

Contents

| | | |
|----|---|----|
| 1 | Introducing GS-Base..... | 4 |
| 2 | Creating new databases | 5 |
| 3 | Using table, form and binary views | 7 |
| 4 | GS-Base zip database file format..... | 8 |
| 5 | Managing database fields and tables..... | 10 |
| 6 | Editing field contents | 11 |
| 7 | Entering Unicode characters | 14 |
| 8 | Inserting images/files vs inserting links to images/files | 14 |
| 9 | Loading images/files form entire folders | 16 |
| 10 | Entering formulas | 16 |
| 11 | 'Cross-table' calculation functions..... | 19 |
| 12 | Using the makeUnique() and isUnique() functions..... | 35 |
| 13 | Using objectStats() functions to obtain file/image data | 35 |
| 14 | Default field values | 37 |
| 15 | Inserting series..... | 37 |
| 16 | Entering data: Drop-down lists | 41 |
| 17 | Generating passwords | 44 |
| 18 | Encrypting field contents | 44 |
| 19 | Using the Copy-With-Options command..... | 45 |
| 20 | Using the Increment, Decrement and Operations commands | 48 |
| 21 | Dual Views | 49 |
| 22 | Linking tables in dual views by the 1-n relation | 50 |
| 23 | Merging and joining tables..... | 50 |
| 24 | Spell-checking and adding Hunspell dictionaries | 52 |
| 25 | Using Pivot Tables | 56 |
| 26 | Pivot Table Functions..... | 60 |
| 27 | Creating Pivot Table Reports | 61 |
| 28 | Selecting fields and records, scrolling, shortcut keys | 63 |

| | | |
|------|---|-----|
| 29 | Formatting | 64 |
| 30 | Formatting - Hyperlinks | 67 |
| 31 | Formatting – Images..... | 68 |
| 32 | Formatting: Charts in Record Fields | 69 |
| 33 | Searching / Filtering Records | 70 |
| 34 | Searching / Saving Current Filters As Named Search Keys | 72 |
| 35 | Searching - Duplicates | 73 |
| 36 | Searching - Unique field values | 74 |
| 37 | Searching - Quartiles, mean values | 75 |
| 38 | Searching - Random records | 76 |
| 39 | Searching - Selected records | 76 |
| 40 | Full Text Searches and Replacing..... | 77 |
| 41 | Find-As-You-Type and Find-Similar searching options..... | 80 |
| 42 | Sorting | 80 |
| 43 | Sending e-mail messages | 81 |
| 44 | Verifying URLs entered in database fields | 84 |
| 45 | Printing tables, forms and labels | 85 |
| 46 | Password protection and encryption | 87 |
| 47 | Saving files: PDF files..... | 88 |
| 48 | Opening and saving text files | 89 |
| 49 | Opening and saving MySQL *.sql files | 91 |
| 50 | Opening and saving html files..... | 92 |
| 51 | Opening and saving dBase/FoxPro/Clipper files | 93 |
| 52 | Opening and saving Excel workbooks | 94 |
| 53 | Settings | 97 |
| 54 | Managing the list of "recently-used" files..... | 102 |
| 55 | JScript and VBscript scripting..... | 103 |
| 55.1 | Creating and saving a new database | 104 |
| 55.2 | Adding and removing records, updating calculated fields..... | 106 |
| 55.3 | Adding, removing and renaming tables and folders | 107 |
| 55.4 | Importing tables from GS-Base databases, text files and Excel workbooks | 108 |
| 55.5 | Merging records from all text and Excel files in a folder | 109 |
| 55.6 | Formatting fields..... | 110 |
| 55.7 | Setting column/field widths and row heights..... | 111 |

| | | |
|-------|--|-----|
| 55.8 | Browsing the folders/tables tree..... | 112 |
| 55.9 | Searching and sorting..... | 113 |
| 55.10 | Predefined searching..... | 115 |
| 55.11 | File password protection..... | 115 |
| 55.12 | Enabling sharing databases and text, Excel and other files | 116 |
| 55.13 | Opening and saving text files..... | 116 |
| 55.14 | Error handling | 118 |
| 55.15 | File/database opening functions..... | 119 |
| 55.16 | Saving recordsets | 121 |
| 55.17 | Saving databases | 122 |
| 55.18 | Saving databases as new files | 122 |
| 55.19 | Importing tables databases, text and Excel files | 123 |
| 55.20 | Performing JOIN operations for tables in the same database file..... | 123 |
| 55.21 | Merging/adding records from other files | 123 |
| 55.22 | Switching the active table | 125 |
| 55.23 | Record counters | 125 |
| 55.24 | Searching..... | 125 |
| 55.25 | Adding, modifying and removing fields..... | 126 |
| 55.26 | Browsing the database folder and tables tree structure..... | 128 |
| 55.27 | Editing records | 129 |
| 55.28 | Inserting series | 129 |
| 55.29 | Changing column widths and row heights..... | 130 |
| 55.30 | Formatting fields..... | 130 |
| 55.31 | Setting record flags | 133 |
| 55.32 | Setting database passwords for GS-Base *.gsb/*.zip files | 133 |
| 55.33 | File information and access | 133 |
| 55.34 | Updating all calculation formulas..... | 134 |
| 55.35 | Input/output methods | 134 |
| 55.36 | Window methods | 135 |
| 55.37 | Obtaining the last error details | 135 |
| 56 | GS-Base - methods and properties | 135 |
| 57 | Support | 146 |

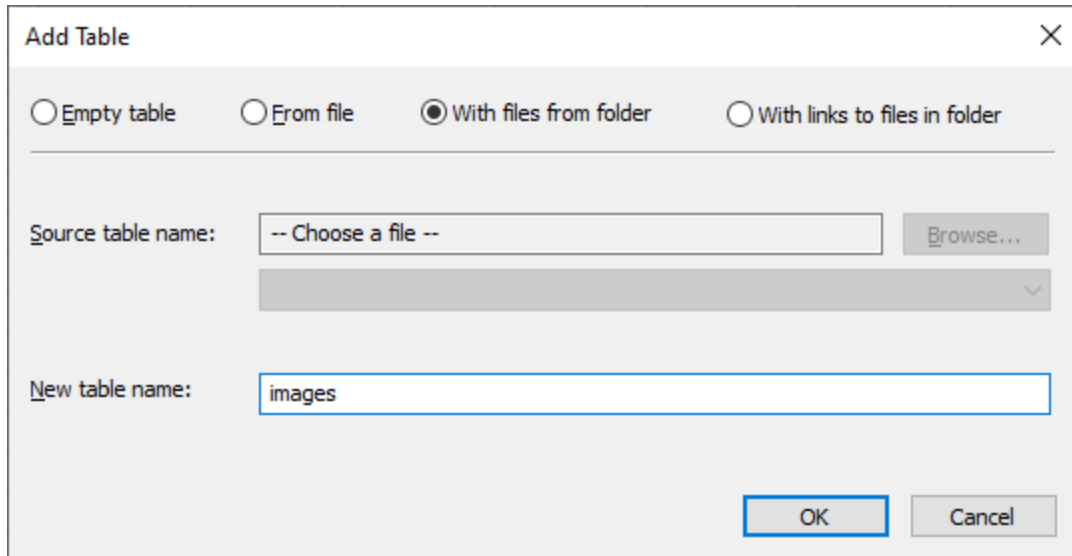
1 Introducing GS-Base

- Up to 256 million records x 16,484 fields in one table; any number of tables in one file.
- Managing any type of data: numbers, plain text fields, long formatted text documents, images and any files.
- Around 300 standard built-in formulas that can be used to automatically calculate field values, create filtering expressions or to create links between tables.
- Extensive searching capabilities - searching for duplicate records or field values, full-text searches etc.; multiple sort keys.
- Pivot tables capable of processing very large amounts of data.
- Standard editing and formatting tools and options; predefined numeric styles and user-defined styles.
- Printing tables, forms, letters and any labels.
- Sending personalized e-mail messages with customized attachments.
- Strong password protection and encryption.
- Saving tables to PDF.
- Safe zip file format - data can be browse even without the program.
- Clean installation (no registry entries are required).
- Importing, exporting, editing splitting and merging csv, text, xml, xls, xlsx, html, dBase III-IV, Clipper, FoxPro 2.x and Excel 2003 XML files.

2 Creating new databases

To create a new database

1. On the File menu, click **New Database**.
2. In the displayed Add Table dialog box, enter the name of the first table.
Optionally, you can choose to import that new table from another existing file/database in any of the supported file formats.
To add other tables or folders later, use the **File > New** commands.



3. If this is a new empty table, you'll be prompted to add at least one field.
To add other fields later, use the **Database > Insert Field** and **Database > Append Field** command.
4. **Using date/time fields:**
Any **Text** field containing (as shown below) generic date/time strings can be used as a date/time field.
This standard/generic format is required by all built-in date/time functions (save **datevalue()** and **timevalue()**) and ensures proper sorting.
To display and enter dates/times in the local system format and to display the calendar control (see **Settings > Options > Editing > Use date picker when editing dates**), set the desirable "date" or "time" style for such a **Text** field.
Text fields containing non-generic dates/times should be converted with the **Edit > Convert Field Data > Text To Standard Date String** command prior to setting the "date" style.

