DBGizmo

by Wingenious

License Agreement

VERY IMPORTANT – READ COMPLETELY – READ CAREFULLY

This documentation, and the Product it describes, is copyrighted material.

You may use the Product only in strict accordance with this License Agreement.

Through the act of using the Product you agree to be bound by all the terms and conditions of this License Agreement.

The Product is provided as-is. There is no warranty, expressed or implied. The author shall not be held liable for any damages resulting from use of the Product.

The Product may not be distributed without prior approval from the author. The user must take reasonable precautions to prevent any unauthorized copying of the Product.

The Product must be licensed for use. A license can be issued for a single entity only, such as a person or company. Any association must obtain a special license through negotiation.

The Product is licensed according to the number of people using it. It may not be used by more people than the license fee indicates. Anybody who operates the application or user interface of the Product is considered a user.

The License Agreement is governed by, and construed in accordance with, the laws in effect in the state of Minnesota.

DBGizmo

The features of the DBGizmo application are organized under 17 tabs...

The tab names are Home, Tables, Routines, SQL², SQL³, Notes, Facts, About, and Grid 1 through Grid 9.

The application opens on the Home tab and the upper left portion contains controls for making a server connection and choosing a database to examine. The first seven tabs (Home, Tables, Routines, SQL², SQL³, Notes, Facts) contain the key features. The key features provide numerous ways to analyze your databases and generate powerful SQL code for working with your databases.

The Home tab provides many searching and scripting features, as well as quick access to four lists of 10 (each) query data sets for database analysis.

The Tables tab provides convenient ways to browse table relationships and view column definitions or associated objects.

The Routines tab provides convenient ways to review SQL routine dependencies and the SQL code definitions.

The SQL² tab provides several options to generate handy SQL code for a wide variety of mostly DDL operations.

The SQL³ tab provides two options to generate extremely powerful SQL code for various common DML operations.

The Notes tab provides an editor for database object notes (extended properties), along with options to save them for distribution.

The Facts tab provides a viewer for database facts/settings (database properties). It also includes information about SQL Server Agent jobs.

The About tab contains general information, a control to enable/disable ToolTips, and controls to determine how query data sets in a grid are saved.

The nine Grid tabs each contain an identical set of controls. They are used to view query data sets from the Tables, Routines, and Home tabs. Any Grid tab may contain a data set from any of the 48 different query options. A grid is merely a working location for the data set, similar to a worksheet in a Microsoft Excel spreadsheet file.

The query data sets on Grid tabs can be filtered to temporarily remove certain rows, saved as a CSV file to analyze with another tool (such as Microsoft Excel), saved as an HTML page to view later, or cleared to make the tab available for a different data set.

The DBGizmo application is available in four editions...

The About tab includes some information about the features available with each edition.

DBGizmo Free

The free version of DBGizmo includes tremendous functionality, but it does not include the same functionality as the more advanced editions (L1, L2, L3).

The Home tab is fully functional (16 database analysis query options are available).

The Tables tab is partly functional (view table columns and associated objects).

The Routines tab is partly functional (view SQL code definitions for objects).

The SQL² tab is partly functional (seven of the 12 options are available).

The SQL³ tab is not available.

The Notes tab is fully functional.

The Facts tab is fully functional.

The About tab is fully functional.

The Grid tabs are mostly functional (the Save buttons are not available).

DBGizmo L1

The DBGizmo L1 edition adds full functionality on the Tables tab (table relationships), Routines tab (object dependencies), and Grid tabs (Save buttons), as well as 10 more database analysis query options.

DBGizmo L2

The DBGizmo L2 edition adds full functionality on the SQL² tab (five more options), as well as 14 more database analysis query options.

DBGizmo L3

The DBGizmo L3 edition adds full functionality on the SQL³ tab.

