MICROSOFT® SQL SERVER™ TPC-E BENCHMARK KIT DATABASE SETUP REFERENCE

Version 1.5.0-1007

Jamie Reding (jamiere@microsoft.com)
SQL Server Performance Team
Microsoft Corporation

CONTENTS

INTRODUCTION

This document provides a reference for all the steps required to build a TPC-E database using the Microsoft TPC-E Benchmark Kit. It includes the names of the scripts used and the order of execution of these scripts.

TPC-E BENCHMARK DATABASE SETUP

1. Remove any existing TPC-E database.

    a. This step ensures that any previous TPC-E database is removed and that SQL Server is ready to begin the creation of a new TPC-E database.

    b. Script Name: Remove_Database.sql

2. Create the database.

    a. The Create Database script defines the file groups and the files within the file groups that SQL Server will use to store the TPC-E data.

    b. Script Name: Create_Database.sql

3. Create the TPC-E Data Type synonyms.

    a. The TPC-E synonyms map the native data types in SQL Server to the data types defined in the TPC-E specification.

    b. Example: bigint (SQL Server native data type) ➔ IDENT_T (TPC-E data type)

    c. Script Name: Create_TPCE_Types.sql

4. Create the TPC-E scaling tables and appropriate check constraints.

    a. This step creates the scaling TPC-E tables as defined in Clause 2.6.1.10 in the TPC-E specification.

    b. The script also creates the check constraints on the scaling tables as required by the TPC-E specification in Clauses 2.2.4, 2.2.5., 2.2.6, and 2.2.7.

    c. Script Name: Create_Tables_Scaling.sql

5. Create the TPC-E growing tables.

    a. This step creates the growing TPC-E tables as defined in Clause 2.6.1.11 in the TPC-E specification.

    b. The script also creates the check constraints on the growing tables as required by the TPC-E specification in Clauses 2.2.4, 2.2.5., 2.2.6, and 2.2.7.

    c. Script Name: Create_Tables_Growing.sql

6. Create the TPC-E fixed tables.

    a. This step creates the fixed TPC-E tables as defined in Clause 2.6.1.9 in the TPC-E specification.

    b. The script also creates the check constraints on the fixed tables as required by the TPC-E specification in Clauses 2.2.4, 2.2.5., 2.2.6, and 2.2.7.

c. Script Name: Create_Tables_Fixed.sql

7. Install the TPC-E stored procedures and Data Maintenance Audit table.

    a. Script Names:

        - BrokerVolume.sql
        - CustomerPosition.sql
        - MarketFeed.sql
        - MarketWatch.sql
        - SecurityDetail.sql
        - TradeOrder.sql
        - TradeResult.sql
        - TradeStatus.sql
        - TradeLookup.sql
        - TradeUpdate.sql
        - DataMaintenance.sql
        - Create_DM_Audit_Table.sql

8. Install Trade ID Range Components

    a. This script installs the Get_Next_T_ID stored procedure

    b. Script Name: Get_Next_T_ID.sql

9. Setup the pre-load database options.

    a. This script sets the TPC-E database options for bulk loading.

    b. Script Name: Database_Options_1.sql

10. Load the database using the TPC-E provided EGenLoader.

    a. ODBC Load Type

        i. This step utilizes one or more Windows command files that are generated by the Microsoft TPC-E Benchmark Kit. These command files contain the appropriate EGenLoader settings based on the number of customers and the parallelism desired.

        ii. The EGenLoader.exe has the following input parameters:

            –b Beginning customer ordinal position
            –c Number of customers (for this instance)
            –t Number of customers (total in the database)
            –f Scale factor (customers per 1 tpsE)
            –w Number of Workdays (8-hour days)
            –i Directory for input files
            –l Type of load
            –s Database server
            –d Database name

        iii. Example: (10,000 Customers and 2 instances of EGenLoader)

