

gsqlcmd

Command Line Utility for Working with Databases

Version 2.1, December 17, 2014

User's Manual

Table of Content

Table of Content	2
Getting Started	4
Getting Started	4
Change History	5
Edition Comparison	7
System Requirements	8
End-User License Agreement	9
Using gsqlcmd	11
Using gsqlcmd	11
Database Connections	11
Executing SQL Scripts	11
Exporting Database Data	11
Using Variables in SQL Scripts	12
Internal Script Commands	12
Transaction Modes	12
Trace Mode	12
Importing CSV Data into Databases	12
Command Line	18
Command Line Modes	19
Execute	19
Insert	19
Merge	19
Fmt+	19
Fmt	19
Create	19
Ace	19
Bulk	19
Ini	19
LocalConfig UserConfig AppConfig	20
Connections	20
Help	20
Command Line Positional Parameters	20
<Connection>	20
<SQL Query> <SQL File>	21
<CSV File>	21
<Output File>	21
Query Execution Options	22
/CommandTimeout= <seconds>	22
/ConnectionTimeout= <seconds>	22
/NoTransaction	22
/InputCodePage= <codepage>	22
/OutputCodePage= <codepage>	22
/Set=parameter=value[;...]	22
/Trace	22
Common Output Options	23
/AddRowNum	23
/Append	23
/AsText AsCSV AsHTML	23
/DateTimeFormat= <datetime format>	23
/NoHeaders	23
CSV Output Options	23
/Add= <header=value>[<separator>...]	23
/Separator= <separator> tab	23
/QuoteChar= <char>	23
/EscapeChar= <char>	23
HTML Output Options	24
/NoTemplate	24

/Placeholder= <placeholder>	24
/Template= <HTML template file>	24
/Title= <title>	24
Code Generation Options	24
/Table= <table>	24
/fmt= <format file>	24
/InsertIdentity	24
/mssql sqlce mysql oracle db2 nuodb	24
Exit Codes	24
gConnectionManager	25
Configuration File	27
Product Registration	28
Selecting Product	28
Selecting Edition	29
Licensee Data	29
Online Registration	30
Registration by Email	31
Technical Support	33
Technical Support	33
Frequently Asked Questions	33

Getting Started

gsqcmd is a command line utility for working with databases. It allows:

- Executing SQL commands and scripts with parameters.
- Exporting data from databases into text, CSV, and HTML.
- Preparing configuration files for importing CSV files into databases.
- Importing CSV files into databases.

gsqcmd allows working with different database servers in the same manner and performing usual developer tasks with fewer efforts.

Note that you may load from the web and convert XML, JSON, HTML, CSV and text into CSV using the companion utility gwebcmd.

gsqcmd supports the following database platforms:

- Windows Azure SQL Database.
- Microsoft SQL Server.
- Microsoft SQL Server Compact.
- Oracle Database.
- IBM DB2.
- MySQL.
- MariaDB.
- NuoDB.

You may start learning about gsqcmd in the following topics:

- [Using gsqcmd](#)
- [Importing CSV Data into Databases](#)
- [Edition Comparison](#)
- [Command Line](#)
- [Connection Manager](#)
- [Configuration File](#)
- [Frequently Asked Questions](#)

Change History

Legend:

- [+] Added Feature
- [-] Fixed Bug
- [*] Improved/changed feature

Version 2.1, December 17, 2014

- [-] Adding results to text files has been fixed.
- [-] Registration issues have been fixed.

Version 2.0, November 25, 2014

- [+] gsqlcmd includes three editions: Express, Personal and Enterprise.
See [Edition Comparison](#) for details.
By default, gsqlcmd Express Edition is activated.
Use RegisterProduct.exe to register Personal or Enterprise edition after purchase.
- [+] Several modes to generate configuration files for importing CSV data into databases have been added.
See [Command Line Modes](#) for details.
- [+] The Insert mode to generate INSERT statements for importing CSV data has been added.
- [+] The Merge mode to generate INSERT and UPDATE statements for merging CSV data has been added.
- [+] gsqlcmd examples have been added to the installation package.
- [*] Help in CHM format has been improved.
The command line help has been splitted into short and extended forms.

Version 1.4, October 15, 2014

- [-] MySQL DateTime conversion issues with .NET Provider have been fixed.

Version 1.3, August 27, 2014

- [+] Microsoft SQL Server 2000 support has been added.
- [+] HTML templates can contain multiple place holders to insert result tables like {Sales}.
The target place holder can be specified in the command line as a parameter like
gsqlcmd FinanceDB Sales.sql Report.htm /placeholder={Sales}
- [*] SQL codes can contain "DECLARE <Parameter>" SQL statements.
Declared parameters are not populated with actual values and are sent to servers as is.
- [*] The output encoding for the console output is the same as the console encoding.
- [*] The utility processes all the returned result sets, not the first only.

Version 1.2, July 15, 2014

- [+] Support for NuoDB has been added.
NuoDB is an innovative database management system with advanced features:
the distributed cloud database, scale-out performance, geo-distribution, auto-administration, multi-tenancy.
NuoDB has also a free edition with 4GB database size limit.
Visit <http://www.nuodb.com>.
- [+] NuoDB ADO.NET Driver is installed with the setup package.
You do not need to install additional software to work with NuoDB.
- [*] Support for Oracle Data Provider for .NET has been added.
Oracle Data Provider for .NET 12c Release 2 (12.1.0.1.2) is installed with the setup package.
To use the provider, you must accept license terms at <http://www.oracle.com/technetwork/licenses/distribution-license-152002.html>.
- [*] Support for IBM DB2 .NET Provider has been added.
- [+] gsqlcmd can execute Microsoft SQL Server Compact Engine commands:
CreateDatabase, Compact, Repair, Shrink, Upgrade.
For example:
gsqlcmd "System.Data.SqlServerCe.4.0;Data Source=payments.sdf" CreateDatabase

- [!] NULL values in string columns of the CSV output are changed to "" to have consistent string value formats.
- [*] Oracle PL/SQL parsing has been improved.
- [+] The ConnectionTimeout command line option has been added.
- [+] The CommandTimeout command line option has been added.

Version 1.1, July 15, 2014

- [+] Support for SkySQL MariaDB has been added.
MariaDB is an open source MySQL compatible DBMS. All application features for MySQL are suitable for MariaDB.
Visit <http://mariadb.com>.

Version 1.0, February 25, 2014

- [+] The first release.

Edition Comparison

gsqcmd allows easily executing the following tasks:

1. Executing SQL queries and scripts with parameters.
2. Exporting database data into CSV, HTML, and text.
3. Preparing configuration files for importing CSV files into databases.
4. Importing CSV files into databases.

This one tool can replace sqlcmd, bcp, sqlplus, db2, and mysql in most scenarios with additional benefits.

It allows getting results with fewer efforts and on all major database platforms in the same way.

Feature	Express	Personal	Enterprise
Features available on all supported database platforms			
Executing SQL queries and scripts with parameters	✓	✓	✓
Exporting data into CSV, HTML, and text	✓	✓	✓
Generating CREATE TABLE statements for importing CSV files	✓	✓	✓
Generating INSERT statements for importing CSV files	500 rows	✓	✓
Generating MERGE statements for importing CSV files	500 rows	✓	✓
Generating schema.ini for importing CSV files using ODBC	✓	✓	✓
Using named connection strings, opened and encrypted	✓	✓	✓
Supported database servers: SQL Server, SQL Server Compact, MySQL, Oracle, DB2, NuoDB	✓	✓	✓
Features specific for Microsoft SQL Server			
Generating format files and SQL codes for OPENROWSET(BULK...)	✓	✓	✓
Generating SQL codes for OPENROWSET using MICROSOFT.ACE.OLEDB.12.0	✓	✓	✓
Licensing			
Price per copy*	Free	\$49.95	\$99.95
Commercial use	✓	✗	✓

* Prices are subject to change. Visit www.savetodb.com to see actual prices.

System Requirements

Supported Architectures:

- x86
- x64

Supported Operating Systems:

- Windows XP SP3
- Windows Vista SP1
- Windows 7
- Windows 8 / 8.1
- Windows Server 2003 SP3
- Windows Server 2008 / 2008 R2
- Windows Server 2012 / 2012 R2

Supported Versions of Microsoft SQL Server:

- Microsoft SQL Server 2000
- Microsoft SQL Server 2005
- Microsoft SQL Server 2008 / 2008 R2
- Microsoft SQL Server 2012 including Express LocalDB
- Microsoft SQL Server 2014 including Express LocalDB
- Windows Azure SQL Database

Supported Versions of Microsoft SQL Server Compact:

- Microsoft SQL Server Compact 3.5
- Microsoft SQL Server Compact 4.0

Supported Versions of Oracle Database:

- Oracle Database 10g Release 1
- Oracle Database 10g Release 2
- Oracle Database 11g Release 1
- Oracle Database 11g Release 2
- Oracle Database 12c Release 1

Supported Versions of IBM DB2:

- IBM DB2 9.5
- IBM DB2 9.7
- IBM DB2 10.1

IBM DB2 .NET Provider, IBM DB2 OLE DB Provider or IBM DB2 ODBC driver installed is required.