The 48 different query options (data sets) for the nine Grid tabs are:

Home - List 1 - Tables
Home - List 1 - Table Columns
Home - List 1 - SQL Routines
Home - List 1 - SQL Routine Columns
Home - List 1 - SQL Routine Parameters
Home - List 1 - Key Constraints, Primary
Home - List 1 - Key Constraints, Another
Home - List 1 - Key Constraints, Foreign
Home - List 1 - Indexes, Key/Performance
Home - List 1 – Indexes, Summary

Home - List 2 - Table Column Names
Home - List 2 - Table Column Types
Home - List 2 - Table Prominence
Home - List 2 - SQL Routine Parameter Names
Home - List 2 - SQL Routine Parameter Types
Home - List 2 - SQL Routine Prominence
Home - List 2 - Object Name Conflicts
Home - List 2 - Database Schemas
Home - List 2 - Table Partitions
Home - List 2 - Files, Data/Log

Home - List 3 - Object Name Pieces
Home - List 3 - Object Name Problems
Home - List 3 - Questionable Data Types
Home - List 3 - Questionable SQL Routines
Home - List 3 - Questionable SQL References
Home - List 3 - Questionable Parameters
Home - List 3 - Questionable Indexes
Home - List 3 - Index Redundancy
Home - List 3 - Index Column Summary
Home - List 3 - Index Usage, Since Restart

Home - List 4 - Users, Database Access
Home - List 4 - Roles, Object Access
Home - List 4 - Object Permissions
Home - List 4 - Object Synonyms
Home - List 4 - CLR.NET Objects
Home - List 4 - Check Constraints
Home - List 4 - Standard Columns
Home - List 4 - Standard Objects
Home - List 4 - Possible Foreign Keys
Home - List 4 - Table Storage Information

Home - Table/Column Names matching specified criteria
Home - SQL Routine Coding matching specified criteria

Tables - Hierarchy of Parents (ancestors) of a selected Table
Tables - Hierarchy of Children (descendents) of a selected Table

Routines - References To Hierarchy for a selected SQL Routine
Routines - Referenced By Hierarchy for a selected SQL Routine
Routines - Referenced By Hierarchy for a selected Table
Routines - See All (SQL Code Cross References)

The first 40 query options (Lists - 1, 2, 3, 4) do not require any parameters. Most of them are executed immediately upon connecting to a database and the data sets are saved for examination at any time. Some of them are not available until after the Analyze SQL Code Cross References button (on the Home tab) is clicked. The process is executed on demand because the intense analysis can run for several minutes with a complex database.

Any of these 40 data sets (up to nine of them) can be automatically loaded into an empty Grid tab upon connecting to a database. Any of them can be reloaded into an empty Grid tab later. There are controls for such functions in the lower left portion of the Home tab, with 10 query options on each of four tabs (Lists - 1, 2, 3, 4). The lower right portion of the Home tab contains a grid with a summary of the contents of the nine Grid tabs, along with some controls to manage the contents.

The last eight query options are always executed on demand because they depend on user actions or choices. The last four of these are not available until after the Analyze SQL Code Cross References button (on the Home tab) is clicked.

List 1 - Tables

Tables is a list of tables. The list includes layer (hierarchy) indicators, approximate row counts, trigger counts, and primary key indicators.

List 1 - Table Columns

Table Columns is a list of columns by table. The list includes positions within primary key constraints and number of uses in foreign key constraints.

List 1 - SQL Routines

SQL Routines is a list of SQL routines. The list includes layer (hierarchy) indicators.

List 1 - SQL Routine Columns

SQL Routine Columns is a list of columns by SQL routine (the list applies to views and table-valued user-defined functions).

List 1 - SQL Routine Parameters

SQL Routine Parameters is a list of parameters by SQL routine (the list applies to stored procedures and user-defined functions).

List 1 - Key Constraints, Primary

Key Constraints, Primary is a list of primary key constraints.

List 1 - Key Constraints, Another

Key Constraints, Another is a list of UNIQUE constraints (other than primary keys), also known as alternate keys.

List 1 - Key Constraints, Foreign

Key Constraints, Foreign is a list of foreign key constraints. The list includes supporting index names, where they exist.

List 1 - Indexes, Key/Performance

Indexes, Key/Performance is a list of indexes by table.

List 1 - Indexes, Summary

Indexes, Summary is a handy summary of the indexes detailed by the previous option.

List 2 - Table Column Names

Table Column Names is an analysis of column names in tables. For each different column name the query shows how many times the name was used (tables), how many different data types were used, and whether all uses require data. You can double click on a row to see the details.