1. ** EGenLoaderInstance1.cmd **
```
@ECHO OFF
@CD C:\MSTPCE.1.5.0-1007\EGen\Bin\Release\x86
@EGenLoader.exe -b 1 -c 5000 -t 10000 -f 500 -w 300 -i C:\MSTPCE. 1.5.0-
1007\EGen\Flat_In -l ODBC -s SQLPERFTPCE1 -d tpce
@GOTO PostProcessLoader%errorlevel%
:PostProcessLoader1
:PostProcessLoader2
:PostProcessLoader3
@ECHO.
@ECHO !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
@ECHO This instance of EGenLoader has encountered an error.
@ECHO Please capture any error information prior to pressing
@ECHO the Enter key since this window will close at that time.
@ECHO.
@ECHO The TPC-E build is being aborted.
@ECHO !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
@ECHO.
@cd..\..\..\SetupVBS
@copy EGenInstanceError1.txt ..\EGenInstanceError1.txt >nul
@CD ..
PAUSE
@GOTO END
:PostProcessLoader0
:END
@cd..\..\..\SetupVBS
@copy EGenInstanceFlag1.txt ..\EGenInstanceFlag1.txt >nul
@CD ..
```

2. ** EGenLoaderInstance2.cmd **
```
@ECHO OFF
@CD C:\MSTPCE. 1.5.0-1007\EGen\Bin\Release\x86
@EGenLoader.exe -b 5001 -c 5000 -t 10000 -f 500 -w 300 -i C:\MSTPCE. 1.5.0-
1007\EGen\Flat_In -l ODBC -s SQLPERFTPCE1 -d tpce
@GOTO PostProcessLoader%errorlevel%
:PostProcessLoader1
:PostProcessLoader2
:PostProcessLoader3
@ECHO.
@ECHO !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
@ECHO This instance of EGenLoader has encountered an error.
@ECHO Please capture any error information prior to pressing
@ECHO the Enter key since this window will close at that time.
@ECHO.
@ECHO The TPC-E build is being aborted.
@ECHO !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
@ECHO.
@cd..\..\..\SetupVBS
@copy EGenInstanceError2.txt ..\EGenInstanceError2.txt >nul
@CD ..
```

```
PAUSE
@GOTO END
:PostProcessLoader0
:END
@cd..\..\..\SetupVBS
@copy EGenInstanceFlag2.txt ..\EGenInstanceFlag2.txt >nul
@CD ..
```

b. Flat File Load Type

iv. This step utilizes one or more Windows command files that are generated by the Microsoft TPC-E Benchmark Kit. These command files contain the appropriate EGenLoader settings based on the number of customers and the parallelism desired.

v. The EGenLoader.exe has the following input parameters:

–b	Beginning customer ordinal position
–c	Number of customers (for this instance)
–t	Number of customers (total in the database)
–f	Scale factor (customers per 1 tpsE)
–w	Number of Workdays (8-hour days)
–i	Directory for input files
–l	Type of load
- o	Directory for output files
–s	Database server
–d	Database name

vi. Example: (10,000 Customers and 2 instances of EGenLoader)

1. ** EGenLoaderInstance1.cmd **

```
@ECHO OFF
@CD C:\MSTPCE.1.5.0-1007\EGen\Bin\Release\
@EGenLoader.exe -b 1 -c 5000 -t 10000 -f 500 -w 300 -i C:\MSTPCE. 1.5.0-
1007\EGen\Flat_In -l FLAT –o E:\ -s SQLPERFTPCE1 -d tpce
@GOTO PostProcessLoader%errorlevel%
:PostProcessLoader1
:PostProcessLoader2
:PostProcessLoader3
@ECHO.
@ECHO !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
@ECHO This instance of EGenLoader has encountered an error.
@ECHO Please capture any error information prior to pressing
@ECHO the Enter key since this window will close at that time.
@ECHO.
@ECHO The TPC-E build is being aborted.
@ECHO !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
@ECHO.
@cd..\..\..\SetupVBS
@copy EGenInstanceError1.txt ..\EGenInstanceError1.txt >nul
@CD ..
PAUSE
```

```
@GOTO END
:PostProcessLoader0
:END
@cd..\..\..\SetupVBS
@copy EGenInstanceFlag1.txt ..\EGenInstanceFlag1.txt >nul
@CD ..
```