Supported Versions of MySQL:

- MySQL 5.0
- MySQL 5.1
- MySQL 5.2
- MySQL 5.5
- MySQL 5.6

Supported Versions of SkySQL MariaDB:

- MariaDB 5.1
- MariaDB 5.2
- MariaDB 5.3
- MariaDB 5.5
- MariaDB 10.0

Supported Versions of NuoDB:

- NuoDB 2.0.4

End-User License Agreement

This End-User License Agreement (EULA) is a legal agreement between you (either an individual or a single entity) and Gartle Technology Corporation for any gsqlcmd software, use examples and documentation (Software) that accompany this EULA.

YOU AGREE TO BE BOUND BY THE TERMS OF THIS EULA BY INSTALLING, COPYING, OR OTHERWISE USING THE SOFTWARE.

IF YOU DO NOT AGREE, DO NOT INSTALL, COPY, OR USE THE SOFTWARE.

Your licensing of Software is in accordance with the terms of the EULA in effect at the time of such licensing. By licensing Software, you accept and agree to the EULA in effect at such time.

1. GRANT OF LICENSE: Gartle Technology Corporation grants you the following rights provided that you comply with all terms and conditions of this EULA:
 - A. Installation and Use: You may install, use, access, display and run one (1) copy of the Software per license, on a non-exclusive basis and without right of sublicense or concurrent use, on a single computer or machine, such as a workstation.
Software has several editions with different features and limits described on the official website www.savetodb.com.
You may use gsqlcmd Express Edition for free. You may purchase and register the gsqlcmd Personal or Enterprise edition.
The commercial use of the gsqlcmd Personal Edition is not permitted.
 - B. Software Transfer: You may transfer the Software to a different internal workstation or user so long as you have purchased a License for each such workstation or user. You may not, however, transfer the Software to a Third Party.
 - C. Use of Examples: You may install, access, modify and use Software examples, including SQL codes and batch files, for your private or company internal purposes.
2. LIMITATIONS: You may not use, copy, modify, display, rent, lease, loan, transfer, distribute, download, merge, or make any translation or derivative work of the Software, except as expressly provided herein. You may not reverse engineer, decompile, or disassemble the Software, except and only to the extent that such activity is expressly permitted by applicable law notwithstanding this limitation.
3. INTELLECTUAL PROPERTY RIGHTS AND CONFIDENTIALITY: The Software, including methods, processes and/or techniques utilized therein, is owned by, proprietary to and valuable trade secrets of Gartle Technology Corporation and is protected by Russian Federation copyright law and international treaties. You agree to take no actions that impair or infringe Gartle Technology Corporation's intellectual property rights in the Software. You agree not to remove, efface or obscure any copyright notices, other proprietary markings or confidentiality legends placed upon or contained within the Software.
4. DISCLAIMER OF WARRANTIES: Gartle Technology Corporation disclaims all warranties concerning the Software and Services (if any), express, implied, or statutory, including without limitation, any warranties, duties or conditions of merchantability or fitness for a particular purpose, warranties of reliability or availability, of accuracy or completeness of responses, of results, of workmanlike effort, of lack of viruses, and of lack of negligence, all with regard to the Software, and the provision of or failure to provide support or other services, information, software, and related content through the Software or otherwise arising out of the use of the Software. Gartle Technology Corporation does not warrant that the Software will operate in combination with other software products selected by you, or that the Software will operate uninterrupted or error-free. Additionally, Gartle Technology Corporation and its suppliers provide the Software and Services AS IS AND WITH ALL FAULTS. THERE IS NO WARRANTY OR CONDITION OF TITLE, QUIET ENJOYMENT, QUIET POSSESSION, CORRESPONDENCE TO DESCRIPTION, OR NONINFRINGEMENT WITH REGARD TO THE SOFTWARE.
5. NO LIABILITY: TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL GARTLE TECHNOLOGY CORPORATION OR ITS SUPPLIERS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, PUNITIVE, INDIRECT, OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING, BUT NOT LIMITED TO, DAMAGES FOR LOSS OF PROFITS OR CONFIDENTIAL OR OTHER INFORMATION, FOR LOSS OF DATA, FOR BUSINESS INTERRUPTION, FOR PERSONAL INJURY, FOR LOSS OF PRIVACY, FOR FAILURE TO MEET ANY DUTY INCLUDING OF GOOD FAITH OR OF REASONABLE CARE, FOR NEGLIGENCE, AND FOR ANY OTHER PECUNIARY OR OTHER LOSS WHATSOEVER) ARISING OUT OF OR IN ANY WAY RELATED TO THE USE OF OR INABILITY TO USE THE SOFTWARE, THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT OR OTHER SERVICES, INFORMATION, SOFTWARE, AND RELATED CONTENT THROUGH THE SOFTWARE OR OTHERWISE ARISING OUT OF THE USE OF THE SOFTWARE, OR OTHERWISE UNDER OR IN CONNECTION WITH ANY PROVISION OF THIS EULA, EVEN IN THE EVENT OF THE FAULT, TORT (INCLUDING NEGLIGENCE), MISREPRESENTATION, STRICT LIABILITY, BREACH OF CONTRACT OF GARTLE TECHNOLOGY CORPORATION OR ANY SUPPLIER, AND EVEN IF GARTLE TECHNOLOGY CORPORATION OR ANY SUPPLIER HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
6. LIMITATION ON REMEDIES: Within the first thirty (30) days after your receipt of the Software, should you encounter and report to Gartle Technology Corporation within such time period a reproducible error that causes the Software not to perform in all material respects as set forth in the Software documentation, then Gartle Technology Corporation will, at its sole discretion, either: a) resolve the error or malfunction, and modify or replace the Software (if deemed necessary by Gartle Technology Corporation); or b) allow you to terminate this EULA with respect to the non-conforming Software and, upon your return of the Software to Gartle Technology Corporation, Gartle Technology Corporation shall provide you with the lesser of a) the actual damages incurred by you; or b) the amount you paid for the nonconforming Software. The remedies described in this section shall be your sole and exclusive remedies under this EULA.
7. GENERAL PROVISIONS
 - A. Reservation of Rights and Ownership: Gartle Technology Corporation reserves all rights not expressly granted to you in this EULA. The Software is licensed, not sold.
 - B. Consent to Use of Data: You agree that Gartle Technology Corporation and its affiliates may collect and use technical information gathered as part of the Software support services provided to you, if any, related to the Software. Gartle Technology Corporation may use this information solely to improve Gartle Technology Corporation products or to provide customized services or technologies to you and will not

disclose this information in a form that personally identifies you.

- C. **Links to Third Party Sites:** We may link to third party sites through the use of the Software. The third party sites are not under the control of Gartle Technology Corporation, and Gartle Technology Corporation is not responsible for the contents of any third party sites, any links contained in third party sites, or any changes or updates to third party sites. Gartle Technology Corporation is not responsible for webcasting or any other form of transmission received from any third party sites. Gartle Technology Corporation is providing these links to third party sites to you only as a convenience, and the inclusion of any link does not imply an endorsement by Gartle Technology Corporation of the third party site.
- D. **Additional Software/Services:** This EULA applies to updates, supplements, add-on components, or Internet-based services components, of the Software that Gartle Technology Corporation may provide to you or make available to you after the date you obtain your initial copy of the Software, unless Gartle Technology Corporation provides other terms along with the update, supplement, add-on component, or Internet-based services component. Gartle Technology Corporation reserves the right to discontinue any Internet-based services provided to you or made available to you through the use of the Software.
- E. **Upgrades:** To use Software identified as an upgrade, you must first be licensed for the software identified by Gartle Technology Corporation as eligible for the upgrade. After upgrading, you may no longer use the software that formed the basis for your upgrade eligibility.
- F. **Applicable Law:** This EULA is governed by the laws of the Russian Federation. Any legal action or proceeding relating to this EULA shall be instituted in a court of arbitration in the Moscow City, Russian Federation. Gartle Technology Corporation and you agree to submit to the jurisdiction of, and agree that venue is proper in, these courts in any such action or proceeding. The prevailing party in any action to enforce this EULA will be entitled to recover its attorney fees and costs in connection with such action.
- G. **Waiver:** The failure of either party to enforce any of the terms of this EULA shall not be construed as a waiver of future enforcement of that or any other term.
- H. **Entire Agreement and Severability:** This EULA (including any addendum or amendment to this EULA which is included with the Software) is the entire agreement between you and Gartle Technology Corporation relating to the Software and the support services (if any) and it supersedes all prior or contemporaneous oral or written communications, proposals and representations with respect to the Software or any other subject matter covered by this EULA. To the extent the terms of any Gartle Technology Corporation policies or programs for support services conflict with the terms of this EULA, the terms of this EULA shall control. If any provision of this EULA is held to be void, invalid, unenforceable or illegal, the other provisions shall continue in full force and effect.
- I. **Termination:** Without prejudice to any other rights, Gartle Technology Corporation may terminate this EULA if you fail to comply with the terms and conditions of this EULA. In such event, you must destroy all copies of the Software and all of its component parts and you will not be entitled to any refund of monies.

Using gsqlcmd

The gsqlcmd command line utility solves common database developer tasks:

- Executing SQL scripts for creating, modifying, and deleting database objects (DDL) and data (DML).
- Exporting database data into text, CSV, and HTML.
- Preparing configuration files for importing CSV files into databases.
- Importing CSV files into databases.

gsqlcmd solves these tasks easily than native database command line utilities and has the same syntax for all supported database servers.

Note that you may load from the web and convert XML, JSON, HTML, CSV and text into CSV using the companion utility gwecbcmd.