List 2 - Table Column Types

Table Column Types is an analysis of column data types in tables. For each different data type the query shows how many times the data type was used (columns), how many different column names were used, and how many different tables are involved. You can double click on a row to see the details.

List 2 - Table Prominence

Table Prominence is an analysis of how prominent tables are within the database (only an estimate). For each table the query shows how many columns were defined, how many indexes were created, how many child table relationships exist, how many SQL routine references exist, and an estimated prominence value based on a combination of these.

List 2 - SQL Routine Parameter Names

SQL Routine Parameter Names is an analysis of parameter names in SQL routines. For each different parameter name the query shows how many times the name was used (routines), how many different data types were used, and whether all uses are in/out. You can double click on a row to see the details.

List 2 - SQL Routine Parameter Types

SQL Routine Parameter Types is an analysis of parameter data types in SQL routines. For each different data type the query shows how many times the data type was used (parameters), how many different parameters names were used, and how many different SQL routines are involved. You can double click on a row to see the details.

List 2 - SQL Routine Prominence

SQL Routine Prominence is an analysis of how prominent SQL routines are within the database (only an estimate). For each SQL routine the query shows how many parameters were defined, how many characters of SQL code were written, how many times it's being referenced by another SQL routine, how many times it references another SQL routine, and an estimated prominence value based on a combination of these.

List 2 - Object Name Conflicts

Object Name Conflicts is a list of the database objects where the same name is used in different schemas.

List 2 - Database Schemas

Database Schemas is a list of the schemas in the database and a summary of the objects contained in each one.

List 2 - Table Partitions

Table Partitions is a list of the partitions for tables configured as partitioned in SQL Server 2005 and newer.

List 2 - Files, Data/Log

Files, Data/Log is a list of the Windows operating system files for the database.

List 3 - Object Name Pieces

Object Name Pieces is a list of words or abbreviations found within object names. The name segments often represent business terminology. This analysis makes it easy to notice inconsistent use of terms.

List 3 - Object Name Problems

Object Name Problems is a list of the objects which have names containing "invalid" characters. Object names with "invalid" characters (such as spaces) require qualifiers when used in SQL code.

List 3 - Questionable Data Types

Questionable Data Types is a list of table columns defined with unusual or obsolete data types. The issue found is included with each table column.

List 3 - Questionable SQL Routines

Questionable SQL Routines is a list of SQL routines which could have potential behavioral problems. The issues found are included with each SQL routine.

List 3 - Questionable SQL References

Questionable SQL References is a list of SQL code references which could have potential performance problems. The issue found is included with each SQL reference.

List 3 - Questionable Parameters

Questionable Parameters is a list of SQL routine parameters with names corresponding to table column names where the data types do not match.

List 3 - Questionable Indexes

Questionable Indexes is a list of indexes which could have potential performance problems. The issue found is included with each index.

List 3 - Index Redundancy

Index Redundancy is an analysis of columns used in index keys. The query shows pairs of indexes where the two lists of index keys appear to be redundant. Redundancy is detected when the first N (one or more) columns are the same.

List 3 - Index Column Summary

Index Column Summary is an analysis of columns used in index keys. For each different column the query shows how many times it was used in an index key (indexes) and how many times it was placed at various positions (1, 2, 3, 4, 5+) in an index key.

List 3 - Index Usage, Since Restart

Index Usage, Since Restart is a list of indexes with usage statistics. The list shows index usage information since the instance was last restarted.

List 4 - Users, Database Access

Users, Database Access is a list of database users, with their fixed database roles. You can double click on a row to see the explicit object/column permissions for a user.

List 4 - Roles, Object Access

Roles, Object Access is a list of database roles, members of each role, and a summary of explicit object permissions.

List 4 - Object Permissions

Object Permissions is a list of explicit object/column permissions by user/role.

List 4 - Object Synonyms

Object Synonyms is a list of synonyms defined in the database.

List 4 - CLR.NET Objects

CLR.NET Objects is a list of CLR.NET objects.

List 4 - Check Constraints

Check Constraints is a list of check constraints.

