_id)
 values (:o_id, :o_d_id, :o_w_id); */
#line 973 "tpccld.sqc"
#line 972 "tpccld.sqc"
            {
#line 973 "tpccld.sqc"
            struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[3];}
__SQLV_tpccld_19    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(3-1)*sizeof(SQLDA_VARIABLE),    3, 3 };
#line 973 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_19)->sqlvar[0].sqltype = (unsigned short)DT_INT;
#line 973 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_19)->sqlvar[0].sqldata = (void *)(&(o_id));
#line 973 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_19)->sqlvar[0].sqlind = (short int *)((SQLNULL));
#line 973 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_19)->sqlvar[1].sqltype = (unsigned short)DT_INT;
#line 973 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_19)->sqlvar[1].sqldata = (void *)(&(o_d_id));
#line 973 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_19)->sqlvar[1].sqlind = (short int *)((SQLNULL));
#line 973 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_19)->sqlvar[2].sqltype = (unsigned short)DT_INT;
#line 973 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_19)->sqlvar[2].sqldata = (void *)(&(o_w_id));
#line 973 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_19)->sqlvar[2].sqlind = (short int *)((SQLNULL));
#line 973 "tpccld.sqc"
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_20, ((SQLDA *)&__SQLV_tpccld_19), (SQLDA *)0 );
#line 973 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 973 "tpccld.sqc"
            }
#line 973 "tpccld.sqc"

            }
      } 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); */
#line 986 "tpccld.sqc"
```

```
#line 985 "tpccld.sqc"
            {
#line 986 "tpccld.sqc"
            struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[7];}
__SQLV_tpccld_21    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(7-1)*sizeof(SQLDA_VARIABLE),    7, 7 };
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[0].sqltype = (unsigned short)DT_INT;
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[0].sqldata = (void *)(&(o_id));
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[0].sqlind = (short int *)((SQLNULL));
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[1].sqltype = (unsigned short)DT_INT;
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[1].sqldata = (void *)(&(o_d_id));
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[1].sqlind = (short int *)((SQLNULL));
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[2].sqltype = (unsigned short)DT_INT;
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[2].sqldata = (void *)(&(o_w_id));
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[2].sqlind = (short int *)((SQLNULL));
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[3].sqltype = (unsigned short)DT_INT;
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[3].sqldata = (void *)(&(o_c_id));
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[3].sqlind = (short int *)((SQLNULL));
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[4].sqltype = (unsigned short)DT_STRING;
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[4].sqllen = 31L;
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[4].sqldata = (void *)((timestamp));
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[4].sqlind = (short int *)((SQLNULL));
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[5].sqltype = (unsigned short)DT_INT;
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[5].sqldata = (void *)(&(o_carrier_id));
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[5].sqlind = (short int *)((SQLNULL));
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[6].sqltype = (unsigned short)DT_INT;
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[6].sqldata = (void *)(&(o_ol_cnt));
#line 986 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_21)->sqlvar[6].sqlind = (short int *)((SQLNULL));
#line 986 "tpccld.sqc"
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_22, ((SQLDA *)&__SQLV_tpccld_21), (SQLDA *)0 );
#line 986 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 986 "tpccld.sqc"
            }
#line 986 "tpccld.sqc"

            }
    }
```

```
    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); */
#line 1021 "tpccld.sqc"
#line 1020 "tpccld.sqc"
            {
#line 1021 "tpccld.sqc"
            struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[9];}
__SQLV_tpccld_23    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(9-1)*sizeof(SQLDA_VARIABLE),    9, 9 };
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[0].sqltype = (unsigned short)DT_INT;
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[0].sqldata = (void *)(&(o_id));
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[0].sqlind = (short int *)((SQLNULL));
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[1].sqltype = (unsigned short)DT_INT;
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[1].sqldata = (void *)(&(o_d_id));
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[1].sqlind = (short int *)((SQLNULL));
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[2].sqltype = (unsigned short)DT_INT;
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[2].sqldata = (void *)(&(o_w_id));
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[2].sqlind = (short int *)((SQLNULL));
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[3].sqltype = (unsigned short)DT_INT;
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[3].sqldata = (void *)(&(ol));
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[3].sqlind = (short int *)((SQLNULL));
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[4].sqltype = (unsigned short)DT_INT;
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[4].sqldata = (void *)(&(ol_i_id));
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[4].sqlind = (short int *)((SQLNULL));
```

```
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[5].sqltype = (unsigned short)DT_INT;
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[5].sqldata = (void *)(&(ol_supply_w_id));
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[5].sqlind = (short int *)((SQLNULL));
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[6].sqltype = (unsigned short)DT_INT;
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[6].sqldata = (void *)(&(ol_quantity));
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[6].sqlind = (short int *)((SQLNULL));
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[7].sqltype = (unsigned short)DT_INT;
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[7].sqldata = (void *)(&(ol_amount));
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[7].sqlind = (short int *)((SQLNULL));
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[8].sqltype = (unsigned short)DT_STRING;
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[8].sqllen = 25L;
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[8].sqldata = (void *)((ol_dist_info));
#line 1021 "tpccld.sqc"
            ((SQLDA *)&__SQLV_tpccld_23)->sqlvar[8].sqlind = (short int *)((SQLNULL));
#line 1021 "tpccld.sqc"
            dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_24, ((SQLDA *)&__SQLV_tpccld_23), (SQLDA *)0 );
#line 1021 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 1021 "tpccld.sqc"
            }
#line 1021 "tpccld.sqc"