Command line formats:

```
gsqlcmd [execute] <connection> <sql query> [<output file>] [<options>]
gsqlcmd [execute] <connection> <sql file> [<output file>] [<options>]
```

```
gsqlcmd insert <connection> <csv file> [<output file>] [<options>]
gsqlcmd merge <connection> <csv file> [<output file>] [<options>]
```

```
gsqlcmd fmt+ <connection> <csv file> [<output file>] [<options>]
gsqlcmd fmt <connection> <csv file> [<output file>] [<options>]
gsqlcmd create|bulk|ini|ace <connection> <csv file> [<output file>] [<options>]
```

```
gsqlcmd localconfig | userconfig | appconfig
gsqlcmd connections
```

```
gsqlcmd help [chm]
```

See [Command Line](#) for a complete description.

You may use [exit codes](#) in batch files.

Database Connections

Executing an SQL command requires a database connection.

Specify the connection in the first positional parameter.

You may specify a connection string or the connection string name from the [configuration file](#).

The connection string is searched by the name in the following configuration files:

1. gsqlcmd.exe.config in the current directory;
2. gsqlcmd.exe.config in the %LocalAppData%\Gartle\gsqlcmd directory;
3. gsqlcmd.exe.config in the utility home directory.

You may use [gConnectionManager](#) to create, edit, copy, or delete connection strings in visual mode.

Use localconfig, userconfig, and appconfig commands to open the configuration files.

Use the connections command to list defined connection string names.

Executing SQL Scripts

There is an example of executing the rtd-setup-sqlce.sql script against the sqlce-rtd connection defined in the configuration file:

```
gsqlcmd sqlce-rtd rtd-setup-sqlce.sql
```

There is an example of executing the rtd-setup-sqlce.sql script against the Microsoft SQL Server Compact rtd.sdf database:

```
gsqlcmd "Data Source=rtd.sdf" rtd-setup-sqlce.sql
```

There is an example of executing the rtd-setup-mssql.sql script against the Microsoft SQL Server RTD database:

```
gsqlcmd "Provider=SQLOLEDB;Data Source=.\SQLEXPRESS;Integrated Security=SSPI;Initial Catalog=RTD" rtd-setup-mssql.sql
```

Exporting Database Data

There is an example of executing an SQL statement with the output to the console:

```
gsqlcmd mysql-rtd "SELECT * FROM INFORMATION_SCHEMA.TABLES"
```

By default, the text output is used for the console and for the files with the *.txt extensions.

There is an example of executing the export-quotes.sql script with the output to the quotes.csv file in the CSV format:

```
gsqcmd db2-rtd export-quotes.sql quotes.csv
```

The CSV output format is used for files with the *.csv extensions.

If you need to redirect the output in the CSV format, specify the CSV format explicitly. For example:

```
gsqcmd rtd-db2 export-quotes.sql /asCSV >> quotes.csv
```

Also you may specify additional options for output:

```
gsqcmd rtd-db2 export-quotes.sql quotes.csv /separator=, /datetimeformat=yyyy-MM-dd
```

There is an example of HTML output:

```
gsqcmd rtd-ora export-quotes.sql quotes.htm
```

This command uses the default built-in HTML template. You may specify using your template:

```
gsqcmd rtd-ora export-quotes.sql quotes.htm /template=table-template.htm
```

You may turn off using the template using the /notemplate option.

Using Variables in SQL Scripts

SQL commands and scripts can use values from the /set option of the command line.

For example:

```
gsqcmd mssql-rtd "SELECT * FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME = @Table" /set=Table=RealTimeTables
```

```
gsqcmd mysql-rtd "SELECT * FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME = :Table" /set=Table=real_time_tables
```

Use declarations in scripts as @<Parameter> for Microsoft SQL Server and Microsoft SQL Server Compact, and as :<Parameter> for MySQL, MariaDB, Oracle Database, IBM DB2, and NuoDB.

Do not use variables with the scripts that contain variable declarations as SQL statements for database servers.

For example, in create statements for stored procedures and triggers.

Internal Script Commands

SQL scripts can contain the -- print <Message> command that is executed internally and outputs the message.

It is useful for monitoring script execution progress.

This command works for all databases servers in the same manner, even the server does not supports print commands.

For example:

```
-- print Table ColumnTranslation has been created
```

Transaction Modes

SQL commands are executed in the ReadCommitted transaction isolation level.

The default rule is "all or nothing".

You may turn off the transaction mode using the /notransaction option.

It is useful, especially, for deleting database objects using scripts as some of the deleted objects may not exist.

Trace Mode

If a script has errors or is being executed for long time, turn on the trace mode using the /trace option.

By default, trace messages are written to the console. You may redirect the trace messages using the gsqlcmd.exe.config configuration file.

Importing CSV Data into Databases

gsqcmd allows generating the following files for importing CSV data into databases:

- CREATE TABLE SQL statements.
- Format files for OPENROWSET(BUCK...) import (SQL Server only).
- INI files for ODBC drivers.

- INSERT INTO ... SELECT statements for OPENROWSET(BUCK...) import (SQL Server only).
- INSERT INTO ... SELECT statements for OPENROWSET using MICROSOFT.ACE.OLEDB.12.0 (SQL Server only).
- INSERT statements.
- INSERT and UPDATE statements.

Note that you may load from the web and convert XML, JSON, HTML, CSV and text into CSV using the companion utility gwebcmd.

Creating Format Files

Originally, format files are used with the Microsoft SQL Server bcp command line utility and the OPENROWSET(BUCK...) function.

gsqlcmd allows generating such files and uses them as column name data sources for other modes.

So you may generate the format file first, edit it, and then generate other files.

For example, the source CSV file, payments.csv, has the following data:

```
ID;Date;Sum;"Account Name";"Company Name";"Item Name";"Comment"
1;01/10/2014 00:00:00;200000;"My Bank";"Rose, Inc";"Revenue";""
2;01/10/2014 00:00:00;-50000;"My Bank";"Land, Inc";"Expenses";""
```

You may generate the format file using the fmt mode:

```
gsqlcmd fmt payments.csv payments.fmt
```

and edit the column names (remove spaces, for example):

```
9.0
7
1  SQLCHAR  0  255  ";"          1  "ID"          ""
2  SQLCHAR  0  255  ";"          2  "Date"        ""
3  SQLCHAR  0  255  ";\\"        3  "Sum"         ""
4  SQLCHAR  0  255  "\";\\"      4  "AccountName" ""
5  SQLCHAR  0  255  "\";\\"      5  "CompanyName" ""
6  SQLCHAR  0  255  "\";\\"      6  "ItemName"    ""
7  SQLCHAR  0  255  "\"\r\n"    7  "Comment"     ""
```

You may disable column import replacing the column index (column 6) with 0.

Creating Tables in Databases

Then you may create the CREATE TABLE statement:

```
gsqlcmd create payments.csv payments.create.sql /table=dbo.Payment /fmt=payments.fmt
```

You may specify the target database platform using the options: /mssql, /sqlce, /mysql, /oracle, /db2, or /nuodb.

The result in payments.create.sql:

```
CREATE TABLE [dbo].[Payment] (
    [ID] int IDENTITY(1,1) NOT NULL,
    [Date] datetime NULL,
    [Sum] int NULL,
    [AccountName] nvarchar(255) NULL,
    [CompanyName] nvarchar(255) NULL,
    [ItemName] nvarchar(255) NULL,
    [Comment] nvarchar(255) NULL,
    CONSTRAINT [PK_Payment_dbo] PRIMARY KEY CLUSTERED (
        [ID] ASC
    )
)
GO
print 'Table [dbo].[Payment] has been created'
GO
```

Verify and change column data types. In the example, the [Sum] column created as int as the source column contains the interger values only.

You may change int to money or to float, for example.

The you may execute the CREATE TABLE code using the execute mode:

```
gsqlcmd execute <Connection> payments.create.sql
```

Note that you may create named connections in a visual mode.

Use one of the following:

```
gsqlcmd appconfig
gsqlcmd userconfig
gsqlcmd localconfig
```

And you may list available connections using the command:

```
gsqlcmd connections
```

Importing CSV Data into Microsoft SQL Server using BCP

You may import CSV data into Microsoft SQL Server using the bcp command line utility included into Microsoft SQL Server client packages.

For example, you may import CSV data from payments.csv into the dbo.Payment table using the following command:

```
bcp "dbo.Payment" in payments.csv -S . -d <database> -T -f payments.fmt -E
```

The -E option keeps identity values.

Note that the bcp utility don't support the UTF-8 encoding.

So you may convert CSV data into Windows ANSI encoding or to use methods described below.

See details about the bcp utility at <http://msdn.microsoft.com/en-us/library/ms162802.aspx>.

Creating and Using OPENROWSET(BULK...) Statements

To generate the INSERT INTO ... SELECT use the following command:

```
gsqlcmd bulk payments.csv payments.insert.bulk.sql /table=dbo.Payment /fmt=payments.fmt /insertIdentity
```

The result file:

```
SET IDENTITY_INSERT dbo.Payment ON;
INSERT INTO dbo.Payment
( [ID]
, [Date]
, [Sum]
, [AccountName]
, [CompanyName]
, [ItemName]
, [Comment]
)
SELECT
t.[ID]
, t.[Date]
, t.[Sum]
, t.[AccountName]
, t.[CompanyName]
, t.[ItemName]
, t.[Comment]
FROM
(
SELECT
[ID] AS [ID]
, [Date] AS [Date]
, [Sum] AS [Sum]
, [AccountName] AS [AccountName]
, [CompanyName] AS [CompanyName]
, [ItemName] AS [ItemName]
, [Comment] AS [Comment]
FROM
OPENROWSET(
BULK 'D:\payments.csv',
FORMATFILE = 'D:\payments.fmt',
CODEPAGE = '1250',
FIRSTROW = 2) t
) t
SET IDENTITY_INSERT dbo.Payment OFF;
```

If you do not specify the /table option, the INSERT INTO header is not generated.