List 4 - Standard Columns

Standard Columns is a list of tables with an indication of which standard columns (as defined on the SQL² tab) are present for each table.

List 4 - Standard Objects

Standard Objects is a list of tables with an indication of which standard objects (such as CRUD routines) have been generated for each table.

List 4 - Possible Foreign Keys

Possible Foreign Keys is a list of table pairs where their columns suggest a foreign key constraint might be missing.

List 4 - Table Storage Information

Table Storage Information is a list of tables with some physical data storage details.

Table/Column Names matching specified criteria

SQL Routine Coding matching specified criteria

These two query options require parameters and they are executed using controls in the upper right portion of the Home tab. The Find buttons search for the specified string of characters (pattern matching characters are supported) among Table/Column Names or SQL Routine Coding. The searching of SQL code is especially handy because the data set includes SQL Routine names, line numbers, and matching lines of SQL code. The SQL Routines check box causes object/column/parameter names of SQL Routines to be included in a Table/Column Names search. The No Comments check box causes comments in SQL code to be ignored during a SQL Routine Coding search.

The pattern matching functionality in DBGizmo is very similar to the Like operators in SQL and VB. The pattern matching characters of both languages are recognized and supported. It's a very powerful filtering mechanism.

Hierarchy of Parents (ancestors) of a selected Table

Hierarchy of Children (descendents) of a selected Table

These two query options require parameters and they are executed using controls in the middle of the Tables tab. The Get Hierarchy of Parents button starts with the selected Table and lists every generation above the selected Table (ancestors) in the tree structure of the database. The Get Hierarchy of Children button starts with the selected Table and lists every generation below the selected Table (descendents) in the tree structure of the database.

References To Hierarchy for a selected SQL Routine

Referenced By Hierarchy for a selected SQL Routine

Referenced By Hierarchy for a selected Table

See All (SQL Code Cross References)

The three Reference... query options require parameters and they are executed using controls in the middle right portion of the Routines tab.

Each button starts with the selected object from a list box.

The Selected SQL Routine comes from the top center list box.

The Selected Table comes from the bottom center list box.

The References To option finds any object that the selected SQL Routine references, any object that the referencing object references, and so on. The data set is a layered list of dependencies for the selected SQL Routine, which is very useful for assessing the scope of the existing behavior of the SQL Routine.

The Referenced By options find any SQL Routine that references the selected object, any SQL Routine that references the referencing SQL Routine, and so on. Each data set is a layered list of dependent SQL Routines, which is very useful for assessing the impact of a proposed change to the selected object.

The See All option lists all SQL Code Cross References, as found by the Analyze SQL Code Cross References button on the Home tab.

The features of the DBGizmo application are organized under 17 tabs...

Tab > Home

In addition to the functionality mentioned above, the middle right portion of the Home tab contains controls for generating SQL script for existing SQL Routines. The SQL Routines to be placed in the SQL script file(s) can be limited by type and/or by name. SQL Routines can be included (LIKE) by name and/or excluded (NOT LIKE) by name, and SQL (or VB) pattern matching characters are supported. The Save SQL Code File button always creates one main file with all the selected SQL Routines, but it can optionally create a separate file for each SQL Routine as well. The separate files are convenient for use with a source code control system. The main file with all the selected SQL Routines can be limited to DROP statements only, to be used for removal of the selected objects. There is an option to include existing permissions with each SQL Routine. There is an option to remove any/all comments from each SQL Routine.

The middle left edge of the Home tab contains a large graphical button. The button opens a maximized dialog box with numerous controls, including an item for each table in the database. The items are organized into layers representing generations. The layers are oriented vertically. Tables without parents are listed in a column down the left side (Layer 0). Tables with the most ancestral generations are listed in a column down the right side (Layer N). A table can be singled out by clicking on an item or selecting from the list. The item for the selected table is red. The items for parents (left) and children (right) of the selected table are orange. The items for ancestors (left) and descendents (right) of the selected table are gold. The items for tables unrelated to the selected table are yellow. There's a list to choose how many generation lines should be drawn. The choices include 0 for no generation lines, 1 adds parents and children, 2+ adds ancestors and descendents, All adds every relationship in the database. There's a check box to show/hide a column count for each table.