2. ** EGenLoaderInstance2.cmd **

```
@ECHO OFF
@CD C:\MSTPCE. 1.5.0-1007\EGen\Bin\Release\
@EGenLoader.exe -b 5001 -c 5000 -t 10000 -f 500 -w 300 -i C:\MSTPCE. 1.5.0-
1007\EGen\Flat_In -l FLAT –o F: -s SQLPERFTPCE1 -d tpce
@GOTO PostProcessLoader%errorlevel%
:PostProcessLoader1
:PostProcessLoader2
:PostProcessLoader3
@ECHO.
@ECHO !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
@ECHO This instance of EGenLoader has encountered an error.
@ECHO Please capture any error information prior to pressing
@ECHO the Enter key since this window will close at that time.
@ECHO.
@ECHO The TPC-E build is being aborted.
@ECHO !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
@ECHO.
@cd..\..\..\SetupVBS
@copy EGenInstanceError2.txt ..\EGenInstanceError2.txt >nul
@CD ..
PAUSE
@GOTO END
:PostProcessLoader0
:END
@cd..\..\..\SetupVBS
@copy EGenInstanceFlag2.txt ..\EGenInstanceFlag2.txt >nul
@CD ..
```

vii. After the flat files are generated, the following bulk insert scripts will be executed

1. ** BulkInsert_1.sql **

```
USE  tpce
GO
PRINT 'BULK INSERT ACCOUNT_PERMISSION'
GO
BULK INSERT ACCOUNT_PERMISSION
FROM 'e:\AccountPermission.txt'
WITH ( FIELDTERMINATOR = '|',
     ROWS_PER_BATCH = 35500,
     TABLOCK,
     KEEPNULLS)
```

```
        GO
        PRINT 'BULK INSERT ADDRESS'
        GO
        BULK INSERT ADDRESS
        FROM 'e:\Address.txt'
        WITH ( FIELDTERMINATOR = '|',
             ROWS_PER_BATCH = 7501,
             TABLOCK)
        GO
PRINT 'BULK INSERT BROKER'
GO
BULK INSERT BROKER
FROM 'e:\Broker.txt'
WITH ( FIELDTERMINATOR = '|',
     ROWS_PER_BATCH = 50,
     TABLOCK,
     KEEPNULLS)
GO
PRINT 'BULK INSERT CASH_TRANSACTION'
GO
BULK INSERT CASH_TRANSACTION
FROM 'e:\CashTransaction.txt'
WITH ( FIELDTERMINATOR = '|',
     ROWS_PER_BATCH = 79488000,
     TABLOCK,
     KEEPNULLS)
GO
PRINT 'BULK INSERT CHARGE'
GO
BULK INSERT CHARGE
FROM 'e:\Charge.txt'
WITH ( FIELDTERMINATOR = '|',
     ROWS_PER_BATCH = 5,
     TABLOCK,
     KEEPNULLS)
GO
PRINT 'BULK INSERT COMMISSION_RATE'
BULK INSERT COMMISSION_RATE
FROM 'e:\CommissionRate.txt'
WITH ( FIELDTERMINATOR = '|',
     ROWS_PER_BATCH = 240,
     TABLOCK,
     KEEPNULLS)
GO
PRINT 'BULK INSERT COMPANY'
GO
BULK INSERT COMPANY
FROM 'e:\Company.txt'
WITH ( FIELDTERMINATOR = '|',
     ROWS_PER_BATCH = 2500,
```

```
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT COMPANY_COMPETITOR'
GO
BULK INSERT COMPANY_COMPETITOR
FROM 'e:\CompanyCompetitor.txt'
WITH ( FIELDTERMINATOR = '|',
     ROWS_PER_BATCH = 7500,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT CUSTOMER'
GO
BULK INSERT CUSTOMER
FROM 'e:\Customer.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 5000,
    TABLOCK)
GO
PRINT 'BULK INSERT CUSTOMER_ACCOUNT'
GO
BULK INSERT CUSTOMER_ACCOUNT
FROM 'e:\CustomerAccount.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 25000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT CUSTOMER_TAXRATE'
GO
BULK INSERT CUSTOMER_TAXRATE
FROM 'e:\CustomerTaxrate.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 10000,
    TABLOCK,
     KEEPNULLS)
GO
PRINT 'BULK INSERT DAILY_MARKET'
GO
BULK INSERT DAILY_MARKET
FROM 'e:\DailyMarket.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 4469625,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT EXCHANGE'
GO
BULK INSERT EXCHANGE
```

```