Pay attention to the /insertIdentity option. It inserts SET IDENTITY_INSERT statements to insert the identity column from the source CSV file. As another option, you may skip the source identity column specifying 0 in column 6 in the format file.

Note that the OPENROWSET function requires absolute paths to the CSV and format files. You have to edit paths for your real paths.

You may import the data using this file, if the paths are available for your SQL Server instance.

Execute the code using SQL Server Management Studio, the native SQL Server sqlcmd utility or gsqlcmd like:

```
gsqlcmd execute <Connection> payments.insert.bulk.sql
```

Also you may insert this code into a stored procedure.

You may check loaded data using the SELECT statement in the command line. For example:

```
gsqcmd execute <Connection> "SELECT * FROM dbo.Payment"
```

The result:

ID	Date	Sum	AccountName	CompanyName	ItemName	Comment
1	01/10/2014 00:00:00	200000	My Bank	Rose, Inc	Revenue	
2	01/10/2014 00:00:00	-50000	My Bank	Land, Inc	Expenses	

Creating SCHEMA.INI Sections for Using with ODBC

You may easily create schema.ini file sections for further using with ODBC.

For example, execute:

```
gsqcmd ini payments.csv payments.ini /fmt=payments.fmt
```

The result in payments.ini:

```
[payments.csv]
ColNameHeader=True
Format=Delimited(;)
MaxScanRows=100
CharacterSet=1250
Col1="ID" Char Width 255
Col2="Date" Char Width 255
Col3="Sum" Char Width 255
Col4="AccountName" Char Width 255
Col5="CompanyName" Char Width 255
Col6="ItemName" Char Width 255
Col7="Comment" Char Width 255
```

Copy and paste these rows into the schema.ini file.

Creating and Using OPENROWSET with MICROSOFT.ACE.OLEDB.12.0

The bcp utility and the OPENROWSET(BUCK...) function don't support the UTF-8 encoding.

As a one solution, you may use the MICROSOFT.ACE.OLEDB.12.0 provider with the OPENROWSET function.

You may download the MICROSOFT.ACE.OLEDB.12.0 provider at

<http://www.microsoft.com/en-us/download/details.aspx?id=13255>

To generate the SQL code, use the following command:

```
gsqcmd ace payments.csv payments.insert.ace.sql /table=dbo.Payment /fmt=payments.fmt
```

The result in payments.insert.ace.sql:

```
INSERT INTO dbo.Payment
( [ID]
, [Date]
, [Sum]
, [AccountName]
, [CompanyName]
, [ItemName]
, [Comment]
)
SELECT
t.[ID]
, t.[Date]
, t.[Sum]
, t.[AccountName]
, t.[CompanyName]
, t.[ItemName]
, t.[Comment]
FROM
(
SELECT
[ID] AS [ID]
, [Date] AS [Date]
, [Sum] AS [Sum]
, [AccountName] AS [AccountName]
, [CompanyName] AS [CompanyName]
, [ItemName] AS [ItemName]
, [Comment] AS [Comment]
FROM
OPENROWSET ('MICROSOFT.ACE.OLEDB.12.0',
'Text;Database=D:\;HDR=Yes;Format=Delimited(;)',
'SELECT * FROM [payments.csv]') t
) t
```

To use this code you have to insert [payments.csv] section into the schema.ini file. See the previous topic.

You may execute the code using SQL Server Management Studio, the native SQL Server sqlcmd utility or gsqlcmd like:

```
gsqlcmd execute <Connection> payments.insert.ace.sql
```

Also you may insert this code into a stored procedure.

Creating INSERT Statements for Importing CSV Data

You may generate INSERT statements and execute them using your favorite database IDE, native database command line utility, or gsqlcmd.

For example:

```
gsqlcmd insert AzureDemo payments.csv payments.insert.sql /table=dbo.Payment /fmt=payments.fmt
```

The result in the payments.insert.sql:

```
INSERT INTO [dbo].[Payment] ([Date], [Sum], [AccountName], [CompanyName], [ItemName])
VALUES ('20140110 00:00:00.000', 200000, N'My Bank', N'Rose, Inc', N'Revenue');
INSERT INTO [dbo].[Payment] ([Date], [Sum], [AccountName], [CompanyName], [ItemName])
VALUES ('20140110 00:00:00.000', -50000, N'My Bank', N'Land, Inc', N'Expenses');
```

Note that the insert mode requires the connection as the first parameter.

gsqlcmd loads the target table definition and generates the code specific for the target database platform.

You may specify the /insertIdentity option to insert identity values from the source CSV file.

It is easy to execute the generated code using gsqlcmd:

```
gsqlcmd execute AzureDemo payments.insert.sql
```

gsqlcmd Express Edition allows generating INSERT statements for CSV files with less than 500 rows.

You may purchase Personal or Enterprise Edition to remove this limit.

Creating INSERT and UPDATE Statements for Merging CSV Data

You may generate INSERT and UPDATE statements to merge CSV data into desired table.

For example:

```
gsqlcmd merge AzureDemo payments.csv payments.merge.sql /table=dbo.Payment /fmt=payments.fmt
```

The result in the payments.merge.sql:

```
UPDATE [dbo].[Payment]
SET
    [Date] = s.[Date]
    , [Sum] = s.[Sum]
    , [AccountName] = s.[AccountName]
    , [CompanyName] = s.[CompanyName]
    , [ItemName] = s.[ItemName]
    , [Comment] = s.[Comment]
FROM
    [dbo].[Payment] t INNER JOIN (
    SELECT 1 AS [ID], '20140110 00:00:00.000' AS [Date], 200000 AS [Sum], N'My Bank' AS [AccountName],
        N'Rose, Inc' AS [CompanyName], N'Revenue' AS [ItemName], NULL AS [Comment]
    UNION ALL SELECT 2 AS [ID], '20140110 00:00:00.000' AS [Date], -50000 AS [Sum], N'My Bank' AS [AccountName],
        N'Land, Inc' AS [CompanyName], N'Expenses' AS [ItemName], NULL AS [Comment]
    ) s ON t.[ID] = s.[ID];

INSERT INTO [dbo].[Payment] ([Date], [Sum], [AccountName], [CompanyName], [ItemName], [Comment]) SELECT s.[Date],
    s.[Sum], s.[AccountName], s.[CompanyName], s.[ItemName], s.[Comment]
FROM (
    SELECT s.[ID], s.[Date], s.[Sum], s.[AccountName], s.[CompanyName], s.[ItemName], s.[Comment]
    FROM (
        SELECT 1 AS [ID], '20140110 00:00:00.000' AS [Date], 200000 AS [Sum], N'My Bank' AS [AccountName],
            N'Rose, Inc' AS [CompanyName], N'Revenue' AS [ItemName], NULL AS [Comment]
        UNION ALL SELECT 2 AS [ID], '20140110 00:00:00.000' AS [Date], -50000 AS [Sum], N'My Bank' AS [AccountName],
            N'Land, Inc' AS [CompanyName], N'Expenses' AS [ItemName], NULL AS [Comment]
        ) s
    LEFT OUTER JOIN [dbo].[Payment] t ON t.[ID] = s.[ID] WHERE t.[ID] IS NULL
    ) s;
GO
print 'Processed 2 total records';
```

Note that the merge mode requires the connection as the first parameter.

gsqlcmd loads the target table definition and generates the code specific for the target database platform.

You may specify the /insertIdentity option to insert identity values from the source CSV file.

It is easy to execute the generated code using gsqlcmd:

```
gsqlcmd execute AzureDemo payments.merge.sql
```

gsqlcmd Express Edition allows generating INSERT and UPDATE statements for CSV files with less than 500 rows. You may purchase Personal or Enterprise Edition to remove this limit.

Command Line

Command Line Modes:

```
gsqlcmd [execute] <connection> <sql query> [<output file>] [<options>]
gsqlcmd [execute] <connection> <sql file> [<output file>] [<options>]

gsqlcmd insert <connection> <csv file> [<output file>] [<options>]
gsqlcmd merge <connection> <csv file> [<output file>] [<options>]

gsqlcmd fmt+ <connection> <csv file> [<output file>] [<options>]
gsqlcmd fmt <connection> <csv file> [<output file>] [<options>]
gsqlcmd create|bulk|ini|ace <connection> <csv file> [<output file>] [<options>]

gsqlcmd localconfig | userconfig | appconfig
gsqlcmd connections

gsqlcmd help [chm]
```

<Connection> Format:

```
<ConnectionString name> ConnectionString name from gsqlcmd.exe.config
or <ConnectionString> ConnectionString for OLEDB, ODBC, and DSN
or <ProviderName>;<ConnectionString> ConnectionString for .NET providers
```

Query Execution Options:

```
/commandTimeout=<seconds>
/connectionTimeout=<seconds>
/noTransaction
/inputCodepage=<codepage>
/outputCodepage=<codepage>
/set=<parameter=value>[;...]
/trace
```

Common Output Options:

```
/addRowNum
/append
/asText
/asCsv
/asHtml
/dateTimeFormat=<format>
/noHeaders
```

CSV Output Options:

```
/add=<header=value>[<separator>...]
/separator=<separator>|Tab
/quoteChar=<char>
/escapeChar=<char>
```

HTML Output Options:

```
/noTemplate
/placeholder=<placeholder>
/template=<HTML template file>
/title=<title>
```

Code Generation Options:

```
/table=<target database table>
/fmt=<format file>
/insertIdentity

/mssql
/sqlce
/mysql
/oracle
/db2
/nuodb
```

Command Line Modes

Execute

Executes the SQL query or SQL script against the specified connection and outputs the result.