There are two small buttons immediately above the query option tabs. The # button puts a list of all 40 query options, along with current row counts, into a Grid tab. The Grid tab can be used as a menu for viewing the data sets. The Save All button generates an HTML page and a CSV file for each of the 40 query options. It also creates an index (home) page for the 40 separate HTML pages.

Tab > Tables

The Tables tab provides an integrated set of controls for browsing database architecture and examining the relationships between tables. The top center list box contains all Tables in the database. When you double click on an item it becomes the selected Table and the other five areas of the tab are populated accordingly. The top left list box contains parents of the selected Table, and double clicking on a parent makes it the new selected Table. The top right list box contains children of the selected Table, and double clicking on a child makes it the new selected Table.

The middle left grid contains the column(s) involved in the relationship between the selected Table and the currently highlighted parent (click in the Reference name area to copy a SELECT statement to the clipboard). The middle right grid contains the column(s) involved in the relationship between the selected Table and the currently highlighted child (click in the Reference name area to copy a SELECT statement to the clipboard). The bottom part of the tab is a grid which contains either the columns of the selected Table or the objects associated with the selected Table (such as constraints, indexes, and triggers). You can alternate between the two displays by clicking on the label immediately above the grid. The middle part of the tab contains six buttons.

The Get Hierarchy of Parents button is described above.

The Get Hierarchy of Children button is described above.

The Get SQL Code References button shows (on a Grid tab) all the SQL Routines that reference the selected Table.

The Generate SELECT for Parents button copies a SELECT statement to the clipboard. The statement includes all parents of the selected Table.

The Generate SELECT for Children button copies a SELECT statement to the clipboard. The statement includes all children of the selected Table.

The Generate SQL Code Snippets button copies several snippets of SQL code to the clipboard. The snippets are SQL statements related to the selected Table.

Tab > Routines

The Routines tab provides an integrated set of controls for reviewing SQL Routine cross references and analyzing dependencies between objects. The top center list box contains all SQL Routines in the database. When you double click on an item it becomes the selected SQL Routine and four other areas of the tab are populated accordingly. The top left list box contains SQL Routines the selected SQL Routine is referenced by, and double clicking on an item makes it the new selected SQL Routine. The top right list box contains SQL Routines the selected SQL Routine references, and double clicking on an item makes it the new selected SQL Routine. The middle right list box contains Tables the selected SQL Routine references, and double clicking on an item makes it the new selected Table.

The text box in the middle left of the tab displays the SQL code for the selected SQL Routine and the Copy SQL Code button copies the SQL code to the clipboard. The bottom center list box contains all Tables in the database. The bottom left list box contains SQL Routines the selected Table is referenced by, and double clicking on an item makes it the new selected SQL Routine. The bottom right list box contains the columns of the selected Table. The three large buttons, and the one small button, in the middle right portion of the tab are described above. The Save SQL Code File button behaves like the button of the same name on the Home tab. The default selection of SQL Routines to be scripted is determined by the previous action.

Tab > SQL²

The SQL² tab provides 12 features for generating SQL code (or application code) to perform a variety of functions. All of the features use the selection of tables from the list box on the left side of the tab. Tables can be selected using standard methods, or double clicking on a table opens a dialog box with three selection options. The dialog box allows selecting tables by name (pattern matching characters are supported), selecting all tables without parents (such as lookup tables), or selecting all tables descendent from the selected table. The Create Date Column text box and the Modify Date Column text box can be used to specify standard column names, if applicable in your database. The SQL Routine Prefixes controls can be used to specify standard prefixes for the names of certain objects (often referred to as CRUD routines), if appropriate for your database. These specifications are used by only a couple of the SQL code generation features.

There are six different kinds of SQL Routines that can be generated for each table (one of the 12 features). The INSERT trigger populates the Create Date column (if present) with a server date/time during an insert. The UPDATE trigger populates the Modify Date column (if present) with a server date/time during an update. The SELECT stored procedure selects all columns for a specific row based on a primary key value. The DELETE stored procedure deletes a specific row based on a primary key value. The INSERT stored procedure inserts a new row. The UPDATE stored procedure updates all columns for a specific row based on a primary key value. The options for SELECT and DELETE can result in more than one stored procedure each. A separate stored procedure is generated for each foreign key constraint. These extra stored procedures accept a foreign key value and they SELECT or DELETE rows based on the foreign key value.