FROM 'e:\Exchange.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 4,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT FINANCIAL'
GO
BULK INSERT FINANCIAL
FROM 'e:\Financial.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 50000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT INDUSTRY'
GO
BULK INSERT INDUSTRY
FROM 'e:\Industry.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 102,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT HOLDING'
GO
BULK INSERT HOLDING
FROM 'e:\Holding.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 4406400,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT HOLDING_HISTORY'
GO
BULK INSERT HOLDING_HISTORY
FROM 'e:\HoldingHistory.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 115776000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT HOLDING_SUMMARY'
GO
BULK INSERT HOLDING_SUMMARY
FROM 'e:\HoldingSummary.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 248900,
    TABLOCK,
    KEEPNULLS)
```

```
GO
PRINT 'BULK INSERT LAST_TRADE'
GO
BULK INSERT LAST_TRADE
FROM 'e:\LastTrade.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 3425,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT NEWS_ITEM_TEMP'
GO
BULK INSERT NEWS_ITEM_TEMP
FROM 'e:\NewsItem.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 5000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT NEWS_XREF'
GO
BULK INSERT NEWS_XREF
FROM 'e:\NewsXref.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 5000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT SECTOR'
GO
BULK INSERT SECTOR
FROM 'e:\Sector.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 12,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT SECURITY'
GO
BULK INSERT SECURITY
FROM 'e:\Security.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 3425,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT SETTLEMENT'
GO
BULK INSERT SETTLEMENT
FROM 'e:\Settlement.txt'
```

```
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 86400000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT STATUS_TYPE'
GO
BULK INSERT STATUS_TYPE
FROM 'e:\StatusType.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 5,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT TAXRATE'
GO
BULK INSERT TAXRATE
FROM 'e:\TaxRate.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 320,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT TRADE'
GO
BULK INSERT TRADE
FROM 'e:\Trade.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 86400000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT TRADE_HISTORY'
GO
BULK INSERT TRADE_HISTORY
FROM 'e:\TradeHistory.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 207360000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT TRADE_TYPE'
GO
BULK INSERT TRADE_TYPE
FROM 'e:\TradeType.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 5,
    TABLOCK,
    KEEPNULLS)
GO
```

```
PRINT 'BULK INSERT WATCH_ITEM'
GO
BULK INSERT WATCH_ITEM
FROM 'e:\WatchItem.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 500000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT WATCH_LIST'
GO
BULK INSERT WATCH_LIST
FROM 'e:\WatchList.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 5000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT ZIP_CODE'
GO
BULK INSERT ZIP_CODE
FROM 'e:\ZipCode.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 14741,
    TABLOCK,
    KEEPNULLS)
GO
```