The mode is used to execute queries, export and import data.

Keyword "execute" can be omitted.

Insert

Generates insert statements for the specified CSV file.

The /table option is used to specify the target table and the /fmt= option is used to specify the format file with actual column names.

You may execute the generated SQL file using the execute mode.

gsqcmd Express Edition has limit in 500 rows.

Merge

Generates insert and update statements to merge data from the specified CSV file.

The /table option is used to specify the target table and the /fmt= option is used to specify the format file with actual column names.

You may execute the generated SQL file using the execute mode.

gsqcmd Express Edition has limit in 500 rows.

Fmt+

Generates all configuration files for importing CSV files using different techniques.

This mode is equivalent to executing the series of Fmt, Create, Ace, Buck, and Ini.

The format file is created only if it is not exist. If the format file exists, it used as a column name source.

You may run gsqlcmd in this mode twice: the first run for generating all files as is, and the second run after aditing the format file.

Fmt

Generates format files for importing CSV files into Microsoft SQL Server using the OPENROWSET(BUCK...) function.

Also the format files are used as a column name sources for other modes.

Create

Generates CREATE TABLE statements for further importing CSV files.

The /table option is used to specify the target table and the /fmt= option is used to specify the format file with actual column names.

Ace

Generates OPENROWSET statements for importing CSV files using the MICROSOFT.ACE.OLEDB.12.0 provider.

The /table option is used to specify the target table and the /fmt= option is used to specify the format file with actual column names.

OPENROWSET using the MICROSOFT.ACE.OLEDB.12.0 provider allows importing data in the UTF-8 encoding that is not supported in the OPENROWSET BULK statement.

Bulk

Generates OPENROWSET(BULK...) statements for importing CSV files.

The /table option is used to specify the target table and the /fmt= option is used to specify the format file with actual column names.

Ini

Generates decriptions for the schema.ini file that is used for importing CSV files using ODBC drivers into Microsoft Excel or into databases.

The /fmt= option is used to specify the format file with actual column names.

LocalConfig | UserConfig | AppConfig

Starts [gConnectionManager](#) to edit named connections in [configuration files](#):

- LocalConfig - gsqlcmd.exe.config in the current directory.
- UserConfig - gsqlcmd.exe.config in the %LocalAppData%\Gartle\gsqlcmd folder.
- AppConfig - gsqlcmd.exe.config in the utility folder.

Connections

Lists all connections from all configuration files described above.

Help

Starts extended command line help or starts this CHM help file if the chm parameter specified.

Command Line Positional Parameters

<Connection>

Defines a connection to a database.

There are three options to specify the connection:

1. A connection name defined in gsqlcmd.exe.config (or gsqlcmd32.exe.config for gsqlcmd32.exe).
2. A connection string for OLEDB, ODBC, and DSN.
3. A .NET Framework data provider name and a connection string separated by semicolon.

The first option is the most handy as you can use short names and store encrypted connection strings in the configuration files.

For example, gsqlcmd.exe.config can contain the following lines:

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <connectionStrings>
    <add name="AzureDemo" connectionString="Provider=SQLOLEDB;Data Source=.\SQLEXPRESS;Initial Catalog=AzureDemo;Integrated Security=SSPI"
      providerName="System.Data.OleDb" />
    <add name="mssql-rttd" connectionString="Data Source=.;Initial Catalog=RTD;Password=r#td_2014_abc!;User ID=rttd"
      providerName="System.Data.SqlClient" />
    <add name="sqlce-rttd" connectionString="Data Source=%LOCALAPPDATA%\Gartle\RealTimeToDB\rttd.sdf"
      providerName="System.Data.SqlServerCe.4.0" />
    <add name="mysql-rttd" connectionString="Server=localhost;Password=r#td_2014_abc!;User ID=rttd;Database=rttd"
      providerName="MySql.Data.MySqlClient" />
    <add name="ora-rttd" connectionString="Provider=OraOLEDB.Oracle;Password=r#td_2014_abc!;User ID=RTD;Data Source=Oracle/Oracl;PLSQ
LRSet=True"
      providerName="System.Data.OleDb" />
    <add name="db2-rttd" connectionString="Driver=IBM DB2 ODBC DRIVER;Hostname=DB2;Port=50000;Protocol=TCPIP;Database=SAMPLE;Pwd=r#t
d_2014_abc!;UID=RTD;LONGDATACompat=1"
      providerName="System.Data.Odbc" />
    <add name="nuodb-rttd" connectionString="Server=localhost;Password=r#td_2014_abc!;User=RTD;Database=rttd"
      providerName="NuoDb.Data.Client" />
  </connectionStrings>
</configuration>
```

This allows using names like AzureDemo, mssql-rttd, sqlce-rttd, and others as the command line connection parameter.

Use LocalConfig, UserConfig, or AppConfig gsqlcmd modes to edit configurations files in visual mode using [gConnectionManager](#).

The second option allows specifying only the connection string if an OLE DB provider, ODBC driver, or a DSN file is used.

For example:

```
gsqlcmd "Provider=SQLOLEDB;Data Source=.\SQLEXPRESS;Initial Catalog=AzureDemo;Integrated Security=SSPI" framework-install-en.sql
```

You may pass connection strings in environment variables. For example:

```
set connection=Provider=SQLOLEDB;Data Source=.\SQLEXPRESS;Initial Catalog=AzureDemo;Integrated Security=SSPI
gsqlcmd "%connection%" framework-install-en.sql
```

Use double quotes for such variables.

The third connection option with a provider name before the connection string is used for .NET Framework data providers.

For example:

```
gsqcmd "MySQL.Data.MySqlClient;Server=localhost;Password=r#td_2014_abc!;User ID=rtd;Database=rtd" framework-install-en.sql
```

<SQL Query> | <SQL File>

The second mandatory positional parameter of the execute mode specifies an SQL query or an SQL script file name.

For example:

```
gsqcmd rtd-ora "SELECT * FROM SYS.ALL_USERS"
```

```
gsqcmd rtd-ora rtd-setup-ora.sql
```

The utility checks the file with the parameter value and reads the script if the file exists. Otherwise, it sends the value to a server as a query.

<CSV File>

The parameter specifies the CSV file used for generating configuration files and SQL statements.

<Output File>

This optional parameter specifies an output file name.

The console is used by default.

The output file extension is used to determine the default output format for the execute mode:

- *.txt - text format;
- *.csv - CSV;
- *.htm or *.html - HTML.

Query Execution Options

`/CommandTimeout=<seconds>`

This option is used to change the timeout of command execution.

`/ConnectionTimeout=<seconds>`

This option is used to change the server connection timeout.

`/NoTransaction`

Disables the transaction mode for executing queries and scripts.

You may turn off the transaction mode, for example, for deleting database objects using scripts.

Otherwise, the scripts can be discarded by rollback if some of deleted objects are not exist.

`/InputCodePage=<codepage>`

Defines the input file code page.

Example:

```
/inputcodepage=65001
```

`/OutputCodePage=<codepage>`

Defines the output file code page.

Example:

```
/outputcodepage=1250
```

`/Set=parameter=value>[;...]`

Defines SQL statement or script parameter values.

For example:

```
gsqlcmd mssql-rtd "SELECT * FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME = @Table" /set=Table=RealTimeTables  
gsqlcmd mysql-rtd "SELECT * FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME = :Table" /set=Table=real_time_tables
```

Use the following parameter declarations:

@Parameter for Microsoft SQL Server and Microsoft SQL Server Compact,
:Parameter for MySQL, MariaDB, Oracle Database, IBM DB2, and NuoDB.

Do not use the /set option with scripts that contain parameter declarations as a part of SQL code for a database server.
For example, if the script contains codes for creating stored procedures and triggers.

Parameter types are determined by value types: string, number, or datetime.

Use single quotes to define a string type for a number value. For example: '123' instead of 123.

`/Trace`

Turns on tracing SQL commands sent to a server.

Use this mode to debug scripts or to measure execution time of SQL commands.

By default, the trace messages are written to the console.

You may define the log file for trace messages in gsqlcmd.exe.config in the gsqlcmd home directory.

Common Output Options

/AddRowNum

If the option is specified, the first column with row numbers is added to the output.

/Append

If the option is specified, the data are added to the output file. Otherwise, the existing file is replaced by a new one.

/AsText | AsCSV | AsHTML

Use this option to specify the output format, if the format can't be determined by the output file extension.

/DateTimeFormat=<datetime format>

This option is used to specify the format for datetime values in the output.

See [http://msdn.microsoft.com/en-us/library/zdtaw1bw\(v=vs.100\).aspx](http://msdn.microsoft.com/en-us/library/zdtaw1bw(v=vs.100).aspx) about the format string.

Example:

```
gsqcmd rtd-db2 "SELECT * FROM RTD.QUOTES_YAHOO" QUOTES_YAHOO.CSV /datetimeformat=yyyy-MM-dd
```

Use double quotes to specify formats with spaces. For example:

```
"/datetimeformat=yyyy-MM-dd hh:mm:ss"
```

/NoHeaders

Disables header output.

CSV Output Options

/Add=<header=value>[<separator>...]

This option is used to specify additional data in the CSV output.

Example:

```
gsqcmd rtd-sqlce "SELECT * FROM QuotesYahoo" QuotesYahoo.csv /add=File=QuotesYahoo.csv
```

In this example, the first column File will contain the value: QuotesYahoo.csv.

/Separator=<separator>|tab

Defines the CSV separator.

The default separator is a semicolon.

Use the **Tab** value to specify the tab.

For example:

```
gsqcmd rtd-mysql "SELECT * FROM rtd.quotes_yahoo" quotes_yahoo.csv /separator=,
```

/QuoteChar=<char>

Defines the quote character for string values.

The default value is a double quote. You may disable quotes using the form: /QuoteChar=

/EscapeChar=<char>

Defines the escape character for escaping used quote character in string values.

The default value is a double quote. You may disable escaping using the form: /EscapeChar=

HTML Output Options

`/NoTemplate`

Disables using a template for HTML output.

`/Placeholder=<placeholder>`

Defines the placeholder that replaced with the table data output html.

For example, if an HTML template contains the {sales} placeholder, you may use the option: `/placeholder={Sales}`

The default placeholder is {table}.

`/Template=<HTML template file>`

Defines a template for HTML output.

The template can contain the {table} placeholder that replaced with the table data output html.

By default, the table data are inserted before the `</body>` tag.

The template can also contain the {title} placeholder that replaced with the title option value.

`/Title=<title>`

Defines the value for the {title} placeholder in the HTML template.

Code Generation Options

`/Table=<table>`

Defines the target database table for code generation.

For example: `/table=dbo.Payments`

`/fmt=<format file>`

Defines the format file for code generation.

The file is used as a column name source instead of the CSV file.

The default value of this option is a file name of the CSV file with the .fmt extension.

`/InsertIdentity`

Defines including auto-generated identity columns into INSERT and MERGE SQL codes.

`/mssql | sqlce | mysql | oracle | db2 | nuodb`

Defines target database platform for code generation.

The option is used in modes that have no connection positional parameter.

Exit Codes

Exit Code	Description
0	Success
1	Incomplete command line parameters
2	Wrong command line parameters
3	Exceptions occurred
4	Database server returns an error

gConnectionManager

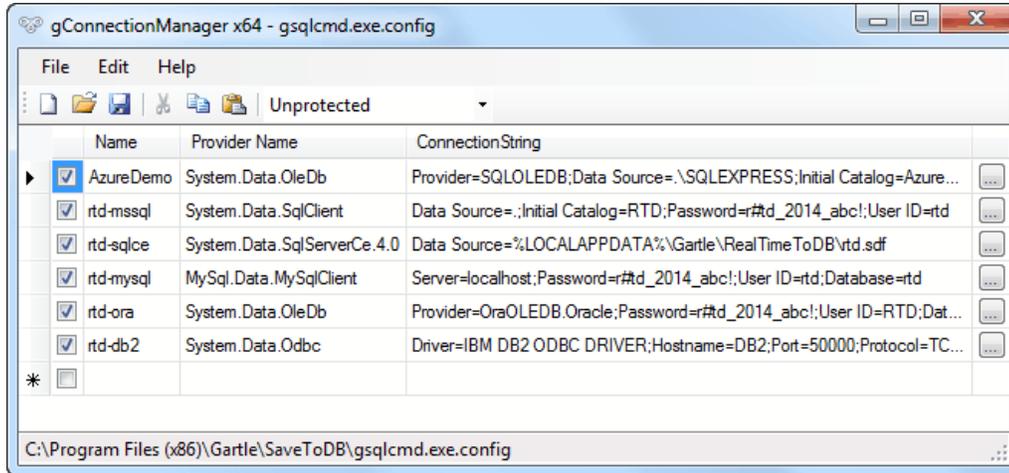
gConnectionManager allows editing connections in application configuration files.

You may use the gsqlcmd commands to start the gConnectionManager with a specific configuration file:

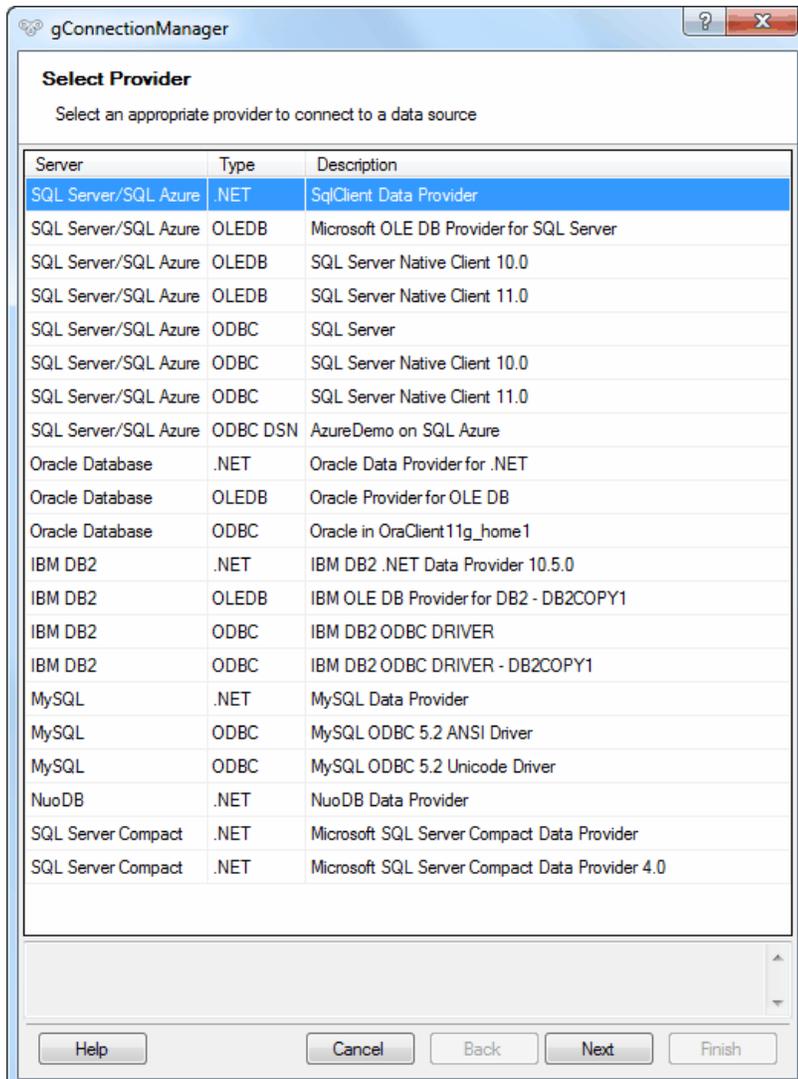
```
gsqlcmd localconfig | userconfig | appconfig
```

Also you may open the desired configuration file using the **File, Open** menu item.

Here is a configuration file example:



Here is an example of selecting a connection provider:



Here is an example of connecting to a database:

The image shows a software dialog box titled "gConnectionManager" with a subtitle "Connect to Database". The main instruction is "Enter the information required to connect to a database".

The dialog is organized into three numbered sections:

- 1. Server name:** A dropdown menu with a single visible option: ".".
- 2. Logon credentials:** Two radio buttons are present:
 - Use Windows Authentication (with a blue "Examples" link to its right)
 - Use the following user name and passwordBelow these are two text input fields:
 - "User name:" containing the text "rtd"
 - "Password:" containing a series of asterisks "*****"
- 3. Database:** A radio button is selected next to the label "Database:", followed by a dropdown menu showing "RTD". Below this is another radio button labeled "File:" followed by an empty text input field and a "Browse" button.

At the bottom of the dialog, there are several buttons:

- "Test Connection" (centered)
- A row of navigation buttons: "Help", "Cancel", "Back", "Next", and "Finish" (which is highlighted with a dashed border).

Configuration File

Connections can be specified in the gsqlcmd.exe.config configuration file.

For example:

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <connectionStrings>
    <add name="AzureDemo" connectionString="Provider=SQLOLEDB;Data Source=.\SQLEXPRESS;Initial Catalog=AzureDemo;Integrated Security=SSPI"
      providerName="System.Data.OleDb" />
    <add name="mssql-rtd" connectionString="Data Source=.;Initial Catalog=RTD;Password=r#td_2014_abc!;User ID=rtd"
      providerName="System.Data.SqlClient" />
    <add name="sqlce-rtd" connectionString="Data Source=%LOCALAPPDATA%\Gartle\RealTimeToDB\rtd.sdf"
      providerName="System.Data.SqlServerCe.4.0" />
    <add name="mysql-rtd" connectionString="Server=localhost;Password=r#td_2014_abc!;User ID=rtd;Database=rtd"
      providerName="MySql.Data.MySqlClient" />
    <add name="ora-rtd" connectionString="Provider=OraOLEDB.Oracle;Password=r#td_2014_abc!;User ID=RTD;Data Source=Oracle/Orcl;PLSQ
LRSet=True"
      providerName="System.Data.OleDb" />
    <add name="db2-rtd" connectionString="Driver=IBM DB2 ODBC DRIVER;Hostname=DB2;Port=50000;Protocol=TCPIP;Database=SAMPLE;Pwd=r#t
d_2014_abc!;UID=RTD;LONGDATACOMPAT=1"
      providerName="System.Data.Odbc" />
    <add name="nuodb-rtd" connectionString="Server=localhost;Password=r#td_2014_abc!;User=RTD;Database=rtd"
      providerName="NuoDb.Data.Client" />
  </connectionStrings>
</configuration>
```

This allows using names like AzureDemo, mssql-rtd, sqlce-rtd, and others as the command line connection parameter.