Field types available in GS-Base:

| Type | Examples | Comments |
|------|----------|----------|
| | | |

| | | |
|-----------|---|---|
| Numeric | 123 -123 123.09 1.23e+02 | <p>The actual decimal and thousand separators depend on your system regional settings and the current Settings > Locales menu selection.</p> <p>Max. positive value: 1.8+308 Min. positive value: 2.2-308 Precision: up to 15 digits</p> |
| Text | Dog short text 1 2 3 11:34:30 PM 7/20/2007 P1Y2M3DT10H30M50S.000 -P120D | <p>Any text containing up to 8169 bytes (*).</p> <p>When entering date/time and time period values you must use one of the two string forms: (1) the current date/time style that you specified for a given field in the "Format / Style" dialog box, for example: 7/20/2007 7/20/2007 23:34:30 or (2) the generic date/time and period string form as specified by www.w3.org for the (xsd schema) data/time/period data types, for example: 2006-01-31 2006-01-31T13:10:55 2006-01-31T13:10:55.123 2006-01-31T23:30:00+02:00 13:10:00 13:10:55.123 13:10:55-00:30 P1Y2M3DT10H30M50S.000 (which means a period of: 1 year, 2 months, 3 days, 10 hours, 30 minutes, 50 seconds, 0 milliseconds) -P120D (which means a period of minus 120 days) PT63H (which means 63 hours)</p> |
| Long Text | | <p>Any text of any size. By default, the text is saved in the Rich Text (*.rtf) format and can contain any supported formatting and inserted graphics or objects.</p> <p><u>Right-click the field to display the context menu and available editing commands and options. Clicking the Edit in External Application command you can edit the text in the application associated with the *.rtf or *.txt format. Changes are saved automatically after choosing the Save command in that application.</u></p> <p>To save only plain text (which may result in smaller files and/or faster loading saving if there are a lot of entries of that type), change the</p> |

| | | |
|--------------|--|---|
| | | corresponding option in the Settings > Options dialog. |
| Images/Files | | <p>Any number of images or any other files. Files other than *.jpeg, *.png, and *.bmp are represented as icons associated with their file extensions.</p> <p>Use the context menu or the Settings > Options dialog box to turn on/off displaying thumbnails and/or file information (file name, size, modification date).</p> <p>If you insert some link *.lnk files linking to some disk files and if you later change the location of those target files, you can specify the new shortcut path globally in the Settings > Options dialog.</p> <p><u>Right-click or double-click a given object to display the context menu and available editing commands and options.</u></p> <p><u>Clicking the Edit in External Application command you can edit the object in the application associated with the file type of that object. Changes are saved automatically after choosing the Save command in that application.</u></p> |
| Code | | Any text of any size. The type/language of code is determined by the Subfield selection in the Field Setup dialog box. The code editor uses line numbering and syntax coloring based on the Scintilla (c) project. |

(*) - The maximum number of bytes refers to the UTF-8 string representation. In practice, for users using Latin characters this value equals to the number of characters, however for e.g. far-east languages characters require a few byte UTF-8 sequences hence the character limitation will be smaller.

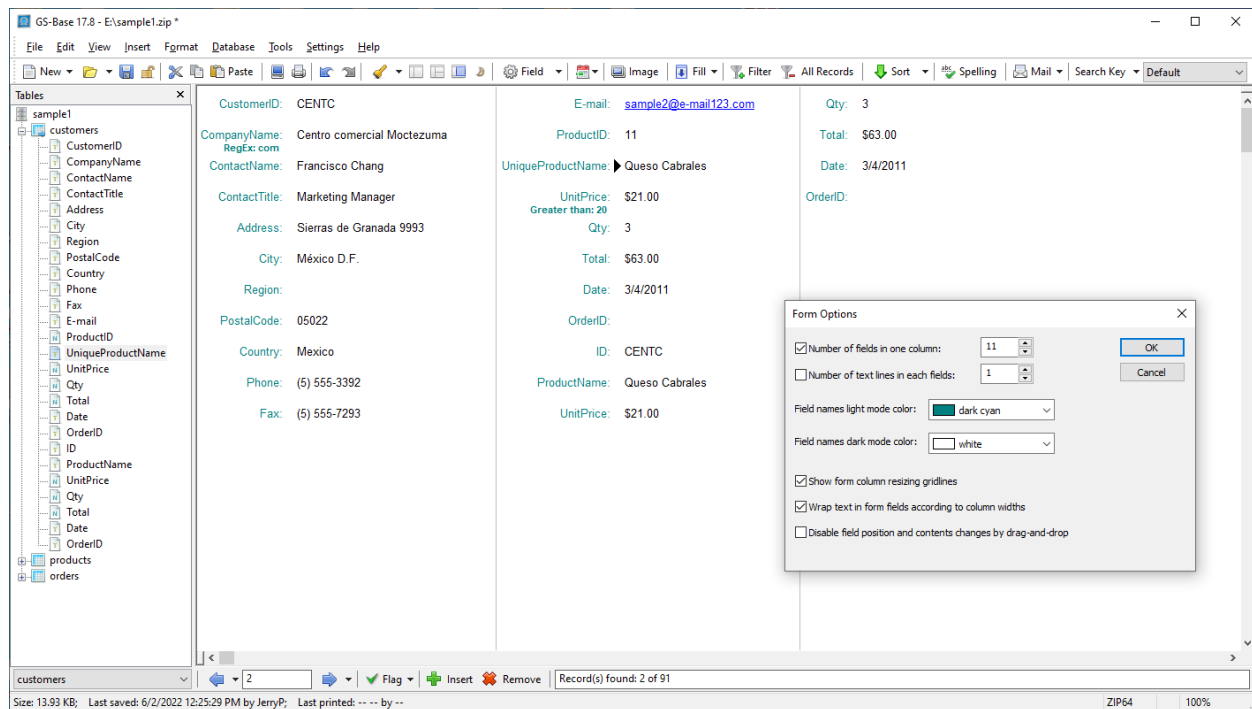
3 Using table, form and binary views

By default, records in a newly created database are displayed in tables. Tables can be split into separate views to enable synchronized scrolling of different regions of a given table.

Form views can be activated using the **View > Form** command. They can be displayed side by side with the corresponding table view or the table view can be hidden.

The form layout is mostly automatic. You can specify various other options using the **View > Form Options** command.

Multi-column forms enables you e.g. to view/inspect tens of fields at once.



Binary views are used to display Long Text, Images/Files and Code fields. They are opened automatically if you add such contents. You can still explicitly show/hide them using the **View** menu commands or simply pressing Enter or double-clicking the corresponding table/form field.

4 GS-Base zip database file format

GS-Base databases are zip archives saved either with the *.zip or *.gsb extensions. For more information about which extension you should choose, please see the Default file extension option in the Settings > Options dialog box.

There are two zip formats that can be used in GS-Base: zip64 and the standard zip (zip32). Zip32 files have a limit of 4GB for one database file and the number of all memo fields and inserted images (in other words: zip streams/entries) in one file is approx. 64K. Zip64 doesn't have the above limitations, but some 3rd party zip archivers may use their own non-standard zip64 format variants so if for any reasons you need to edit such files without GS-Base you should make sure the data exchange is possible.

The ZIP64/ZIP32 button on the main window status bar indicates which zip format is currently in use. You can click it and switch between them freely. If you try to save a zip32 file with the amount of data that might exceed the zip32 limits, GS-Base will display a warning message.

Simplified GS-Base zip files contain only plain text files representing tables and can be created manually by users or by other applications. Typically, they can be used if you want

to transfer a bunch of text files to GS-Base at once, without using the COM programming. For details please see: Simplified GS-Base files.

A database zip file contains files of the following categories:

1. Database records stored as

[table name].txt

text files. These are plain comma-separated UTF-8 text files containing field names in the first row and records in the subsequent rows.

You can edit these files manually without GS-Base using any text editor capable of saving UTF-8 text files.

Deleting such a file from the database zip archive means deleting all records. If you insert any text (*.txt, *.csv, *.tsv, *.tab) file into the zip, GS-Base will detect and report a new table(s) and will enable you either to setup and add it to the database structure or to retain it in the zip as an unused file.

2. **Long Text, Images/Files** and **Code** field subfolders. Each such subfolder has the following form:

[table name].r[9-digit record number]f[5-digit field number]

For **Long Text** fields, each such subfolder contains one *.rtf or *.txt file.

For **Images/Files** fields, each such subfolder contains any number of any files.

Code field subfolders contain one *.txt file. You can edit/add/remove these files manually without GS-Base and add/remove/edit subfolders or their names.

3. One configuration **config.dat** file containing various information about the database: the order of tables, page settings and formatting.
If you delete this file manually, GS-Base can still use all the record tables and binary fields and re-import them automatically as if they were new tables.
4. One **META-INF/manifest.txt** file containing the list of all files used by the database. If the file is encrypted, that file also contains various encryption parameters.
If the database is not encrypted, deleting this file doesn't affect the database.
If the database is encrypted, you must not modify that file in any way or you'll risk losing the data.
5. Any number of any files not used by GS-Base that you can drag-and-drop into the database zip file manually.
If the **Settings > Options > Show the unused files list** option is on, GS-Base will report detecting such files. When the database zip file is saved, all such unused files are stored in the separate **unused/** zip subfolder.

Note: If your database contains a very large numbers of memo/image/files/code objects (e.g. thousands), it's recommended to use the *.gsb extension as processing such zip's by Windows (e.g. when Windows performs file searching/indexing) might be very slow.

5 Managing database fields and tables

The Database Explorer pane enables you to add, delete, re-arrange and rename fields, tables and folders. You can use the **Copy/Paste** commands, the drag-and-drop functions and the commands available via the context menu (displayed after right-clicking the explorer pane).

Dropping the dragged table over a folder places the table at the end of the table list in that folder.

Dropping a field over the table places that field at the end of the field list.

To rename any tree item, select it, then click it or use the **Rename** command from the context menu.

The **Display Field Name With** command toggles on/off displaying optional additional field information: the field type, its status ("calculated", "validation", "conversion", "default value") and whether the field is currently filtered.

To display the Field Setup dialog box, double click a given field in the Database Explorer pane or click that command on the main menu or the context menu. The dialog box offers the following options:

Field name

Any string starting with a letter and containing up to 63 characters.

Field names should be unique within a given table as otherwise formulas referring to record fields may produce incorrect results.

Field type

One of the available four field types: "Text", "Numeric", "Long Text", "Files/Images" and "Code".

For details, please see: Creating new databases

Changing an existing field type may result in conversion and/or modifying the existing field contents.

Special

This value specifies whether the field features some additional functionality.

- **Calculation formula** - the specified formula will be used to calculate the current field value automatically.
Calculated fields display calculation results only and can not be edited. They are updated in the following situations:
 - After a given record is modified, all calculated fields within that records are updated.
 - After choosing the **Tools > Update Data (F9)** command, calculated fields of all records in the current table are updated.

Creating links between two tables is a special case of using calculated fields and the **vLookUp_ex** function from the **Cross-table summarizing & lookup** formula category. It behaves similarly to the standard spreadsheet **vLookUp**.

For example, see the included "sample.zip" database and the calculated fields used to perform calculation on record fields and to link the "products" and "orders" tables. For more information about the formula syntax, see: Entering formulas.

- **Validation formula** - the specified formula will be used to validate data entered in this field. The data is accepted if the validation formula returns a numeric value other than 0/false. For example, if some field ("field_1") should contain only strings at least 4 characters long, you can specify the following formula:
`=len(field_1) >= 4`
For more information about the formula syntax, see: Entering formulas.
- **Conversion formula** - the specified formula will be used to convert data entered in this field.
For example, to ensure that the entered data doesn't contain leading spaces and first letters in words are uppercase letters, the following formula can be used:
`=proper(trim(field_1))`
For more information about the formula syntax, see: Entering formulas.
- **Default value** - a fixed value that is inserted into selected empty fields after choosing the **Insert > Default / Incremented Max (Ctrl+T)** command. Also see: Default field values.
- **Incremented Maximum** - the current (or initial) field maximum value is incremented by 1 and inserted into selected empty fields after choosing the **Insert > Default / Incremented Max (Ctrl+T)** command. The field must be numeric or must contain date/time values.

