GURU® Data Extraction Tool (GDEX) Version 1.0 Prerelease

Savitar Corporation

Legal Notices

Limitation on Warranties and Liability

The documentation in these manuals is provided for reference purposes only. Savitar Corporation has taken great pains to ensure that the manuals reasonably conform to the software. However, Savitar Corporation reserves the right to make changes in specifications and other information in these publications without prior notice.

Savitar Corporation makes no warranties, either express or implied, respecting the software and documentation or its quality, performance, merchantability, or fitness for any particular purpose. Savitar Corporation software and documentation is licensed for use "as is." Savitar Corporation shall not be liable to the user or any other person or entity with respect to any liability, loss or damage caused by, or alleged to be caused by, the software and documentation or its use, whether direct, incidental, or consequential, according to the terms of the license agreement.

Trademarks

GURU Data Extraction Tool (GDEX)

This document describes the GURU Data Extraction Tool (GDEX) version 1.0 Prerelease.

Description

The GURU Data Extraction Tool (GDEX) version 1.0 Prerelease extracts data from GURU table files into an SQL Server command script that will recreate the GURU data in a Microsoft SQL Server database. The SQL script can be executed from the SQL Server Query Analyzer tool. No ODBC driver is required as GDEX generates native SQL for Microsoft SQL Server.

System Requirements

The GURU Data Extraction Tool (GDEX) version runs on a Microsoft Windows 32-bit (Windows 2000 or Windows XP) machine. GDEX may run on other versions of Windows (95, 98, etc.) but no testing has been done in those environments. Execution of the SQL script generated by GDEX requires Microsoft SQL Server 2000 (server and SQL Query Analyzer tool).

Installation

The GURU Data Extraction Tool (GDEX) version is shipped on a compact disc (CD) containing the GDEX executable (GDEX.EXE) and with example command and support files. No installation procedure is required; simply copy the files into a directory on the target machine .

Other Required Files

GDEX 1.0 Prerelease extracts data from GURU table files (.ITB extension); in order to use GDEX these files must be available on the machine where GDEX is installed, either locally or on a network share.

Operation: Open a Command Prompt

The GURU Data Extraction Tool (GDEX) tool is a command line utility that executes from the Command Prompt, accessible from Start->Programs->Accessories->Command Prompt or by entering CMD at the Start->Run menu.

Operation: Executing GDEX

Once the command prompt is available, execute GDEX from the command line using the command line options to specify the .ITB files to extract data from, the name of the SQL command file to be generated, and any desired options.

The generated SQL is sent to the standard output. To save the generated SQL to a file, use the -o option. For example, this command uses the provided sample files to generate SQL output.:

```
GDEX -odeptinfo.sql deptinfo.itb
```

In the above example, the -o flag specifies DEPTINFO.SQL as the output file and DEPTINFO.ITB is specified as as the source of the data.

When the enter key is pressed, a copyright banner appears followed by the message

```
Creating SQL script for table: DEPTINFO
```

When the command prompt reappears, the file DEPTINFO.SQL contains the generated SQL script.

Generating One Script for Many Tables

The GDEX tool can process multiple GURU .ITB tables in one pass; these may be specified either as a list of tables as shown below:

```
GDEX -othreetables.sql deptinfo.itb empinfo.itb qtrsal.itb
```

or as a wildcard to generate one script for all the .ITB files in the current directory:

```
GDEX -oalltables.sql *.itb
```

SQL Output

The output SQL contains an SQL CREATE statement to create the SQL table with the equivalent structure to the GURU table, followed by a series of SQL INSERT statements to insert the data from the GURU table into the created SQL table.

For the DEPTINFO example given, the output SQL is shown below:

```
CREATE TABLE DEPTINFO (
DEPT VARCHAR(10),
DEPTID VARCHAR(3)
) ;
GO
-- row number 1
INSERT INTO DEPTINFO VALUES ( 'Accounting' , '210' );
-- row number 2
INSERT INTO DEPTINFO VALUES ( 'Finance' , '220' );
-- row number 3
INSERT INTO DEPTINFO VALUES ( 'Legal' , '230' );
. . .
```

The CREATE statement creates a table of the same name as the GURU .ITB table, with column definitions for each field in the GURU table. For more details on data type conversions see the section “Generated Data Types”, below.

The INSERT statements each insert a row of data from the GURU .ITB table into the just-created SQL Server table.

For more information on the syntax of CREATE, INSERT, and other SQL statements, see the Microsoft SQL Server Books Online documentation, available by choosing Start->Programs->Microsoft SQL Server->Books Online from the Start menu.

Executing the SQL Script with SQL Query Analyzer

To execute the SQL script generated by GDEX, execute the Micrsoft SQL Server Query Analyzer tool by choosing Start->Programs->Microsoft SQL Server->Query Analyzer from the Start menu.

Choose File->Connect to connect to the SQL Server database in which you wish to add the tables generated from GURU, then choose File->Open to open the .SQL file

generated by GDEX. Click on the Execute Query button or press F5 to execute script once opened.

This completes the process of using GDEX to move data from GURU tables to SQL Server.

Modifying the SQL Script

Because the SQL script generated by GDEX is a text file, it may be easily edited to make corrections or changes specific to your installation. SQL Query Analyzer supports common editing functions in its script window.

Error Messages from SQL Query Analyzer

Error messages generated from SQL Query Analyzer when executing the SQL script may appear in the Messages window below the script text.