Generate SQL Code to create basic table structures

This button produces a SQL file containing SQL code to create the basic table structures for the selected tables. The script includes column definitions, table/column constraints, and various indexes.

Generate SQL Code to drop/add indexes for all foreign key constraints

This button produces a SQL file containing SQL code to create indexes in support of all foreign key constraints for the selected tables. The indexing on foreign keys can be very beneficial for query performance.

Generate SQL Code to drop/add foreign key constraints while doing TRUNCATE TABLE

This button produces a SQL file containing SQL code to drop foreign key constraints for the selected tables, perform TRUNCATE TABLE on the selected tables, and recreate foreign key constraints for the selected tables.

Generate SQL Code to get exact row counts and status information for IDENTITY columns

This button produces a SQL file containing SQL code to return exact row counts and status information for IDENTITY columns for the selected tables.

Generate SQL Code to replace key constraints and related indexes using generated naming

This button produces a SQL file containing SQL code to drop existing primary key constraints, foreign key constraints, unique constraints, and indexes. It recreates objects with identical functionality using consistent generated naming.

Generate a FROM clause surrounding the main table indicated above including all parents/children from the list

This button generates SQL code and places it on the clipboard. The SQL code is a FROM clause that starts with the table selected in the list immediately above the button. The SQL code includes a join to every parent/child table among the selected tables.

Generate a set of basic data access class modules with a basic database connection module in VB.NET or C#

This button produces a small code module for database connection and a class module for each of the selected tables. The database connection module includes general routines to support each class module. The class modules include a property for each column, general properties, general methods, and methods specific to foreign keys (if any). The methods assume the generated stored procedures are present in your database.

The generated VB.NET or C# files can be used as the foundation for a data access layer in a WinForms application or a web application (an example of each is available).

To generate VB.NET… specify a **vb** file name extension when saving

To generate C#… specify a **cs** file name extension when saving

The Save operation creates a database connection module as the specified file. It creates a folder with the database name and it creates class modules inside the folder, one for each table. If all tables are selected then an additional class module is created. The file is named after the database and it contains a method for each custom stored procedure. This code is intended only as a copy/paste starting point for actual application code.

The Save operation creates three template files. If these files are customized and relocated with the DBGizmo application file then they will be used for subsequent code generation.

`DBAccess.*` is a template for the database connection module.

`DBModule.*` is a template for the class modules (one for each table).

`DBSPCore.*` is a template for the additional class module (custom stored procedures).

Generate SQL Code to create selected basic SQL routines

This button produces a SQL file containing SQL code to create up to six kinds of objects for each of the selected tables. The objects are commonly referred to as CRUD routines. The different kinds of objects are described above. The generated stored procedures must be present in your database in order to use generated VB.NET or C# files.

Generate SQL Code to create primary keys based on column names

This button produces a SQL file containing SQL code to create a primary key constraint for each of the selected tables. A single column constraint is created if/when there's a column with a name based on the table name.

Generate SQL Code to create foreign keys based on column name matching

This button produces a SQL file containing SQL code to create foreign key constraints for each of the selected tables. A constraint is created if/when there's a column (or a group of columns) with a name (or names) matching the primary key of another table.

Generate SQL Code to create tables/triggers for auditing using Method 1 (affected primary keys)

This button produces a SQL file containing SQL code to create an audit table and three triggers for each of the selected tables. These generated objects assume the presence of an AUDIT database, but the database name can be easily changed with a global search/replace in the SQL file.

Method 1 auditing consists of: A parallel table is created containing only the primary key column(s) and two audit processing columns. A trigger is created on the main table. The trigger adds a row to the parallel table with every INSERT, UPDATE, or DELETE operation on the main table. The parallel table can be used to guide the processing of affected rows in the main table.