Statistics

Calculates the breakdown (field value/number of occurrences) for a given field. The generated sorted list has a form of plain text so it can be selected and copied on to the Clipboard. This and much more advanced functionality is provided by Pivot Tables.

Formulas

Lists all predefined GS-Base functions that can be used in formula expressions in database fields. Database fields are referenced in these expressions by names.

6 Editing field contents

To edit Numeric and Text fields in tables/forms

Do one of the following:

- Press **Enter**, **F2** or any letter/digit key.
- Double-click a given field.

You can enter up to 8169 characters in the standard **Text** fields. The **Long Text** and **Code** fields can contain up to 4GB of text. The **Images/Files** fields can contain any number of objects. In the GS-Base 16.4 and older versions, the total size of inserted objects and the size of database file itself can't exceed 4GB and the total number of binary field entries can't exceed 65K. In newer GS-Base versions the zip64 file format replaced the standard zip format and all the above limitations were lifted (they now equal thousands of TB).

- To complete editing, press **Enter** or - if editing was started by pressing any character key - any of the cursor keys or simply click the table/form.
- To enter a new line in the edited cell, press **Ctrl+Enter** or **Alt+Enter**.
- To cancel changes, press **Esc**.

For information about entering/editing Unicode characters, please see: Entering Unicode characters.

If you set some standard (not using custom patterns) numeric or date/time format for a given field, GS-Base will try to interpret the data according to that format. Fields formatted using custom patterns requires you to enter the data in the default format or in one of the standard formats. The date/time style can be applied to **Text** fields only.

Note: If you want to apply the date format to a text field that already contains some non-standard (that is, other than YYYY-MM-DD) text strings, you should first use the **Edit > Convert > Text Strings To Date Strings** command (or re-enter each date individually). This step is required to ensure correct sorting of such a field.

Trying to edit a **Long Text**, **Images/Files** or **Code** field within a table/form results in the following actions:

1. If the **Options > Settings > Table/Form Editing Options > Edit LongText/Files fields in new windows** option is **unchecked**, GS-Base will open/hide a new pane displaying that data for editing/viewing.
2. If the **Options > Settings > Table/Form Editing Options > Edit LongText/Files fields in new windows** option is **checked**, GS-Base will launch an application associated with the specified data type (e.g. typically Wordpad for text and Paint for graphics). Choosing the "Save" command in such an external application will automatically update the data in GS-Base.

For the action (2) the following additional functionality is supported:

- Pressing a digit key opens either the text or the n-th (1...9) object for editing in an external application.
- Pressing a letter key prints either the text or the n-th (a...z) object in an external application.

To edit Long Text, Images/Files and Images/Files fields

These fields are saved automatically. They feature all the standard functionality such as drag-and-drop (which includes dropping files/images from other applications) and additional commands (e.g. exporting and importing text, formatting and display options) accessible through the context menus.

To edit a given memo field or a given image/object in an external application, right click that content to display the context menu and choose the **Edit In External Application** command. The data will be saved in GS-Base automatically if you use the "Save" command in that external application.

If a given binary field is not empty, the respective table or form field contains some textual information describing it. If you want to delete the entire **Long Text, Images/Files** or **Code** field contents at once and remove the respective entry from the database zip file, simply delete the corresponding table/form cell.

To move/copy/paste field contents

Tables: Place the mouse cursor over the top edge of the selection and drag it.

You can move the data within the same window, different windows or applications - by default GS-Base copies (and provides to other applications) the table data as text and bitmaps.

If you drop a file(s) within the table view, GS-Base either pastes the file path if the cursor was over a hyperlink field or displays a dialog box enabling you to choose into which **Images/Files** field that file(s) should be inserted.

Binary Fields: The standard Windows drag-and-drop procedure applies. The dropped objects are always copied, not moved. If you want to insert a link into the **Images/Files** field, press and hold down the **Ctrl+Shift** keys at the same time.

To fill a given range with some values

1. Copy the value of the selected field or fields.
2. Select the destination range. The destination range cannot contain more columns than the source range. If it does, the additional columns are ignored and won't be filled.
3. Paste the copied data.

To use a drop-down list to enter data

Use the Format > List command and specify which list should be used for a given field(s). The same list can be shared by any number of fields in any number of tables in one database. A database file can contain up to 255 lists. The number of list items is unlimited. Drop-down list can be static (with values pre-defined by the user) and they can automatically add each new unique value entered by the user in the associated field(s).

To use a "Date and Time Picker" control to enter data

The control can be displayed for each text field that (1) is formatted using one of the standard "Date" styles (not using custom patterns) and (2) doesn't have any drop-down list attached to it at the same time.

Displaying that control can be enabled/disabled in the **Settings > Options** dialog box. The following shortcut keys are used by the date time picker control:

Arrows keys Changes the active control field and/or changes control field values

| | |
|---------------------|---|
| End and Home | Sets the maximum and minimum values for the current part of the date (a year, month or day) |
| Plus and Minus | Increments/decrements the current part of the date (a year, month or day) |
| Alt+Down | Displays the "Month Calendar" control. Same as clicking the "drop-down" arrow |
| PageUp, PageDown | If the "Month Calendar" control is active, moves to the previous/next month |

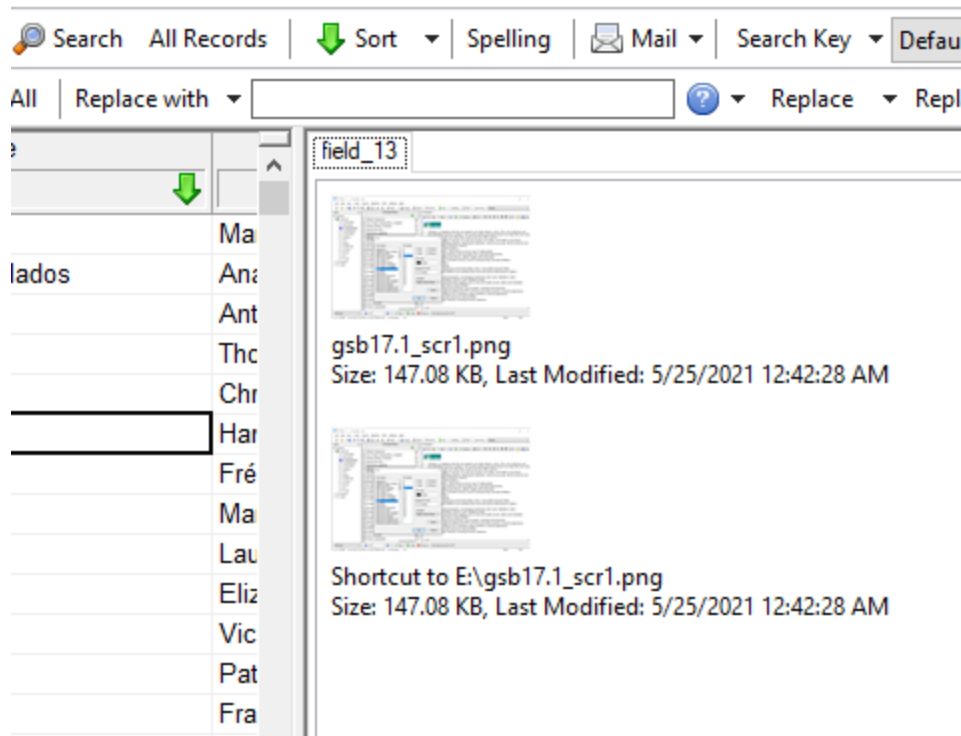
7 Entering Unicode characters

Unicode characters can be edited/entered using the following methods:

1. Enter a sequence of 1-6 hexadecimal digits and immediately press **Alt+X**. Note that any preceding and not separated valid hexadecimal digits will be converted as well. This method can be used only when editing fields; it's not available in dialog boxes.
2. Press **Ctrl+Shift+U** then enter a sequence of 1-6 hexadecimal digits and immediately press **Space**.
3. Press and hold down **Alt**, then on the numeric pad press **+** and a desirable hexadecimal code, then release **Alt**. This method may require adding the following key in the Windows registry:
HKEY_CURRENT_USER/Control Panel/Input Method/EnableHexNumpad
The value of the above key should be set to "1" (entered as REG_SZ).

8 Inserting images/files vs inserting links to images/files

Images and files can be inserted in the Images/Files fields as objects that are stored in the database files along with tables or as links to files that are stored on user's disk and have to be copied separately whenever the database is moved to another computer. On the other hand, if the total size of all objects is very big, a database with embedded objects will have to load them into computer memory all at once when that database is opened while linked objects will be loaded gradually only when they are displayed. For images, both methods result in displaying them in the fields in the same way, for example:



For linked objects, in order not to require to keep their exact paths on different computers, GS-Base enables you to define the "Settings > Relative shortcut path" folder where you intend to copy the linked files. If a given original link point to a location which doesn't exist, GS-Base will look for the specified file name in that alternative folder.

To insert an image or a file in the "Memo/Files" pane use of the following methods:

- Use the "Insert > Image/File (Ctrl+M)" command. If no "Images/Files" fields exists GS-Base will offer to create one.
- Click the "Insert Image/File" toolbar button.
- Drag-and-drop the desirable file.

To insert a link in the "Memo/Files" pane use of the following methods:

- Use the "Insert > Shortcut to Image/File (Ctrl+Shift+M)" command. If no "Images/Files" fields exists GS-Base will offer to create one.
- Press Shift and click the "Insert Image/File" toolbar button.
- Drag-and-drop the desirable file pressing the Ctrl+Shift keys.