Use LocalConfig, UserConfig, or AppConfig gsqlcmd modes to edit configurations files in visual mode using [gConnectionManager](#).

The configuration file can be edited also using notepad.exe.

The configuration file can be used also to specify database connection providers if they components are located in the utility directory.

For example, gsqlcmd includes the MySQL .NET provider that is specified in the DbProviderFactories section:

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  ...
  <system.data>
    <DbProviderFactories>
      <remove invariant="MySql.Data.MySqlClient"/>
      <add name="MySQL Data Provider" invariant="MySql.Data.MySqlClient" description=".Net Framework Data Provider for MySQL" type=
"MySql.Data.MySqlClient.MySqlClientFactory, MySql.Data, Version=6.8.3.0, Culture=neutral, PublicKeyToken=c5687fc88969c44d" />
    </DbProviderFactories>
  </system.data>
</configuration>
```

Product Registration

gsqcmd has several editions. See [Edition comparison](#).

The registration is required to register the Personal or Enterprise Edition.

The Express Edition is activated by default, but you may register it for free and has named license.

If you copy the product from the gsqlcmd package to a desired folder, run **RegisterProduct.exe**.

If you install the product in an integrated mode, click the **Register Product** shortcut in the **gsqcmd** group of the **Start** menu.

You may find the gsqlcmd group in one of the parent groups depend on the setup package:

- Gartle\SaveToDB\gsqcmd
- Gartle\RealTimeToDB\gsqcmd

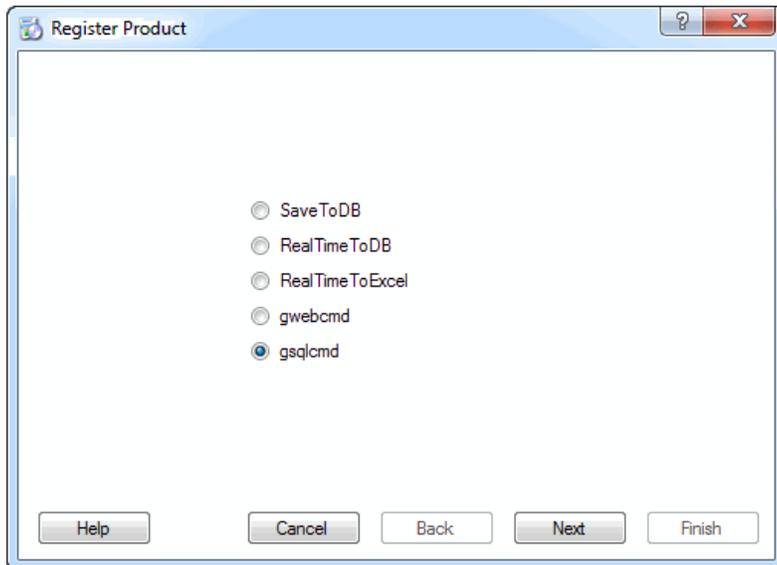
A product code is required to register the Personal or Enterprise edition.

The product code is sent by email after purchasing.

Selecting Product

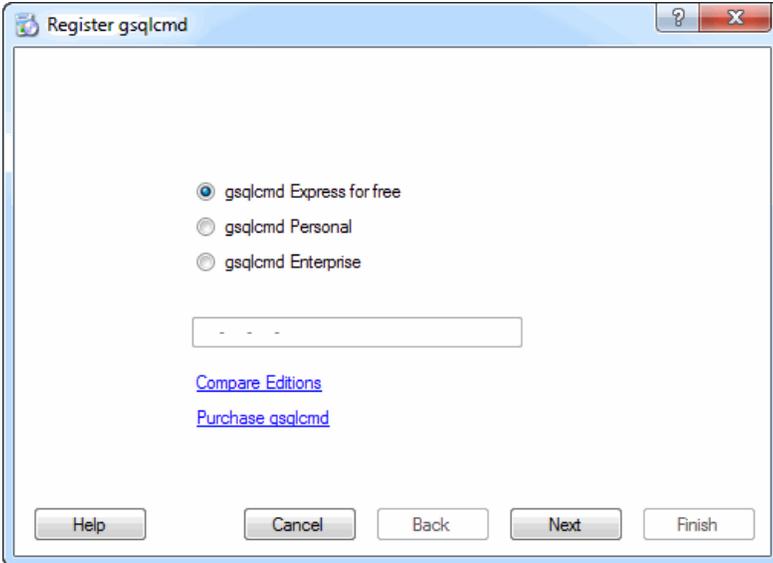
Gartle products have unified registration procedure.

Select the product to register on the start page.



Selecting Edition

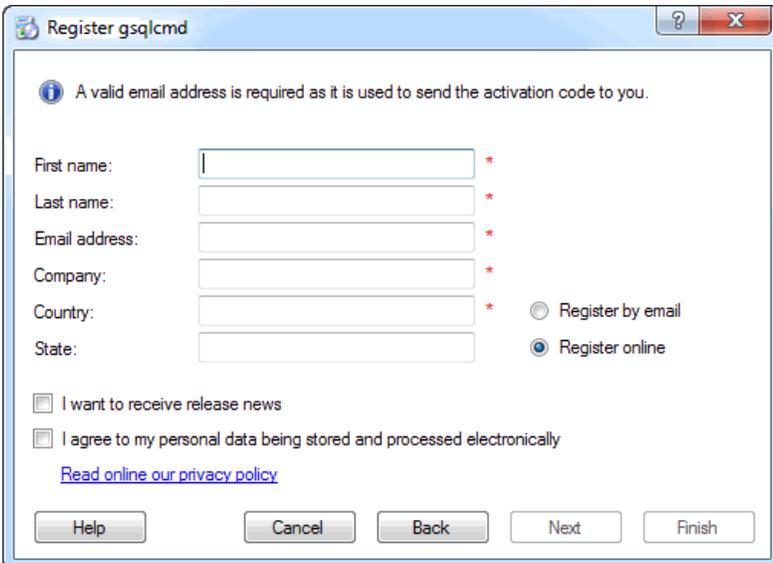
Select an edition and fill in the product code for the Personal, Standard, or Enterprise edition.



The screenshot shows a dialog box titled "Register gsqlcmd". It contains three radio button options for selecting an edition: "gsqlcmd Express for free" (which is selected), "gsqlcmd Personal", and "gsqlcmd Enterprise". Below these options is a text input field containing three dashes "--". There are two blue hyperlinks: "Compare Editions" and "Purchase gsqlcmd". At the bottom of the dialog, there are five buttons: "Help", "Cancel", "Back", "Next", and "Finish".

Licensee Data

Please fill in the registration form carefully.



The screenshot shows the "Register gsqlcmd" dialog box with a message: "A valid email address is required as it is used to send the activation code to you." The form includes several input fields: "First name:", "Last name:", "Email address:", "Company:", "Country:", and "State:". Each of these fields has a red asterisk to its right, indicating they are required. To the right of the "Country:" and "State:" fields are two radio button options: "Register by email" and "Register online", with "Register online" being selected. Below the input fields are two checkboxes: "I want to receive release news" and "I agree to my personal data being stored and processed electronically". A blue hyperlink "Read online our privacy policy" is located below the second checkbox. At the bottom, there are five buttons: "Help", "Cancel", "Back", "Next", and "Finish".

The **Next** button is enabled when all the required fields are filled.

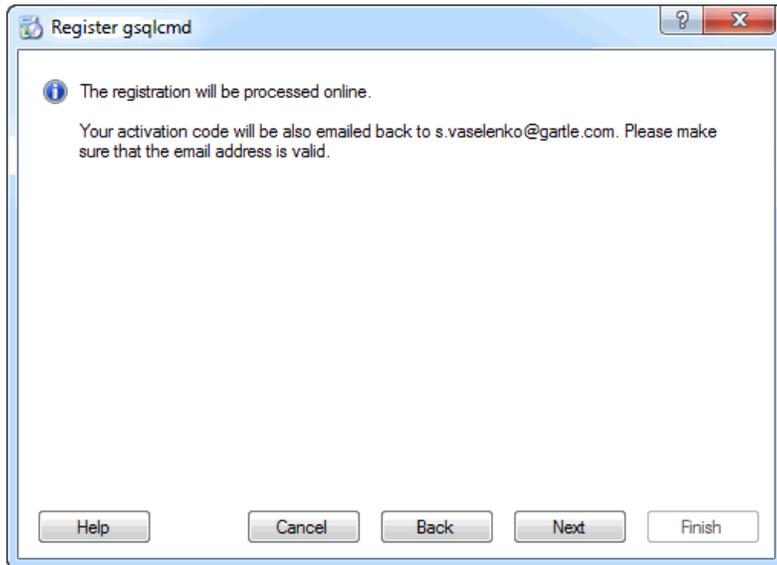
Don't forget to check the required field about the personal data use.

Online Registration

This step allows you to check your email address and to pause before the final step.

If the licensee data is valid, click **Next**.

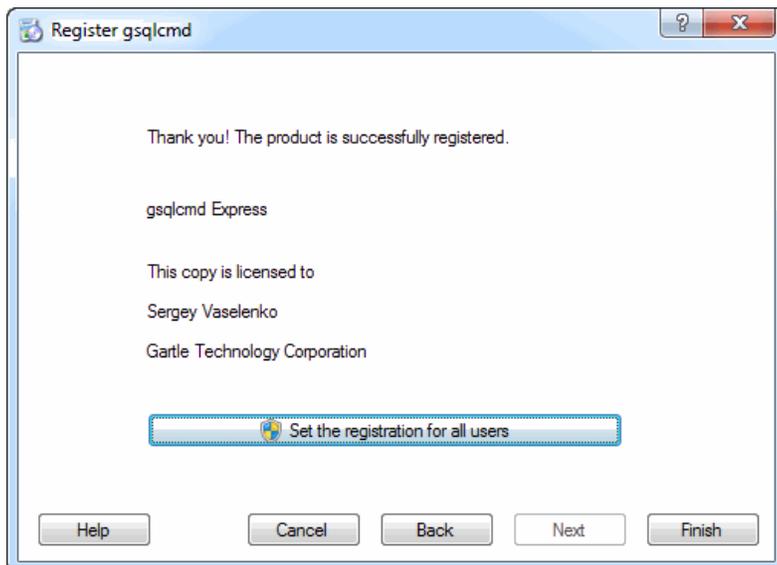
You may return to the previous step using the **Back** button.



After clicking the **Next** button, the gsqlcmd connects to the registration server.

If the connection is successful, the final step screen is shown.

If any error occurred during connection, you may try to register the product later or to register the product by email.

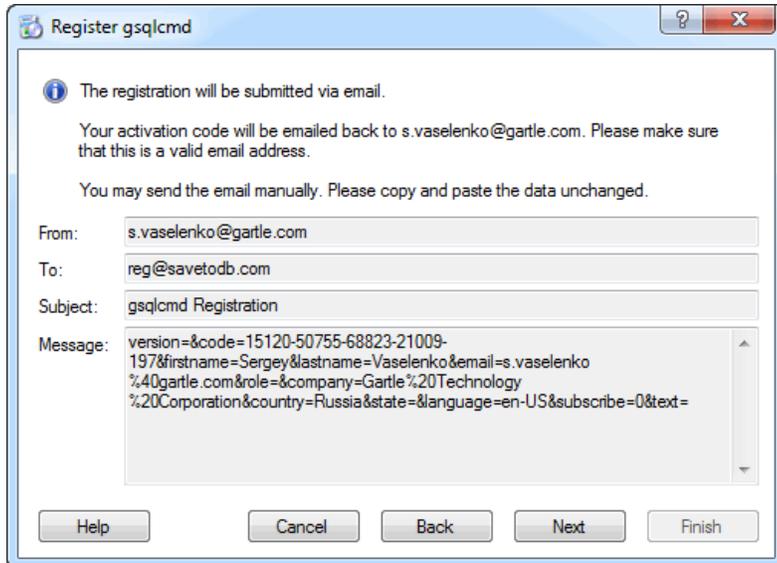


You may set the registration for all users of the computer. This action requires administrator privileges.