2. ** BulkInsert_2.sql **

```
USE  tpce
GO
PRINT 'BULK INSERT ACCOUNT_PERMISSION'
GO
BULK INSERT ACCOUNT_PERMISSION
FROM 'F:\AccountPermission.txt'
WITH ( FIELDTERMINATOR = '|',
     ROWS_PER_BATCH = 35500,
     TABLOCK,
     KEEPNULLS)
GO
PRINT 'BULK INSERT ADDRESS'
GO
BULK INSERT ADDRESS
FROM 'F:\Address.txt'
WITH ( FIELDTERMINATOR = '|',
      ROWS_PER_BATCH = 7501,
      TABLOCK,
     KEEPNULLS)
GO
PRINT 'BULK INSERT BROKER'
```

```
GO
BULK INSERT BROKER
FROM 'F:\Broker.txt'
WITH ( FIELDTERMINATOR = '|',
     ROWS_PER_BATCH = 50,
     TABLOCK,
     KEEPNULLS)
GO
PRINT 'BULK INSERT CASH_TRANSACTION'
GO
BULK INSERT CASH_TRANSACTION
FROM ‘F:\CashTransaction.txt'
WITH ( FIELDTERMINATOR = '|',
     ROWS_PER_BATCH = 79488000,
     TABLOCK,
     KEEPNULLS)
GO
PRINT 'BULK INSERT CHARGE'
GO
BULK INSERT CHARGE
FROM 'F:\Charge.txt'
WITH ( FIELDTERMINATOR = '|',
     ROWS_PER_BATCH = 5,
     TABLOCK,
     KEEPNULLS)
GO
PRINT 'BULK INSERT COMMISSION_RATE'
BULK INSERT COMMISSION_RATE
FROM 'F:\CommissionRate.txt'
WITH ( FIELDTERMINATOR = '|',
     ROWS_PER_BATCH = 240,
     TABLOCK,
     KEEPNULLS)
GO
PRINT 'BULK INSERT COMPANY'
GO
BULK INSERT COMPANY
FROM 'F:\Company.txt'
WITH ( FIELDTERMINATOR = '|',
     ROWS_PER_BATCH = 2500,
     TABLOCK,
     KEEPNULLS)
GO
PRINT 'BULK INSERT COMPANY_COMPETITOR'
GO
BULK INSERT COMPANY_COMPETITOR
FROM 'F:\CompanyCompetitor.txt'
WITH ( FIELDTERMINATOR = '|',
      ROWS_PER_BATCH = 7500,
     TABLOCK,
```

```
    KEEPNULLS)
GO
PRINT 'BULK INSERT CUSTOMER'
GO
BULK INSERT CUSTOMER
FROM 'F:\Customer.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 5000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT CUSTOMER_ACCOUNT'
GO
BULK INSERT CUSTOMER_ACCOUNT
FROM 'F:\CustomerAccount.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 25000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT CUSTOMER_TAXRATE'
GO
BULK INSERT CUSTOMER_TAXRATE
FROM ‘F:\CustomerTaxrate.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 10000,
    TABLOCK,
     KEEPNULLS)
GO
PRINT 'BULK INSERT DAILY_MARKET'
GO
BULK INSERT DAILY_MARKET
FROM 'F:\DailyMarket.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 4469625,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT EXCHANGE'
GO
BULK INSERT EXCHANGE
FROM 'F:\Exchange.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 4,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT FINANCIAL'
GO
BULK INSERT FINANCIAL
```

```
FROM 'F:\Financial.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 50000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT INDUSTRY'
GO
BULK INSERT INDUSTRY
FROM 'F:\Industry.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 102,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT HOLDING'
GO
BULK INSERT HOLDING
FROM 'F:\Holding.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 4406400,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT HOLDING_HISTORY'
GO
BULK INSERT HOLDING_HISTORY
FROM 'F:\HoldingHistory.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 115776000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT HOLDING_SUMMARY'
GO
BULK INSERT HOLDING_SUMMARY
FROM 'F:\HoldingSummary.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 248900,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT LAST_TRADE'
GO
BULK INSERT LAST_TRADE
FROM 'F:\LastTrade.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 3425,
    TABLOCK,
    KEEPNULLS)
```

```
GO
PRINT 'BULK INSERT NEWS_ITEM_TEMP'
GO
BULK INSERT NEWS_ITEM_TEMP
FROM 'F:\NewsItem.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 5000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT NEWS_XREF'
GO
BULK INSERT NEWS_XREF
FROM 'F:\NewsXref.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 5000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT SECTOR'
GO
BULK INSERT SECTOR
FROM 'F:\Sector.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 12,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT SECURITY'
GO
BULK INSERT SECURITY
FROM 'F:\Security.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 3425,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT SETTLEMENT'
GO
BULK INSERT SETTLEMENT
FROM 'F:\Settlement.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 86400000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT STATUS_TYPE'
GO
BULK INSERT STATUS_TYPE
FROM 'F:\StatusType.txt'
```

```
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 5,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT TAXRATE'
GO
BULK INSERT TAXRATE
FROM 'F:\TaxRate.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 320,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT TRADE'
GO
BULK INSERT TRADE
FROM 'F:\Trade.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 86400000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT TRADE_HISTORY'
GO
BULK INSERT TRADE_HISTORY
FROM 'F:\TradeHistory.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 207360000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT TRADE_TYPE'
GO
BULK INSERT TRADE_TYPE
FROM 'F:\TradeType.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 5,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT WATCH_ITEM'
GO
BULK INSERT WATCH_ITEM
FROM 'F:\WatchItem.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 500000,
    TABLOCK,
    KEEPNULLS)
GO
```

```
PRINT 'BULK INSERT WATCH_LIST'
GO
BULK INSERT WATCH_LIST
FROM 'F:\WatchList.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 5000,
    TABLOCK,
    KEEPNULLS)
GO
PRINT 'BULK INSERT ZIP_CODE'
GO
BULK INSERT ZIP_CODE
FROM 'F:\ZipCode.txt'
WITH ( FIELDTERMINATOR = '|',
    ROWS_PER_BATCH = 14741,
    TABLOCK,
    KEEPNULLS)
GO
```