To insert a link in the "Memo/Files" in a table and a plain text field:

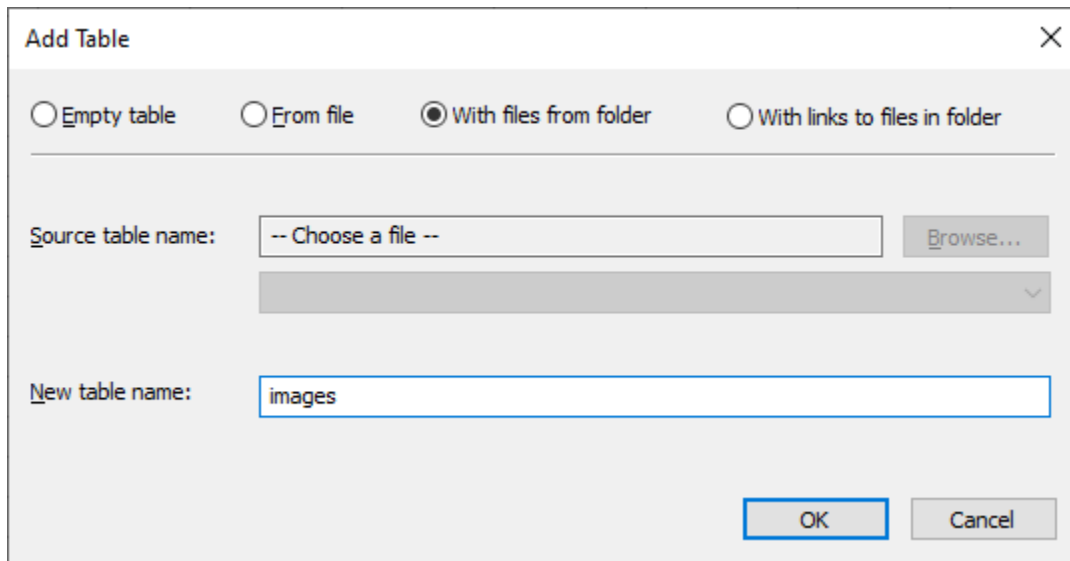
1. Format a given field using the "Format > Hyperlink" command.
2. Drag-and-drop the desirable file to that text field location. This will cause inserting the link/path in that text field. (Another option is to simply enter the path manually.)
Clicking such a link corresponds to the "Edit" Windows shell command for the linked file.

9 Loading images/files form entire folders

To load all images/files from a given folder use **File > New Database** or **File > New Table** command and in the displayed **Add table** dialog box choose the option "With files from folder" or "With links to files in folder".

The first option inserts files directly in the database file, the 2nd one only links to files in that folder.

This will create a table with records with one loaded image/file or link per field.



Note 1.: By default, the loaded files are placed in the newly created "Images/Files" field. If these are images, they will be displayed in the binary field panes as usual. All other files will be displayed as their associated icons. If these are e.g. text, html, code snippets etc. or any files containing textual contents, you can later use the **Database > Field Setup** command and dialog box to change the field type to "Long Text" or "Code" and display the actual text in the binary field pane.

Note 2.: You can also load from all images/files from a given folder to a single "Images/Files" field or save all images/files from a given "Images/Files" field to a folder. To do this, right-click that "Images/Files" field to display the context menu.

10 Entering formulas

To enter/edit a formula

Formulas available in GS-Base are similar to those used in spreadsheets with a few exceptions:

- Cell references are replaced by field names. All field names occurring in a given formula refer to one and the same record.

- The range ":" operator is not available. Formulas requiring array/range parameters can use the "array()" function that creates an array out of database fields, strings, numbers etc.
- References to other tables via the "!" and "_" operators are not available. Instead, there is a separate category of functions: **Cross-table summarizing & lookup** functions that groups functions accessing data from other database tables in the same database. Some of those functions can be used to create links/relations between tables. (See the included "sample.zip" database and its calculated fields used to link the "products" and "orders" tables.)

For example:

- to sum up two fields 'subtotal' and 'tax', use the following formula:

=subtotal + tax

- to merge two fields 'name' and 'country', use the following formula:

=name & ", " & country

- and to obtain a period string representing the difference between two given dates:

=dateDiff("2011-01-01", "2011-12-31") (which returns "P364D")

- to find the current age for birth dates stored in the "date" field:

=year(today()) - year(date) - if(month(date) < month(today()), 0, if(month(date) > month(today()), 1, day(date) >= day(today())))

Note: If the "date" field is not a valid GS-Base date field (that is, a text field containing generic date/time YYYY-MM-DD strings that can be formatted to be displayed in the desirable local format), the above "date" argument should be replaced by dateValue(date).

A few other examples:

=Unit_Price*Qty ="abc"

=1 + 2

="a" & 'b' & 3

=fv(0.75%, 36, -500, -5500, 0)

=LProg({2,2;1,2;4,0}, {1;1;1}, {14;8;16}, {2; 4},0,,)

=sum(field1, field2, field3, sum(field4, field5, field6))

Numbers and dates used as arguments cannot be formatted. For example, the following expressions are incorrect:

=\$1,000.00 + 1

=dateDiff("6/3/11", "8/9/2011")

unless the description specifically states that parameters are arbitrary text strings to be converted to a specific type, like:

```
=value("$1,000.00") + 1  
=dateDiff(dateValue("6/3/11"), dateValue("8/9/2011"))
```

Text strings should be delimited either by single or double quotation marks.

Formulas should not contain circular field references.

To browse all available formula categories and examples, please see the Field Setup dialog box or the Search dialog box.

Note: Square brackets [,] in parameter lists denote parameters that are optional and can be omitted. For example:

LProg(A, s, b, c, vector, **[epsilon]**, **[m]**)

However, you still have to use the corresponding parameter list separators, that is:

LProg(A, s, b, c, vector,,)

If your current Windows regional settings define commas as decimal separators, use semicolons (;) to separate function arguments or - if you prefer to use system independent (US) settings - select the generic locales via the **Settings > Locales > Generic** command.

Available operators:

| Operator | Operation | Comments | Precedence |
|----------|-----------------------|--|------------|
| = | Equal | Compares numbers or text strings (the comparison is not case-sensitive). Example: A1=4 , B2="abc" | 6 |
| < | Less than | Compares numbers or text strings (the comparison is not case-sensitive). Example: A1<4 , B2<"abc" | 6 |
| > | Greater than | Compares numbers or text strings (the comparison is not case-sensitive). Example: A1>4 , B2>"abc" | 6 |
| <= | Less than or equal | Compares numbers or text strings (the comparison is not case-sensitive). Example: A1<=4 , B2<="abc" | 6 |
| >= | Greater than or equal | Compares numbers or text strings (the comparison is not case-sensitive). Example: A1>=4 , B2>="abc" | 6 |

| | | | |
|----|----------------------|--|---|
| <> | Not equal | Compares numbers or text strings (the comparison is not case-sensitive). Example: A1<>4 , B2<>"abc" | 6 |
| + | Addition | Adds numbers. | 5 |
| - | Subtraction | | 5 |
| & | String concatenation | Merges text strings. Example: A1 & "abc" , "a" & "b" | 5 |
| * | Multiplication | Multiplies numbers. | 4 |
| / | Division | Divides numbers. | 4 |
| ^ | To the power of | Calculates the power of. | 3 |
| - | Negative | Example: -A1 | 2 |
| % | Percent | Specifies a number entered as a percentage. Example: 12% | 1 |

11 'Cross-table' calculation functions

The functions listed below are a separate category of functions listed in the **Field Setup** dialog box and can be useful for performing mass calculated field updating in a given table based on its relation with other tables and filtered records values from other tables. For best performance, to update calculating fields when both these table types have huge number of records at the same time, use those functions listed below that use binary searching (and perform automatic background sorting and indexing). You can also increase the number of used processor cores in the Settings > Options dialog box.

sum_ex(table, v, searchField, calcField, options, [emptyValue])

Searches the 'searchField' field of the specified 'table' for the 'v' value and returns the sum of the 'calcField' field values for all found records. If 'v' is a name (text string) specified without quotation marks, it's assumed to be a name of the field from the table where the formula is used. The 'table', 'searchField' and 'calcField' parameters must be placed in quotation marks.

If 'table' is located in a nested folder, its full path must be specified with the inner slashes.

NOTE: if the fast-binary searching is used, the type of the 'v' parameter must match the type of the searched field. They must be both either textual or numeric. Otherwise an error is returned."

The 'option' argument is a sum of the following values:

0 - the 'table' will be searched for 'v' using the fastest binary searches; case-insensitive; this is the recommended option
if the source record set contains a large number of records with the 'v' values and the external 'table' has many records to search as well.

1 - performs case-sensitive searching,

2 - performs string sorting/comparison (hyphen and apostrophe are sorted, as opposed to the default word-sorting),

4 - the 'v' value is a regular expression; instead of the fast binary exact searches, slower sequential searching will be used,

8 - allows empty matches when using regular expressions.

The 'empty value' argument specifies what should be returned (a number or a text string) if no 'v' values are found.

If it's left empty, the function returns the #NULL! error code.

```
=sum_ex("products", ProductID, "ProductID", "ProductName",0, 0)
```

```
=sum_ex("folder1/products", ProductID, "ProductID", "UnitPrice",4, -1)
```

min_ex(table, v, searchField, calcField, options, [emptyValue])

Searches the 'searchField' field of the specified 'table' for the 'v' value and returns the minimum of the 'calcField' field values for all found records. If 'v' is a name (text string) specified without quotation marks, it's assumed to be a name of the field from the table where the formula is used. The 'table', 'searchField' and 'calcField' parameters must be placed in quotation marks. If 'table' is located in a nested folder, its full path must be specified with the inner slashes.

NOTE: if the fast-binary searching is used, the type of the 'v' parameter must match the type of the searched field. They must be both either textual or numeric. Otherwise an error is returned."

The 'option' argument is a sum of the following values:

0 - the 'table' will be searched for 'v' using the fastest binary searches; case-insensitive; this is the recommended option
if the source record set contains a large number of records with the 'v' values and the external 'table' has many records to search as well.

1 - performs case-sensitive searching,

2 - performs string sorting/comparison (hyphen and apostrophe are sorted, as opposed to the default word-sorting),

4 - the 'v' value is a regular expression; instead of the fast-binary exact searches, slower sequential searching will be used,

8 - allows empty matches when using regular expressions.

The 'empty_value' argument specifies what should be returned (a number or a text string) if no 'v' values are found.

If it's left empty, the function returns the #NULL! error code.

```
=min_ex("products", ProductID, "ProductID", "ProductName",0, 0)
```

```
=min_ex("folder1/products", ProductID, "ProductID", "UnitPrice",4, -1)
```

max_ex(table, v, searchField, calcField, options, [emptyValue])

Searches the 'searchField' field of the specified 'table' for the 'v' value and returns the maximum of the 'calcField' field values for all found records. If 'v' is a name (text string) specified without quotation marks, it's assumed to be a name of the field from the table where the formula is used. The 'table', 'searchField' and 'calcField' parameters must be placed in quotation marks. If 'table' is located in a nested folder, its full path must be specified with the inner slashes.