              }
        } 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); */
#line 1039 "tpccld.sqc"
#line 1038 "tpccld.sqc"
              {
#line 1039 "tpccld.sqc"
              struct { unsigned char sqldaid[8]; a_sql_int32   sqldabc; short int sqln; short int sqld; SQLDA_VARIABLE sqlvar[10];}
__SQLV_tpccld_25    = { {'S','Q','L','D','A',' ',' ',' '},    sizeof(SQLDA)+(10-1)*sizeof(SQLDA_VARIABLE),    10, 10 };
#line 1039 "tpccld.sqc"
              ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[0].sqltype = (unsigned short)DT_INT;
#line 1039 "tpccld.sqc"
```

```
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[0].sqldata = (void *)(&(o_id));
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[0].sqlind = (short int *)((SQLNULL));
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[1].sqltype = (unsigned short)DT_INT;
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[1].sqldata = (void *)(&(o_d_id));
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[1].sqlind = (short int *)((SQLNULL));
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[2].sqltype = (unsigned short)DT_INT;
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[2].sqldata = (void *)(&(o_w_id));
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[2].sqlind = (short int *)((SQLNULL));
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[3].sqltype = (unsigned short)DT_INT;
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[3].sqldata = (void *)(&(ol));
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[3].sqlind = (short int *)((SQLNULL));
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[4].sqltype = (unsigned short)DT_INT;
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[4].sqldata = (void *)(&(ol_i_id));
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[4].sqlind = (short int *)((SQLNULL));
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[5].sqltype = (unsigned short)DT_INT;
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[5].sqldata = (void *)(&(ol_supply_w_id));
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[5].sqlind = (short int *)((SQLNULL));
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[6].sqltype = (unsigned short)DT_INT;
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[6].sqldata = (void *)(&(ol_quantity));
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[6].sqlind = (short int *)((SQLNULL));
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[7].sqltype = (unsigned short)DT_INT;
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[7].sqldata = (void *)(&(ol_amount));
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[7].sqlind = (short int *)((SQLNULL));
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[8].sqltype = (unsigned short)DT_STRING;
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[8].sqllen = 25L;
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[8].sqldata = (void *)((ol_dist_info));
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[8].sqlind = (short int *)((SQLNULL));
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[9].sqltype = (unsigned short)DT_STRING;
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[9].sqllen = 31L;
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[9].sqldata = (void *)((timestamp));
#line 1039 "tpccld.sqc"
        ((SQLDA *)&__SQLV_tpccld_25)->sqlvar[9].sqlind = (short int *)((SQLNULL));
#line 1039 "tpccld.sqc"
        dbpp_prepare_exec_drop( (sqlcaptr), __SQLV_tpccld_26, ((SQLDA *)&__SQLV_tpccld_25), (SQLDA *)0 );
#line 1039 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
```

```
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 1039 "tpccld.sqc"
            }
#line 1039 "tpccld.sqc"

              }
            }

       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; */
#line 1051 "tpccld.sqc"
#line 1050 "tpccld.sqc"
            {
#line 1051 "tpccld.sqc"
             dbpp_commit( (sqlcaptr), 0 );
#line 1051 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 1051 "tpccld.sqc"
            }
#line 1051 "tpccld.sqc"

                if ( !(o_id % 1000) ) printf(" %ld\r",o_id);
            }
     }
   }
   if( inserting ) {
            /* EXEC SQL COMMIT WORK; */
#line 1057 "tpccld.sqc"
#line 1056 "tpccld.sqc"
            {
#line 1057 "tpccld.sqc"
             dbpp_commit( (sqlcaptr), 0 );
#line 1057 "tpccld.sqc"
if( (sqlcaptr)->sqlcode < 0 ) goto sqlerr;
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 1057 "tpccld.sqc"
            }
#line 1057 "tpccld.sqc"

   }

   // 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; */
#line 1098 "tpccld.sqc"
#line 1098 "tpccld.sqc"

    /* EXEC SQL ROLLBACK WORK; */
#line 1099 "tpccld.sqc"
#line 1098 "tpccld.sqc"
             {
#line 1099 "tpccld.sqc"
              dbpp_rollback( (sqlcaptr), 0 );
#ifdef __SQLCODE
SQLCODE = __SQLCODE;
#endif
#ifdef __SQLSTATE
strncpy(SQLSTATE,__SQLSTATE,6);
#endif
#line 1099 "tpccld.sqc"
             }
#line 1099 "tpccld.sqc"


    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 2002 iAnywhere Solutions, Inc.  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 ~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 2002-2008 iAnywhere Solutions, Inc.  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 0-255

#define NU_C_C_ID    329                 // must be in range 0-1023
#define NU_C_OL_ID               7413   // must be in range 0-8191

// 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 ) * 2.3283064365386963e-10 );
}

// 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(*) 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 )''' %(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,
                                            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 )
```