Click **Finish**.

Registration by Email

If you choose the **Register by email** option on the **Licensee Data** step, the following screen is displayed:



The screenshot shows a dialog box titled "Register gsqlcmd". It contains the following information:

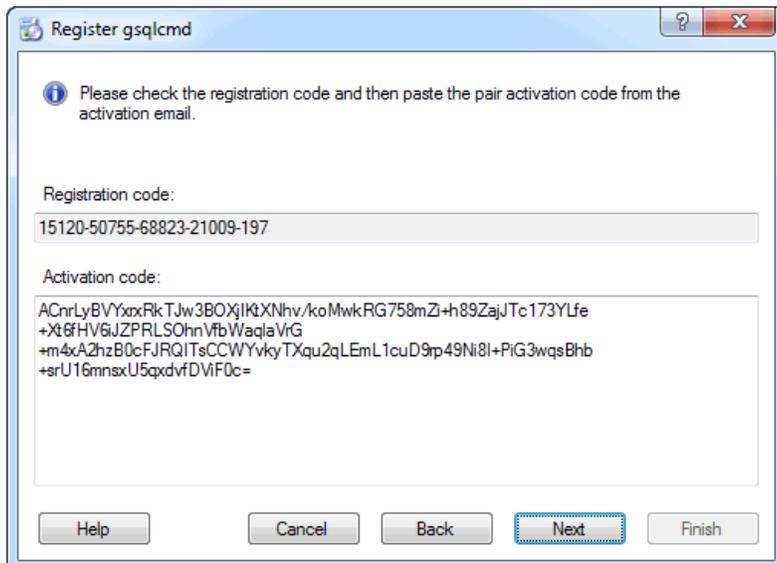
- Information icon: The registration will be submitted via email.
- Text: Your activation code will be emailed back to s.vaselenko@gartle.com. Please make sure that this is a valid email address.
- Text: You may send the email manually. Please copy and paste the data unchanged.
- Form fields:
 - From: s.vaselenko@gartle.com
 - To: reg@savetodb.com
 - Subject: gsqlcmd Registration
 - Message: version=&code=15120-50755-68823-21009-197&firstname=Sergey&lastname=Vaselenko&email=s.vaselenko%40gartle.com&role=&company=Gartle%20Technology%20Corporation&country=Russia&state=&language=en-US&subscribe=0&text=
- Buttons: Help, Cancel, Back, Next, Finish.

The **Next** button starts the default email program, creates a registration email, and activates the next step. Don't forget to send the email.

If starting of the email program has failed, you may create the registration email manually using the shown registration data. Please carefully copy the **To**, **Subject**, and the **Message** fields.

The registration server sends the reply in a couple of seconds; but you may close the dialog box and open it again, on the same step.

Please copy the activation code from the received registration email and paste it to the **Activation code** field.



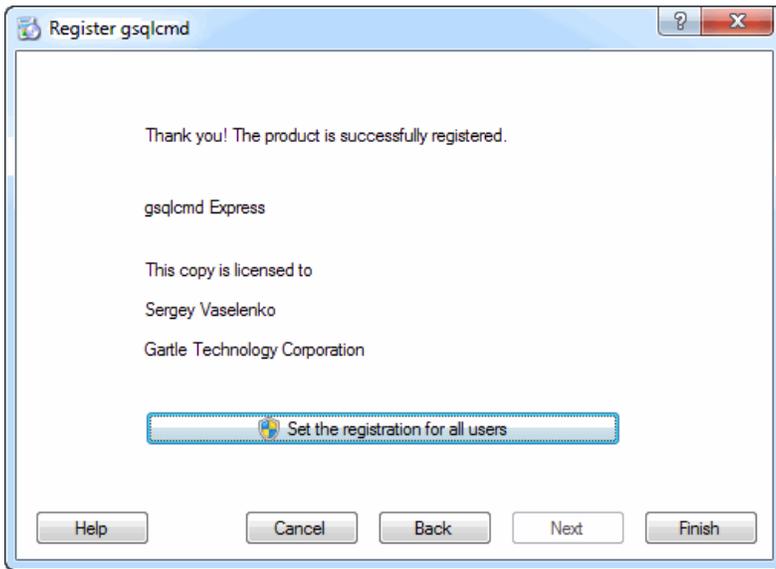
The screenshot shows the same dialog box, but with the following changes:

- Information icon: Please check the registration code and then paste the pair activation code from the activation email.
- Registration code: 15120-50755-68823-21009-197
- Activation code: ACnrLyBVYxxRkTJw3BOxjIKXNlv/koMwkRG758mZi+h89ZajJTc173YLfe+Xi6fHV6jZPRLSOhnVfbWaqjaVrG+m4xA2hzB0cFJRQITsCCWYkyTXqu2qLEmL1cuD9p49Ni8l+PIG3wqsBhb+srU16mnsxU5qxdvfDVf0c=
- Buttons: Help, Cancel, Back, Next (highlighted with a dashed border), Finish.

The **Next** button is enabled when the pasted activation code is valid.

Click **Next** to continue.

The gsqlcmd checks the registration data and confirms the registration.



You may set the registration for all users of the computer. This action requires administrator privileges.

Click **Finish**.

Technical Support

You may download the latest releases at www.savetodb.com.

You may contact us via email support@savetodb.com.

See also [Frequently Asked Questions](#).

Frequently Asked Questions

Why gsqlcmd does not support other database servers if a connection string and an SQL statement are specified in the command line?

gsqlcmd allows executing server specific scripts and works fine with all supported database servers.

Why gsqlcmd uses connection strings instead of connection components (server, database, etc.) in the command line as other command line utilities?

The single parameter as a connection string is handy. You may use a connection string by name. You may pass it in a single environment variable. You may use existing and future features of connection strings.

In addition, you may use gConnectionManager, our companion product, to create connection strings in visual mode using servers, databases, and other connection string components.

What executable, gsqlcmd.exe or gsqlcmd32.exe, should I use?

gsqlcmd uses database providers and drivers of the same bitness.

So, if you have both 64-bit and 32-bit providers installed, you may use any of executables.

But, if you have only 32-bit providers installed on 64-bit Windows, use gsqlcmd32.exe.

gsqlcmd32.exe uses the gsqlcmd32.exe.config configuration file.

My script returns obscure errors. How to debug the script?

gsqlcmd allows executing DDL and DML scripts and supports the most useful extensions of database vendor command line utilities.

If the script is not working, you may try to execute the script using the database vendor command line utility or your favorite IDE.

Also, you may specify the /trace option to get the text of each sent command and corresponding server reply.

If you think, that the standard script is not working, please, send us the script text.