NOTE: if the fast-binary searching is used, the type of the 'v' parameter must match the type of the searched field. They must be both either textual or numeric. Otherwise an error is returned."

The 'option' argument is a sum of the following values:

0 - the 'table' will be searched for 'v' using the fastest binary searches; case-insensitive; this is the recommended option
if the source record set contains a large number of records with the 'v' values and the external 'table' has many records to search as well.

1 - performs case-sensitive searching,

2 - performs string sorting/comparison (hyphen and apostrophe are sorted, as opposed to the default word-sorting),

4 - the 'v' value is a regular expression; instead of the fast-binary exact searches, slower sequential searching will be used,

8 - allows empty matches when using regular expressions.

The 'empty_value' argument specifies what should be returned (a number or a text string) if no 'v' values are found. If it's left empty, the function returns the #NULL! error code.

```
=max_ex("products", ProductID, "ProductID", "ProductName",0, 0)
```

```
=max_ex("folder1/products", ProductID, "ProductID", "UnitPrice",4, -1)
```

quartile1_ex(table, v, searchField, calcField, options, [emptyValue])

Searches the 'searchField' field of the specified 'table' for the 'v' value and returns the 1st quartile of the 'calcField' field values for all found records. If 'v' is a name (text string) specified without quotation marks, it's assumed to be a name of the field from the table where the formula is used. The 'table', 'searchField' and 'calcField' parameters must be placed in quotation marks. If 'table' is located in a nested folder, its full path must be specified with the inner slashes.

NOTE: if the fast-binary searching is used, the type of the 'v' parameter must match the type of the searched field. They must be both either textual or numeric. Otherwise an error is returned."

The 'option' argument is a sum of the following values:

0 - the 'table' will be searched for 'v' using the fastest binary searches; case-insensitive; this is the recommended option
if the source record set contains a large number of records with the 'v' values and the external 'table' has many records to search as well.

1 - performs case-sensitive searching,

2 - performs string sorting/comparison (hyphen and apostrophe are sorted, as opposed to the default word-sorting),

4 - the 'v' value is a regular expression; instead of the fast binary exact searches, slower sequential searching will be used,

8 - allows empty matches when using regular expressions.

The 'empty_value' argument specifies what should be returned (a number or a text string) if no 'v' values are found.

If it's left empty, the function returns the #NULL! error code.

```
=quartile1_ex("products", ProductID, "ProductID", "ProductName",0, 0)  
=quartile1_ex("folder1/products", ProductID, "ProductID", "UnitPrice",4, -1)
```

median_ex(table, v, searchField, calcField, options, [emptyValue])

Searches the 'searchField' field of the specified 'table' for the 'v' value and returns the median of the 'calcField' field values for all found records. If 'v' is a name (text string) specified without quotation marks, it's assumed to be a name of the field from the table where the formula is used. The 'table', 'searchField' and 'calcField' parameters must be placed in quotation marks. If 'table' is located in a nested folder, its full path must be specified with the inner slashes.

NOTE: if the fast-binary searching is used, the type of the 'v' parameter must match the type of the searched field. They must be both either textual or numeric. Otherwise an error is returned."

The 'option' argument is a sum of the following values:

0 - the 'table' will be searched for 'v' using the fastest binary searches; case-insensitive; this is the recommended option

if the source record set contains a large number of records with the 'v' values and the external 'table' has many records to search as well.

1 - performs case-sensitive searching,

2 - performs string sorting/comparison (hyphen and apostrophe are sorted, as opposed to the default word-sorting),

4 - the 'v' value is a regular expression; instead of the fast-binary exact searches, slower sequential searching will be used,

8 - allows empty matches when using regular expressions.

The 'empty_value' argument specifies what should be returned (a number or a text string) if no 'v' values are found.

If it's left empty, the function returns the #NULL! error code.

```
=median_ex("products", ProductID, "ProductID", "ProductName",0, 0)
```

```
=median_ex("folder1/products", ProductID, "ProductID", "UnitPrice",4, -1)
```

quartile3_ex(table, v, searchField, calcField, options, [emptyValue])

Searches the 'searchField' field of the specified 'table' for the 'v' value and returns the 3rd quartile of the 'calcField' field values for all found records. If 'v' is a name (text string) specified without quotation marks, it's assumed

to be a name of the field from the table where the formula is used. The 'table', 'searchField' and 'calcField' parameters

must be placed in quotation marks. If 'table' is located in a nested folder, its full path must be specified with the inner slashes.

NOTE: if the fast-binary searching is used, the type of the 'v' parameter must match the type of the searched field. They must be both either textual or numeric. Otherwise an error is returned."

The 'option' argument is a sum of the following values:

0 - the 'table' will be searched for 'v' using the fastest binary searches; case-insensitive; this is the recommended option

if the source record set contains a large number of records with the 'v' values and the external 'table' has many records to search as well.

1 - performs case-sensitive searching,

2 - performs string sorting/comparison (hyphen and apostrophe are sorted, as opposed to the default word-sorting),

4 - the 'v' value is a regular expression; instead of the fast-binary exact searches, slower sequential searching will be used,

8 - allows empty matches when using regular expressions.

The 'empty_value' argument specifies what should be returned (a number or a text string) if no 'v' values are found.

If it's left empty, the function returns the #NULL! error code.

```
=quartile3_ex("products", ProductID, "ProductID", "ProductName",0, 0)
=quartile3_ex("folder1/products", ProductID, "ProductID", "UnitPrice",4, -1)
```

mode_ex(table, v, searchField, calcField, options, [emptyValue])

Searches the 'searchField' field of the specified 'table' for the 'v' value and returns the most frequently occurring value of the 'calcField' field values for all found records. If 'v' is a name (text string) specified without quotation marks, it's assumed to be a name of the field from the table where the formula is used. The 'table', 'searchField' and 'calcField' parameters must be placed in quotation marks. If 'table' is located in a nested folder, its full path must be specified with the inner slashes.

NOTE: if the fast-binary searching is used, the type of the 'v' parameter must match the type of the searched field. They must be both either textual or numeric. Otherwise an error is returned."

The 'option' argument is a sum of the following values:

- 0 - the 'table' will be searched for 'v' using the fastest binary searches; case-insensitive; this is the recommended option if the source record set contains a large number of records with the 'v' values and the external 'table' has many records to search as well.
- 1 - performs case-sensitive searching,
- 2 - performs string sorting/comparison (hyphen and apostrophe are sorted, as opposed to the default word-sorting),
- 4 - the 'v' value is a regular expression; instead of the fast-binary exact searches, slower sequential searching will be used,
- 8 - allows empty matches when using regular expressions.

The 'empty_value' argument specifies what should be returned (a number or a text string) if no 'v' values are found.

If it's left empty, the function returns the #NULL! error code.

```
=mode_ex("products", ProductID, "ProductID", "ProductName",0, 0)
=mode_ex("folder1/products", ProductID, "ProductID", "UnitPrice",4, -1)
```

count_ex(table, v, searchField, options, [emptyValue])

Searches the 'searchField' field of the specified 'table' for the 'v' value and returns the number of found records.

If 'v' is a name (text string) specified without quotation marks, it's assumed to be a name of the field from the table where the formula is used. The 'table' and 'searchField' parameters must be placed in quotation marks. If 'table' is located in a nested folder, its full path must be specified with the inner slashes.

NOTE: if the fast-binary searching is used, the type of the 'v' parameter

must match the type of the searched field. They must be both either textual or numeric. Otherwise an error is returned."

The 'option' argument is a sum of the following values:

0 - the 'table' will be searched for 'v' using the fastest binary searches; case-insensitive; this is the recommended option if the source record set contains a large number of records with the 'v' values and the external 'table' has many records to search as well.

1 - performs case-sensitive searching,

2 - performs string sorting/comparison (hyphen and apostrophe are sorted, as opposed to the default word-sorting),

4 - the 'v' value is a regular expression; instead of the fast-binary exact searches, slower sequential searching will be used,

8 - allows empty matches when using regular expressions.

The 'empty_value' argument specifies what should be returned (a number or a text string) if no 'v' values are found.

If it's left empty, the function returns the #NULL! error code.

```
=count_ex("products", ProductID, "ProductID", 0, 0)
```

```
=count_ex("folder1/products", ProductID, "ProductID", 4, -1)
```

sumIfs_ex(table, calcField, field1, criteria1 [, field2, criteria2, ...], [emptyValue])

Returns the sum of the 'calcField' field values from the specified 'table' for records that meet the specified list of filters/criteria.

The 'table', 'calcField' and 'field(n)' parameters must be placed in quotation marks. If 'table' is located in a nested folder, its full path must be specified with the inner slashes.

References to field values/names in the same table where the formula resides must be used without any quotation marks.

The criteria can be one of the following:

(1) a name of the field in the current table, optionally combined with the =,>,>=,<,<= operators,

(2) a text string beginning with the =,>,>=,<,<= operators - numeric and date/time searches requires unformatted numbers and generic date/time strings (YYYY-MM-DD),

(3) a number or a search pattern: a text string optionally containing special characters '?' (any character) or '*' (any string). To search for ? or * place a tilde (~) before them. Numbers and dates must be used in the unformatted form.

The 'empty_value' argument specifies what should be returned (a number or a text string) if no 'v' values are found.

If it's left empty, the function returns the #NULL! error code. NOTE: If some 'criteria' can contain pattern characters (*?~) but you don't want to perform pattern searches, use the '&' operator to prefix that criteria with '='.
For example: "=" & CustomerID

NOTE: Unlike the sum_ex() function, the sumIifs_ex() function always performs slower sequential searching. If both the source and target tables contain hundreds of thousands or millions records at the same time, it might be too slow. You can improve its performance by including the leading "=" before each criterion (to avoid pattern matching) and by increasing the number of used processors in the 'Settings > Options' dialog box.

=sumIifs_ex("orders", "Total", "id", CustomerID,0) returns the total sales amount for a given customer; 'orders' is a table with all orders, 'id' and 'Total' are its fields containing respectively customers' IDs and total values for each order and 'CustomerID' is a field of the table where the result is placed.

=sumIifs_ex("folder1/orders", "Total", "id", "=" & CustomerID,) returns the total sales amount for a given customer; the CustomerID may contain pattern characters but no pattern search will be performed.

=sumIifs_ex("orders", "Qty", "ProductName", "Tofu", "ProductName", "Chocolade",0)

=sumIifs_ex("folder1/folder2/orders", "Total", "Date", ">2011-01-01", "Date", "
"

minIifs_ex(table, calcField, field1, criteria1 [, field2, criteria2, ...], [emptyValue])

Returns the minimum of the 'calcField' field values from the specified 'table' for records that meet the specified list of filters/criteria.