11. Convert the NI_ITEM column to a varbinary(max).

    a. The EGenLoader code, when generating flat files, generates the NI_ITEM column as a text field. The TPC-E specification requires that the NI_ITEM column be defined as a LOB (varbinary(max) in SQL Server terms). The column is inserted as a varchar(max) then converted to a varbinary(max).

    b. Script Name: Convert_NI_ITEM_Data.sql

12. Optionally delete the flat files.

    a. If the user request, the flat files will be deleted.

13. Create the clustered indexes on the TPC-E fixed tables.

    a. This step creates the clustered indexes on the TPC-E tables that the TPC-E specification defines as a fixed table. (Clause 2.6.1.9)

    b. Script Name: Create_Clustered_Indexes_Fixed.sql

14. Create the clustered indexes on the TPC-E growing tables.

    a. This step creates the clustered indexes on the TPC-E tables that the TPC-E specification defines as a growing table. (Clause 2.6.1.11)

    b. Script Name: Create_Clustered_Indexes_Growing.sql

15. Create the clustered indexes on the TPC-E scaling tables.

    a. This step creates the clustered indexes on the TPC-E tables that the TPC-E specification defines as a scaling table. (Clause 2.6.1.10)

b. Script Name: Create_Clustered_Indexes_Scaling.sql

16. Create the non-clustered indexes on the TPC-E fixed tables.

    a. This step creates the non-clustered indexes on the TPC-E tables that the TPC-E specification defines as a fixed table. (Clause 2.6.1.9)

    b. Script Name: Create_NC_Indexes_Fixed.sql

17. Create the non-clustered indexes on the TPC-E scaling tables.

    a. This step creates the non-clustered indexes on the TPC-E tables that the TPC-E specification defines as a scaling table. (Clause 2.6.1.10)

    b. Script Name: Create_NC_Indexes_Scaling.sql

18. Create the non-clustered indexes on the TPC-E growing tables.

    a. This step creates the non-clustered indexes on the TPC-E tables that the TPC-E specification defines as a growing table. (Clause 2.6.1.11)

    b. Script Name: Create_NC_Indexes_Growing.sql

19. Create the foreign key constraints.

    a. The script creates the foreign key relationships that are defined in Clauses 2.2.4, 2.2.5, 2.2.6, and 2.2.7.

    b. Script Name: Create_FK_Constraints.sql

20. Setup the post-load database options.

    a. This script sets the TPC-E database options for runtime conditions.

    b. Script Name: Database_Options_2.sql

21. Populate the Trade ID Ranges table

    a. This step creates a TID Ranges table and sets the starting point for trade ID generation during runtime.

    b. Script Name: Create_TID_Ranges_Table.sql

After the successful completion of the above steps in order, the TPC-E database is created, populated, and prepared for runtime.

TPC-E BENCHMARK DATABASE POST SETUP (AUDIT)

After the TPC-E database creation and load has been completed. A series of scripts are run against the database to satisfy the audit requirements in the TPC-E specification. The scripts are:

1. Calculate the space used
    a. Script Names:
        - SPUsed.sql
        - SPFiles.sql
        - SPLog.sql
2. Database Check
    a. Script Names:
        - Create_DB_Audit_Tables.sql
        - DB_Check.sql
        - DB_Primary_Key_Check.sql
        - Drop_DB_Audit_Tables.sql
3. Database Tables
    a. Script Name: DB_Tables.sql
4. Insert Duplicates Tests
    a. Script name: Insert_Duplicates_Tests.sql
5. Referential Integrity Tests
    a. Script name: Referential_Integrity_Tests.sql