A commonly-encountered error message is Msg 2714, caused when the table created in the GDEX script already exists; for example if the DEPTINFO table was already in the database this message would appear in the SQL Query Analyzer window:

```
There is already an object named 'DEPTINFO' in the database.
```

This can be corrected by renaming the table in the SQL script with a global Replace operation (Edit->Replace or Ctrl-H in SQL Query Analyzer). If the message is appearing from repeated executions of the script, a DROP statement can be included to delete the table before re-creating it - see the -D option for GDEX in the GDEX Options section, below.

Generated Data Types

GDEX translates GURU data types to SQL Server data types as shown in the table below:

GURU Data Type	**SQL Server Data Type**
`STR n`	`VARCHAR (n)` -- variable-length character type with matching length `n`
`INT`	`SMALLINT` -- 16-bit integer
`NUM`	`REAL` -- 8-byte (double) floating point number.
`LOGIC`	`TINYINT` -- 8-bit number either 0 (FALSE) or 1 (TRUE).
`BLOB`	Not supported in this version of GDEX.
`VIRTUAL`	Not supported (can be implemented by creating a SQL view).

Handling Date Data

GURU does not have a built-in DATETIME data type as in SQL Server; instead, these fields are implemented as STR (character) fields and so are translated to VARCHAR by GDEX.

Because SQL Server supports entering character-string dates as strings in the INSERT command (for example, `INSERT INTO X VALUES ('12/25/2003', ...)` one can often merely change the VARCHAR data type for the date field in the CREATE command generated by GDEX into a DATETIME data type and the INSERT commands generated will insert date data correctly into the DATETIME column with no further changes needed to the INSERT command. Depending on the format of the data in the

GURU table, SQL Server date options may need to be adjusted (See SQL Server Books Online) or individual rows in the INSERT commands may need editing or cleanup.

Handling Decimal and Money Data

Similarly to date data, GURU does not have a built-in DECIMAL or MONEY data type as supported in SQL Server; instead, these fields are implemented as NUM (floating-point real number) fields and so are translated to REAL by GDEX. Because SQL Server supports entering DECIMAL or MONEY values with the same decimal point numeric format as REAL in the INSERT command (for example, `INSERT INTO X VALUES (25000.00000, ...)` one can often merely change the REAL data type for the decimal/money field in the CREATE command generated by GDEX into a DECIMAL or MONEY data type as desired and the INSERT commands generated will insert the decimal/money data correctly into the DECIMAL/MONEY column with no further changes needed to the INSERT commands. As with the date data, SQL Server options may need to be adjusted and individual rows that need editing or cleanup.

GDEX Command Line Options

The following command line options can be included on the GDEX command line to specify the behavior of the session. End users must supply their own words and letters for those shown in italic; the angle brackets (< >) beside those words must not be typed as part of the argument.

Argument	**Description**
`-o < filename.sql>`	Specifies the name of the output file, conventionally an SQL script with .SQL file extension (must be specified, the output file does not default to .SQL extension). If this argument is omitted from the command line, GDEX outputs the generated SQL to standard out (the console).
`-sql`	Generate SQL output; this is the default behavior so this option may be omitted.
`-m`	Include #MARK column in output; see “#MARK Option”, below.
`-c`	Generate only the SQL CREATE command for the table, omit the INSERT Statements for data.
`-d`	Generate DROP command to drop table before CREATE; see “DROP Option”, below.
`-g`	Omit GO commands from generated output; see “GO Command”, below.

DROP Option

The -D option may be specified to instruct GDEX to include a DROP statement preceding the CREATE command for each table. This is useful especially when one is debugging and editing the SQL script repeatedly, for example to change data types or clean up data as described previously. The DROP statement generated by GDEX is preceded by a SQL-Server-specific IF statement that checks for the existence of the table before issuing the DROP command; with this option the script can be executed repeatedly whether the table already exists or not. An example using the DEPTINFO table is shown below; to specify DROP, include -D on the command line:

```
GDEX -d -oDEPTINFO_DROP.SQL deptinfo.itb
```

The generated SQL script will include the following lines preceding the CREATE command:

```
IF EXISTS (select * from dbo.sysobjects where name='DEPTINFO' and
type = 'U')
DROP TABLE DEPTINFO ;
Go
```

GO Command

The GDEX tool includes GO batch commands following the DROP, CREATE, and every one thousand (1000) INSERT commands. GO is a command for SQL Query Analyzer instructing it to proceed with execution of the preceding SQL commands in the script. GDEX includes the GO command following the CREATE statement and every 1000 INSERT statements in order to prevent SQL batches from becoming too large, consuming excess memory and slowing down execution on busy servers. The GO commands may be omitted from the script (except for the one following the DROP command, which is required) by specifying the -G option:

```
GDEX -g -oDEPTINFO_NOGO.SQL deptinfo.itb
```

#MARK Option

GURU tables include a special column called the #MARK field which is a logical field indicating whether the current row is marked for deletion. All fields in which #MARK is set to TRUE are deleted when the GURU COMPRESS command is executed. By default, GDEX omits the #MARK field in the generated SQL output. If desired, this field can be included with the -M option. The SQL table is created with a column named _MARK of type TINYINT containing either 1 (TRUE) or 0 (FALSE) to indicate if the row had been marked for deletion in the GURU table. For example,

```
GDEX -m -oDEPTINFO_MARK.SQL deptinfo.itb
```