The 'table', 'calcField' and 'field(n)' parameters must be placed in quotation marks. If 'table' is located in a nested folder, its full path must be specified with the inner slashes.

References to field values/names in the same table where the formula resides must be used without any quotation marks.

The criteria can be one of the following:

(1) a name of the field in the current table, optionally combined with the =,>,>=,<,<= operators,

(2) a text string beginning with the =,>,>=,<,<= operators - numeric and date/time searches requires unformatted numbers and generic date/time strings (YYYY-MM-DD),

(3) a number or a search pattern: a text string optionally containing special characters '?' (any character) or '*' (any string). To search for ? or * place a tilde (~) before them. Numbers and dates must be used in the unformatted form.

The 'empty_value' argument specifies what should be returned (a number or a text string) if no 'v' values are found.
If it's left empty, the function returns the #NULL! error code. NOTE: If some 'criteria' can contain pattern characters (*?~) but you don't want to perform pattern searches, use the '&' operator to prefix that criteria with '='.
For example: "=" & CustomerID

NOTE: Unlike the min_ex() function, the minIfs_ex() function always performs slower sequential searching. If both the source and target tables contain hundreds of thousands or millions records at the same time, it might be too slow.
You can improve its performance by including the leading "=" before each criterion (to avoid pattern matching) and by increasing the number of used processors in the 'Settings > Options' dialog box.

=minIfs_ex("orders", "Total", "id", CustomerID, 0) returns the minimum sale amount for a given customer;
'orders' is a table with all orders, 'id' and 'Total' are its fields containing respectively customers' IDs and total values for each order and 'CustomerID' is a field of the table where the result is placed.

=minIfs_ex("folder1/orders", "Total", "id", "=" & CustomerID, -1) returns the minimum sale amount for a given customer;
the CustomerID may contain pattern characters but no pattern search will be performed.

=minIfs_ex("orders", "Qty", "ProductName", "Tofu", "ProductName", "Chocolate", -1)

=minIfs_ex("folder1/folder2/orders", "Total", "Date", ">2011-01-01", "Date", "
"

maxIfs_ex(table, calcField, field1, criteria1 [, field2, criteria2, ...], [emptyValue])

Returns the maximum of the 'calcField' field values from the specified 'table' for records that meet the specified list of filters/criteria.

The 'table', 'calcField' and 'field(n)' parameters must be placed in quotation marks. If 'table' is located in a nested folder, its full path must be specified with the inner slashes.

References to field values/names in the same table where the formula resides must be used without any quotation marks.

The criteria can be one of the following:

(1) a name of the field in the current table, optionally combined with the =,>,>=,<,<= operators,

(2) a text string beginning with the =,>,>=,<,<= operators - numeric and date/time searches requires unformatted numbers and generic date/time strings (YYYY-MM-DD),

(3) a number or a search pattern: a text string optionally containing special characters '?' (any character) or '*' (any string). To search for ? or * place a tilde (~) before them. Numbers and dates must be used in the unformatted form.

The 'empty_value' argument specifies what should be returned (a number or a text string) if no 'v' values are found.
If it's left empty, the function returns the #NULL! error code. NOTE: If some 'criteria' can contain pattern characters (*?~) but you don't want to perform pattern searches, use the '&' operator to prefix that criteria with '='.
For example: "=" & CustomerID

NOTE: Unlike the max_ex() function, the maxIfs_ex() function always performs slower sequential searching. If both the source and target tables contain hundreds of thousands or millions records at the same time, it might be too slow.
You can improve its performance by including the leading "=" before each criterion (to avoid pattern matching) and by increasing the number of used processors in the 'Settings > Options' dialog box.

```
=maxIfs_ex("orders", "Total", "id", CustomerID, -1)    returns the maximum  
sale amount for a given customer;  
'orders' is a table with all orders, 'id' and 'Total' are its fields  
containing respectively customers' IDs and total  
values for each order and 'CustomerID' is a field of the table where the  
result is placed.
```

```
=maxIfs_ex("folder1/orders", "Total", "id", "=" & CustomerID, -1)    returns  
the maximum sale amount  
for a given customer; the CustomerID may contain pattern characters but no  
pattern search will be performed.
```

```
=maxIfs_ex("orders", "Qty", "ProductName", "Tofu", "ProductName",  
"Chocolade", 0)
```

```
=maxIfs_ex("folder1/folder2/orders", "Total", "Date", ">2011-01-01", "Date",  
"
```

**quartile1Ifs_ex(table, calcField, field1, criteria1 [, field2, criteria2, ...],
[emptyValue])**

Returns the first quartile of the 'calcField' field values from the specified 'table' for records that meet the specified list of filters/criteria.

The 'table', 'calcField' and 'field(n)' parameters must be placed in quotation marks. If 'table' is located in a nested folder, its full path must be specified with the inner slashes.

References to field values/names in the same table where the formula resides must be used without any quotation marks.

The criteria can be one of the following:

(1) a name of the field in the current table, optionally combined with the =,>,>=,<,<= operators,

(2) a text string beginning with the =,>,>=,<,<= operators - numeric and date/time searches requires unformatted numbers and generic date/time strings (YYYY-MM-DD),

(3) a number or a search pattern: a text string optionally containing special characters '?' (any character) or '*' (any string). To search for ? or * place a tilde (~) before them. Numbers and dates must be used in the unformatted form.

The 'empty_value' argument specifies what should be returned (a number or a text string) if no 'v' values are found.

If it's left empty, the function returns the #NULL! error code. NOTE: If some 'criteria' can contain pattern characters (*?~) but you don't want to perform pattern searches, use the '&' operator to prefix that criteria with '='. For example: "=" & CustomerID

NOTE: Unlike the quartile1_ex() function, the quartile1Iifs_ex() function always performs slower sequential searching.

If both the source and target tables contain hundreds of thousands or millions records at the same time, it might be too slow.

You can improve its performance by including the leading "=" before each criterion (to avoid pattern matching) and by increasing the number of used processors in the 'Settings > Options' dialog box.

=quartile1Iifs_ex("orders", "Total", "id", CustomerID, -1) returns the 1st quartile of the sales amounts for a given customer;

'orders' is a table with all orders, 'id' and 'Total' are its fields containing respectively customers' IDs and total values for each order and 'CustomerID' is a field of the table where the result is placed.

=quartile1Iifs_ex("folder1/orders", "Total", "id", "=" & CustomerID,) returns the 1st quartile if the sales amounts for a given customer; the CustomerID may contain pattern characters but no pattern search will be performed.

=quartile1Iifs_ex("orders", "Qty", "ProductName", "Tofu", "ProductName", "Chocolade",)

=quartile1Iifs_ex("folder1/folder2/orders", "Total", "Date", ">2011-01-01", "Date", "

medianIifs_ex(table, calcField, field1, criteria1 [, field2, criteria2, ...], [emptyValue])

Returns the median of the 'calcField' field values from the specified 'table' for records that meet the specified list of filters/criteria.

The 'table', 'calcField' and 'field(n)' parameters must be placed in quotation marks. If 'table' is located in a nested folder, its full path must be specified with the inner slashes.

References to field values/names in the same table where the formula resides must be used without any quotation marks.

The criteria can be one of the following:

(1) a name of the field in the current table, optionally combined with the =,>,>=,<,<= operators,

(2) a text string beginning with the =,>,>=,<,<= operators - numeric and date/time searches requires unformatted numbers and generic date/time strings (YYYY-MM-DD),

(3) a number or a search pattern: a text string optionally containing special characters '?' (any character) or '*' (any string). To search for ? or * place a tilde (~) before them. Numbers and dates must be used in the unformatted form.

The 'empty_value' argument specifies what should be returned (a number or a text string) if no 'v' values are found.

If it's left empty, the function returns the #NULL! error code. NOTE: If some 'criteria' can contain

pattern characters (*?~) but you don't want to perform pattern searches, use the '&' operator to prefix that criteria with '='.

For example: "=" & CustomerID

NOTE: Unlike the median_ex() function, the medianIifs_ex() function always performs slower sequential searching.

If both the source and target tables contain hundreds of thousands or millions records at the same time, it might be too slow.

You can improve its performance by including the leading "=" before each criterion (to avoid pattern matching)

and by increasing the number of used processors in the 'Settings > Options' dialog box.

=medianIifs_ex("orders", "Total", "id", CustomerID, 0) returns the median of the sales amounts for a given customer;
'orders' is a table with all orders, 'id' and 'Total' are its fields containing respectively customers' IDs and total values for each order and 'CustomerID' is a field of the table where the result is placed.

=medianIifs_ex("folder1/orders", "Total", "id", "=" & CustomerID,) returns the median of the sales amounts for a given customer; the CustomerID may contain pattern characters but no pattern search will be performed.

=medianIifs_ex("orders", "Qty", "ProductName", "Tofu", "ProductName", "Chocolade",)

=medianIifs_ex("folder1/folder2/orders", "Total", "Date", ">2011-01-01", "Date", "

quartile3Iifs_ex(table, calcField, field1, criteria1 [, field2, criteria2, ...], [emptyValue])

Returns the the third quartile of the 'calcField' field values from the specified 'table' for records that meet the specified list of filters/criteria.

The 'table', 'calcField' and 'field(n)' parameters must be placed in quotation marks.

If 'table' is located in a nested folder, its full path must be specified with the inner slashes.

References to field values/names in the same table where the formula resides must be used without any quotation marks.

The criteria can be one of the following:

(1) a name of the field in the current table, optionally combined with the =,>,>=,<,<= operators,

(2) a text string beginning with the =,>,>=,<,<= operators - numeric and date/time searches requires unformatted numbers and generic date/time strings (YYYY-MM-DD),

(3) a number or a search pattern: a text string optionally containing special characters '?' (any character) or '*' (any string). To search for ? or * place a tilde (~) before them. Numbers and dates must be used in the unformatted form.

The 'empty_value' argument specifies what should be returned (a number or a text string) if no 'v' values are found.

If it's left empty, the function returns the #NULL! error code. NOTE: If some 'criteria' can contain

pattern characters (*?~) but you don't want to perform pattern searches, use the '&' operator to prefix that criteria with '='.

For example: "=" & CustomerID

NOTE: Unlike the quartile3_ex() function, the quartile3Ifs_ex() function always performs slower sequential searching.

If both the source and target tables contain hundreds of thousands or millions records at the same time, it might be too slow.