Generate SQL Code to create tables/triggers for auditing using Method 2 (every version every row)

This button produces a SQL file containing SQL code to create an audit table and two triggers for each of the selected tables. These generated objects assume the presence of an AUDIT database, but the database name can be easily changed with a global search/replace in the SQL file.

Method 2 auditing consists of: A parallel table is created containing all columns from the main table and several audit processing columns. A trigger is created on the main table. The trigger adds a row to the parallel table with every UPDATE or DELETE operation. The parallel table contains every previous version of every row and the main table contains the usual current version of every row.

Tab > SQL³

The SQL³ tab provides two very powerful features for generating SQL code to reconcile row differences in two databases with the same table structure. The top of the tab is for specifying the two databases. The Source Database is the database to be copied from. The Modify Database is the database to be copied into. The left side of the tab is for processing differences across all rows in a selected set of tables. The right side of the tab is for processing differences across a set of related rows in related tables.

The left side of the tab starts by selecting the table(s) to be included from a list box. Tables can be selected using standard methods, or double clicking on a table opens a dialog box with three selection options. The dialog box allows selecting tables by name (pattern matching characters are supported), selecting all tables without parents (such as lookup tables), or selecting all tables descendent from the selected table. The check boxes on the left side of the tab determine which kind(s) of SQL code will be generated. The SELECT check box returns all row differences in the selected tables. The INSERT check box inserts rows into the Modify database from the Source database if the rows do not already exist. The UPDATE check box updates rows in the Modify database if the rows are different from corresponding rows in the Source database. The DELETE check box deletes rows from the Modify database if corresponding rows do not exist in the Source database. The button generates SQL code according to the choices made above.

The right side of the tab starts by selecting the table containing the highest order row(s). The WHERE clause entered below is used with this table to identify one or more rows as the ancestor(s) of related rows in related tables. For example, the selected table might be Customer and the WHERE clause might specify Customer 1, causing all Order rows under that Customer row to be included, causing all OrderLine rows under those Order rows to be included, causing all OrderLineProduct rows under those OrderLine rows to be included, and so on. The check boxes on the right side of the tab determine which kind(s) of SQL code will be generated. The options are a bit more precise than those described above, but they are very similar. There is no option to DELETE existing rows from the destination because there is no way to know those rows do not belong to another branch of data. If such an operation is necessary then the branch can be removed from the destination and copied (again) from the source. The button generates SQL code according to the choices made above.

Tab > Notes

The Notes tab contains an editor for the extended properties, referred to as Notes in the application, of various database objects. Some objects are listed with hints to provide more information. Changes can be made in the Note..NEW column of the data grid. The Clear button will clear all existing Notes. The Copy button will copy hints as new Notes.

The Save SQL Code File button creates one file with SQL statements to implement all modifications to Notes made within the data grid.

The Save CSV / Excel button creates one CSV (Comma Separated Values) file containing all existing Notes.

The Save HTML Page button creates one HTML (HyperText Markup Language) file containing all existing Notes.

Tab > Facts

The Facts tab contains a viewer to see/compare many important settings for all available databases. The information includes 20+ database properties, referred to as Facts in the application, shown in two groups. It also includes all scheduled SQL Server Agent jobs.

Show Facts (database property) Group 1...
This option shows several critical database settings (in words), as well as backup dates, for all available databases on the server.

Show Facts (database property) Group 2...
This option shows about 20 miscellaneous database settings, as either on (Y) or off (N), for all available databases on the server.

Show SQL Server Agent (scheduled) Jobs...
This option shows information about all scheduled SQL Server Agent jobs on the server. You can double click on a row to see the job execution history.

The Refresh button refreshes the information to see current values.

The Save CSV / Excel button creates one CSV (Comma Separated Values) file containing information from the currently displayed grid.

The Save HTML Page button creates one HTML (HyperText Markup Language) file containing information from the currently displayed grid.

Tab > About

The About tab contains general information, a control to enable/disable ToolTips, and controls to determine how query data sets in a grid are saved. The method for saving query data sets (grids) can be temporarily switched by holding the Alt key while clicking the Save button on a Grid tab.