You can improve its performance by including the leading "=" before each criterion (to avoid pattern matching)

and by increasing the number of used processors in the 'Settings > Options' dialog box.

=quartile3Ifs_ex("orders", "Total", "id", CustomerID, 0) returns the 3rd quartile of the sales amounts for a given customer;

'orders' is a table with all orders, 'id' and 'Total' are its fields containing respectively customers' IDs and total values

for each order and 'CustomerID' is a field of the table where the result is placed.

=quartile3Ifs_ex("folder1/orders", "Total", "id", "=" & CustomerID,) returns the 3rd quartile of the sales amounts

amount for a given customer; the CustomerID may contain pattern characters but no pattern search will be performed.

=quartile3Ifs_ex("orders", "Qty", "ProductName", "Tofu", "ProductName", "Chocolade",)

```
=quartile3Iifs_ex("folder1/folder2/orders", "Total", "Date", ">2011-01-01",  
"Date", "
```

modeIifs_ex(table, calcField, field1, criteria1 [, field2, criteria2, ...], [emptyValue])

Returns the most frequently occurring value of the 'calcField' field values from the specified 'table' for records that meet the specified list of filters/criteria.

The 'table', 'calcField' and 'field(n)' parameters must be placed in quotation marks. If 'table' is located in a nested folder, its full path must be specified with the inner slashes.

References to field values/names in the same table where the formula resides must be used without any quotation marks.

The criteria can be one of the following:

(1) a name of the field in the current table, optionally combined with the =,>,>=,<,<= operators,

(2) a text string beginning with the =,>,>=,<,<= operators - numeric and date/time searches requires unformatted numbers and generic date/time strings (YYYY-MM-DD),

(3) a number or a search pattern: a text string optionally containing special characters '?' (any character) or '*' (any string). To search for ? or * place a tilde (~) before them. Numbers and dates must be used in the unformatted form.

The 'empty_value' argument specifies what should be returned (a number or a text string) if no 'v' values are found. If it's left empty, the function returns the #NULL! error code. NOTE: If some 'criteria' can contain pattern characters (*?~) but you don't want to perform pattern searches, use the '&' operator to prefix that criteria with '='. For example: "=" & CustomerID

NOTE: Unlike the mode_ex() function, the modeIifs_ex() function always performs slower sequential searching. If both the source and target tables contain hundreds of thousands or millions records at the same time, it might be too slow. You can improve its performance by including the leading "=" before each criterion (to avoid pattern matching) and by increasing the number of used processors in the 'Settings > Options' dialog box.

```
=modeIifs_ex("orders", "Total", "id", CustomerID, 1)    returns the most  
frequently occurring sale amount for a given customer;  
'orders' is a table with all orders, 'id' and 'Total' are its fields  
containing respectively customers' IDs and total values  
for each order and 'CustomerID' is a field of the table where the result is  
placed.
```

```
=modeIifs_ex("folder1/orders", "Total", "id", "=" & CustomerID, -1)    returns  
the most frequently occurring sale amount
```

for a given customer; the CustomerID may contain pattern characters but no pattern search will be performed.

```
=modeIfs_ex("orders", "Qty", "ProductName", "Tofu", "ProductName",  
"Chocolade", 0)
```

```
=modeIfs_ex("folder1/folder2/orders", "Total", "Date", ">2011-01-01", "Date",  
"
```

countIfs_ex(table, field1, criteria1 [, field2, criteria2, ...], [emptyValue])

For the specified 'table' counts records that meet the specified list of filters/criteria.

The 'table' and 'field(n)' parameters must be placed in quotation marks. If 'table' is located in a nested folder, its full path must be specified with the inner slashes.

References to field values/names in the same table where the formula resides must be used without any quotation marks.

The criteria can be one of the following:

(1) a name of the field in the current table, optionally combined with the =,>,>=,<,<= operators,

(2) a text string beginning with the =,>,>=,<,<= operators - numeric and date/time searches requires unformatted numbers and generic date/time strings (YYYY-MM-DD),

(3) a number or a search pattern: a text string optionally containing special characters '?' (any character) or '*' (any string). To search for ? or * place a tilde (~) before them. Numbers and dates must be used in the unformatted form.

The 'empty_value' argument specifies what should be returned (a number or a text string) if no 'v' values are found.

If it's left empty, the function returns the #NULL! error code. NOTE: If some 'criteria' can contain

pattern characters (*?~) but you don't want to perform pattern searches, use the '&' operator to prefix that criteria with '='.

For example: "=" & CustomerID

NOTE: Unlike the count_ex() function, the countIfs_ex() function always performs slower sequential searching.

If both the source and target tables contain hundreds of thousands or millions records at the same time, it might be too slow.

You can improve its performance by including the leading "=" before each criterion (to avoid pattern matching)

and by increasing the number of used processors in the 'Settings > Options' dialog box.

```
=countIfs_ex("orders", "id", CustomerID, 0)    returns the number of orders a  
given customer; 'orders' is a table  
with all orders, 'id' and 'Total' are its fields containing respectively  
customers' IDs and total values
```

for each order and 'CustomerID' is a field of the table where the result is placed.

=countIifs_ex("folder1/orders", "id", "=" & CustomerID, 0) returns the number of orders for a given customer;
the CustomerID may contain pattern characters but no pattern search will be performed.

=countIifs_ex("orders", "ProductName", "Tofu", "ProductName", "Chocolate", 0)

=countIifs_ex("folder1/folder2/orders", "Total", "Date", ">2011-01-01", "Date", "

vLookup_ex(table, v, searchField, returnField, [type])

Searches the 'searchField' field of the specified table for the 'v' value and - if found - returns the 'returnField' value from the found record. Save the 'v' parameter all other parameters must be placed in quotation marks.
If 'table' is located in nested folders, its full path must be specified with slashes.

The 'type' argument specifies how the searching procedure should be performed:

0 - vLookup_ex will search for an exact match; case-insensitive,

1 - if an exact match is not found, vLookup_ex will search for the largest value that is not greater than 'v',

-1 - if an exact match is not found, vLookup_ex will search for the smallest value than is not smaller than 'v',

2 - vLookup_ex will search for an exact match; 'v' can be a search pattern containing '?' (any single character) and '*' (any string); to search for '?' or '*' place a tilde (~) before them,

4 - vLookup_ex will search for an exact match; 'v' can be a regular expression; add 8 to perform case-sensitive searching; add 16 to allow empty matches.

NOTE: compared fields must be of the same type (either both text or numeric).

If 'type' is omitted, it's assumed to be 0.

If the match is not found, the function returns the #N/A! error value.

Fastest binary searches are performed for type=0, type=1 and type=-1.

=vLookup_ex("products", ProductID, "ProductID", "ProductName",)
=vLookup_ex("folder1/products", ProductID, "ProductID", "UnitPrice",)

12 Using the makeUnique() and isUnique() functions

makeUnique(table, v, field)

Checks whether the value 'v' occurs in the 'field' in the specified 'table' and if so, it creates its copy by adding the (n) suffix for text strings or adding 1 for numbers (till the unique value is found).

The 'v' can be any text, number or a text field name. The type v and the 'field' type must be the same.

```
=makeUnique("products", "tofu", "productnames") returns tofu(1)
=makeUnique("products", "tofu(1)", "productnames") returns tofu(2)
=makeUnique("products", 10, "productID") returns 78
```

isUnique(v)

Checks whether the value 'v' (a text string or a number) already occurs in current field and if so, it returns 1. Otherwise it returns 0.

NOTE: The makeUnique() and isUnique() functions are two special functions that have the following restrictions:

If the 'table' is the same table where the function is used, makeUnique() can be executed only as a single action field conversion or validation formula and is inactive for any block operations or recalculation operations.

The isUnique() function is always used in-place, in the same field it refers to and it can be executed only as a single action field conversion or validation formula.

13 Using objectStats() functions to obtain file/image data

objectStats(field, type)

objectStats(field, imageName, propertyType)

Returns the Images/Files field statistics or the metadata encoded in JPG/TIFF/PNG images (e.g. by digital cameras).

objectStats(field, type)

type = 0 - returns the number of objects

type = 1 - returns the size in bytes of the biggest object in a field

type = 2 - returns the size in bytes of the smallest object in a field

type = 3 - returns 1 if a field contains images (in format supported by GS-Base) or 0 otherwise.

To obtain the total physical size of the Images/Files fields, use the len() function.

objectStats(field, imageName, propertyType)

NOTE: this function should be used in calculated **Text** fields.

imageName - if there are multiple images in a field, you can specify a name of given image;

if you specify an empty string, the first image from a field will be used
propertyType - a numeric property code; to generate a list of all possible codes for a given image, specify -1.

```
=objectStats(files, 0)  
=objectStats(images, 1)  
=objectStats(images, "", hex2dec("132")) returns the 0x132 property value - a date  
=objectStats(images, "some_image.jpg", hex2dec("132")) returns the 0x132 property value  
for a given image  
=objectStats(images, "", -1) returns the list of all properties encoded in a given image.
```

Property values can be read for JPG/TIFF/EXIF/PNG images. A picture taken by a digital camera can contain for example the following tags:

```
ImageWidth(0x100)  
ImageHeight(0x101)  
EquipMake(0x10f)  
EquipModel(0x110)  
Orientation(0x112)  
XResolution(0x11a)  
YResolution(0x11b)  
ResolutionUnit(0x128)  
SoftwareUsed(0x131)  
DateTime(0x132)  
YCbCrPositioning(0x213)  
ExifExposureTime(0x829a)  
ExifFNumber(0x829d)  
ExifExposureProg(0x8822)  
ExifISOSpeed(0x8827)  
ExifVer(0x9000)  
ExifDTOrig(0x9003)  
ExifDTDigitized(0x9004)  
ExifCompConfig(0x9101)  
ExifShutterSpeed(0x9201)  
ExifAperture(0x9202)  
ExifBrightness(0x9203)  
ExifExposureBias(0x9204)  
ExifMaxAperture(0x9205)  
ExifSubjectDist(0x9206)  
ExifMeteringMode(0x9207)  
ExifFlash(0x9209)  
ExifFocalLength(0x920a)  
ExifMakerNote(0x927c)  
ExifDTSubsec(0x9290)  
ExifDTOrigSS(0x9291)  
ExifDTDigSS(0x9292)  
ExifFPXVer(0xa000)  
ExifColorSpace(0xa001)  
ExifPixXDim(0xa002)  
ExifPixYDim(0xa003)  
ExifSensingMethod(0xa217)  
ExifSceneType(0xa301)
```

GpsVer(0x0)
GpsLatitudeRef(0x1)
GpsLatitude(0x2)
GpsLongitudeRef(0x3)
GpsLongitude(0x4)
GpsAltitudeRef(0x5)
GpsAltitude(0x6)
GpsGpsTime(0x7)
GpsGpsDop(0xb)
ThumbnailData(0x501b)
ThumbnailImageWidth(0x5020)
ThumbnailImageHeight(0x5021)
ThumbnailCompression(0x5023)
ThumbnailOrientation(0x5029)
ThumbnailResolutionX(0x502d)
ThumbnailResolutionY(0x502e)
ThumbnailResolutionUnit(0x5030)
JPEGInterFormat(0x201)
JPEGInterLength(0x202)
ChrominanceTable(0x5091)
LuminanceTable(0x5090)
ICCProfile(0x8773)

For the full list of the possible tags please use any i-net search engine and search for the official specification the image / picture metadata tags.