Tab > Grid 1-9

The nine Grid tabs each contain an identical set of controls. They are used to view query data sets from the Tables, Routines, and Home tabs. Any Grid tab may contain a data set from any of the 48 different query options. A grid is merely a working location for the data set, similar to a worksheet in a Microsoft Excel spreadsheet file.

The query data sets on Grid tabs can be filtered to temporarily remove certain rows, saved as a CSV file to analyze with another tool (such as Microsoft Excel), saved as an HTML page to view later, or cleared to make the tab available for a different data set.

The Grid row filtering feature supports pattern matching (see above) to include/exclude rows based on values in certain columns.

Some query data sets contain a Table, Routine, or Object column. When a Grid has such a column it supports an additional feature. You can double click on a row and DBGizmo will use the Table, Routine, or Object name to bring you to more information. A Table, or a table Object, will cause a switch to the Tables tab with a new selected Table. A Routine, or a routine Object, will cause a switch to the Routines tab with a new selected SQL Routine. You can jump back to the Grid tab by using a control key combination.

DBGizmo supports the following handy control key combinations to move between tabs in the GUI:

Ctrl-H	move to Home
Ctrl-T	move to Tables
Ctrl-R	move to Routines
Ctrl-B	move to previous tab
Ctrl-G	move to previous tab
Ctrl-D	move to previous tab
Ctrl-1	move to Grid 1
Ctrl-2	move to Grid 2
Ctrl-3	move to Grid 3
Ctrl-4	move to Grid 4
Ctrl-5	move to Grid 5
Ctrl-6	move to Grid 6
Ctrl-7	move to Grid 7
Ctrl-8	move to Grid 8
Ctrl-9	move to Grid 9

Comments about DBGizmo

DBGizmo was designed to support many different styles of database architecture, but it may not be compatible with every database. Certain practices are problematic. As an example, DBGizmo simply ignores the circular parent/child relationship of a self-referencing table.

Several of the features in the application assume you have already implemented complete Declarative Referential Integrity (DRI), which means every table must have a primary key constraint and every relationship between two tables must be defined and enforced with a foreign key constraint.

Several of the features in the application do not support copying/manipulating columns of certain data types (mostly obsolete). The contents of a column with a data type of text, ntext, image, or sql_variant are simply ignored during processing and the destination column must be allowed to contain null.

DBGizmo Extras

The DBGizmo Extras archive file contains examples of some of the many code generation features. It also contains two example Visual Studio 2005 applications, one is a Windows Forms application and the other is a web application, to demonstrate how to implement some of the generated code.

SQL1.sql

SQL2.sql

SQL3.sql

SQL4.sql

These files contain SQL code, mostly generated by DBGizmo, to create example objects in an example database. Create a new database then execute these four SQL files in sequence against the new database. The objects created in the new database are for the purpose of exploring DBGizmo features. The new database is for demonstration ONLY. It's not intended to be suitable for any real business.

Snippet.sql

This file contains SQL code generated by DBGizmo using the "Generate SQL Code" button on the Tables tab.

All Rows.sql

This file contains SQL code generated by DBGizmo using the "Generate SQL Code to reconcile any row differences" button on the SQL³ tab.

Method1.sql

This file contains SQL code generated by DBGizmo using the "Generate SQL Code to create tables/triggers for auditing using Method 1" button on the SQL² tab. The code assumes an AUDIT database has been created.

Method2.sql

This file contains SQL code generated by DBGizmo using the "Generate SQL Code to create tables/triggers for auditing using Method 2" button on the SQL² tab. The code assumes an AUDIT database has been created.

WinForm

This folder contains a working Visual Studio 2005 solution to demonstrate several pieces of DBGizmo generated code. The WinForm project uses a generated VB.NET class for each database table, along with a generated connection module, for database access. The generated class files reference DBGizmo generated stored procedures in the database. The database is assumed to be the example database mentioned above. The server and database are identified in the VB code and they must be set according to your environment.

WebForm

This folder contains a working Visual Studio 2005 solution to demonstrate several pieces of DBGizmo generated code. The web application uses a generated VB.NET class for each database table, along with a generated connection module, for database access. The generated class files reference DBGizmo generated stored procedures in the database. The database is assumed to be the example database mentioned above. The server and database are identified in the VB code and they must be set according to your environment.