14 Default field values

To specify a default value for a given Text/Numeric field

Open the Field Setup dialog box (double-clicking that field in the Database Explorer window or using the **Field Setup** command), select the **Special > Default value** option and enter the desirable value which can be:

- For numeric fields: any unformatted number.
- For text fields: any text string.
If that text field is formatted as "Date", the default value should represent the generic date/time string: YYYY-MM-DD.

To insert the default field value

Select a range of fields and/or records and use the **Insert > Default Value (Ctrl+T)** command. Default values are inserted into all empty fields within that selection.

15 Inserting series

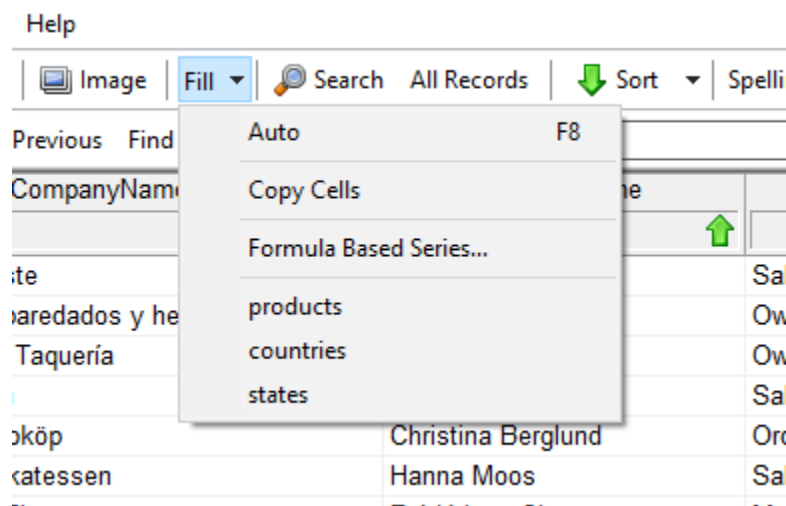
Inserting automatic series (F8)

This will fill a given range of fields with values that can be numbers, dates/times, built-in lists (day/month names) or custom lists.

The type of the series is determined automatically by the contents of the first field in that range. The "step" (for numbers, dates/times) is determined automatically by the value of the next field in that range and if the next field is empty or contains incorrect data types, a unit step is used (1, one day, one hour).

If the contents of the first field doesn't match any of the above series types, a simple "copy" operation is performed.

- Built-in list (day/month names) items can be substrings (words) within text strings in fields. For example, if the first field contains "departure on Monday", the rest of the inserted series will be in the same form: "departure on Tuesday" etc.
- Custom list items can also be substrings (words) within a field, but to be recognized, instead of the "F8" command such series must be used explicitly by its name, clicking the "Fill" button.

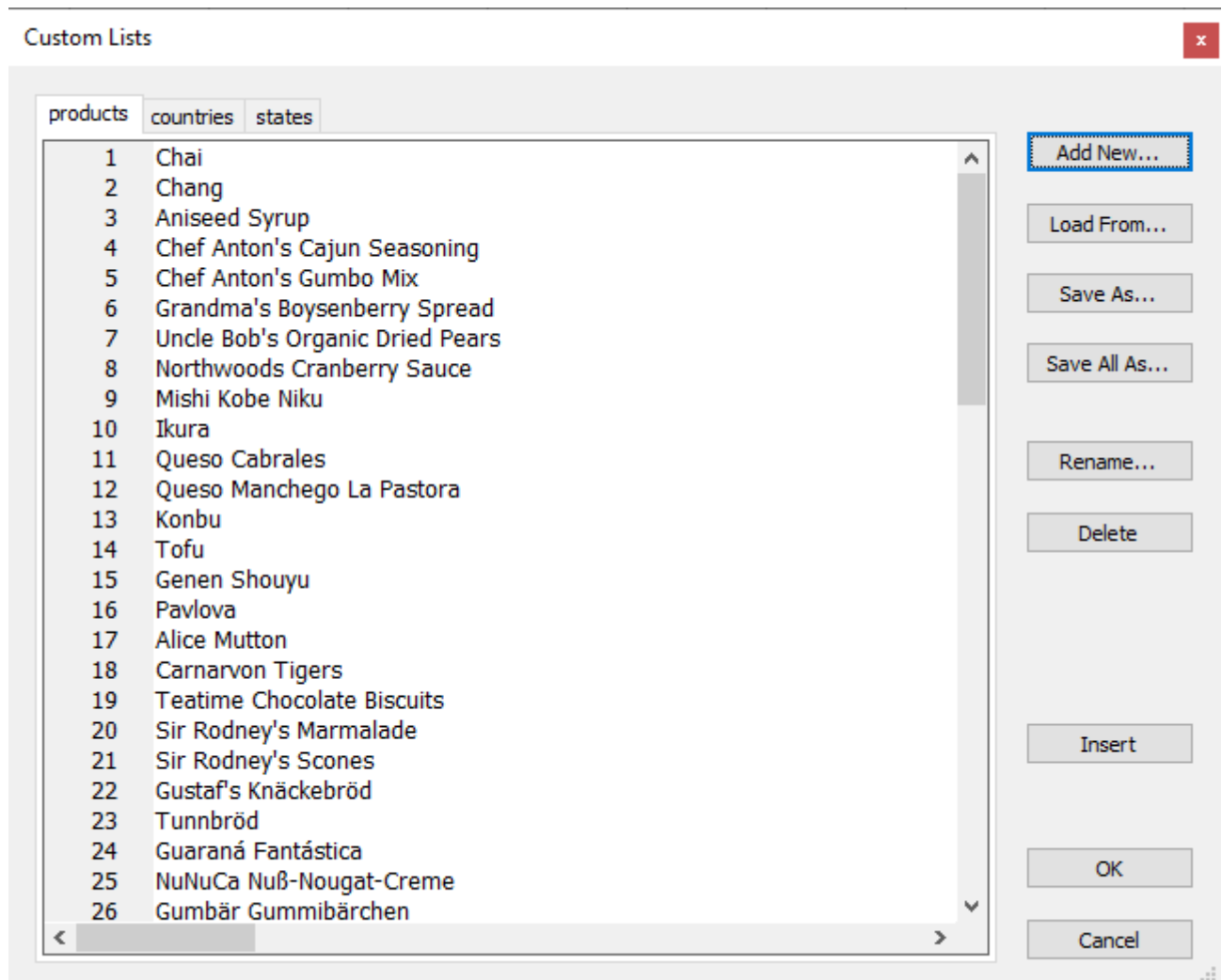


Choosing a specific custom list to insert series.

Click the "Fill" toolbar button and choose the desirable list or use the "Manage" dialog box and use the "Insert" button.

If you choose a specific custom list by name:

- Custom list items can also be substrings (words) within a field, but to be recognized, instead of the "F8" command such series must be used explicitly by its name, clicking the "Fill" button.
- The selected range of fields doesn't have to contain the initial series item. If it's empty, the series will simply start from the 1st list item.



Inserting formula-based series

This enables you to define any type of data series including those referring to resources in other tables.

For this series type you need to define a formula that will be calculated for each record within the selected range.

Used formulas are add to the "recently used" list which is stored in the settings file.

? Next Previous Find All Replace with ? Repl

| D | CompanyName | ContactName | ContactTitle |
|---|--------------------------------------|--------------------|-----------------------|
| | | | |
| | Alfreds Futterkiste | Maria Anders | Sales Representative |
| | Ana Trujillo Emparedados y helados | Ana Trujillo | Owner |
| | Antonio Moreno Taquería | Antonio Moreno | Owner |
| | Around the Horn | Thomas Hardy | Sales Representative |
| | Berglunds snabbköp | Christina Berglund | Order Administrator |
| | Blauer See Delikatessen | | Sales Representative |
| | Blondel père et fils | | Marketing Manager |
| | Bólido Comidas típicas | | Owner |
| | Bon app' | | Owner |
| | Bottom-Dollar Markets | | Accounting Manager |
| | B's Beverages | | Sales Representative |
| | Cactus Comidas para llevar | | Sales Agent |
| | Centro comercial Antojitos | | Marketing Manager |
| | Chop-suey Chinese Restaurant | | Owner |
| | Comércio Mineiro | | Sales Associate |
| | Consolidated Holdings | | Sales Representative |
| | Die Wandernde Kuh | | Sales Representative |
| | Drachenblut Delikatessen | | Order Administrator |
| | Du monde entier | | Owner |
| | Eastern Connection | Ann Devon | Sales Agent |
| | Ernst Handel | Roland Mendel | Sales Manager |
| | Familia Arquibaldo | Aria Cruz | Marketing Assistant |
| | FISSA Fabrica Inter. Salchichas S.A. | Diego Roel | Accounting Manager |
| | Folies gourmandes | Martine Rancé | Assistant Sales Agent |

Formula Based Series

Enter a formula that will be calculated for each of the selected records to generate subsequent series items. Formulas can access field values within a given record and can use functions accessing whole tables.

Field: Fields ? Recent

=proper(companyname)

OK Cancel

Inserting random series

You can use several popular distribution types. Additionally, for the uniform distribution you can choose to create a series of items randomly selected from a given custom list.

Insert Random Data

☒ **Uniform distribution**

☒ From: To:

☐ List:

☐ **Normal distribution**

Mean: Std.dev.:

16 Entering data: Drop-down lists

You can use the **Drop-Down Lists** dialog box to enable displaying drop-down lists for the current field(s) and/or to manage (create, edit, delete) lists in the current database.

When editing a field with a drop-down list attached to it, GS-Base opens a windows with a list of values with checkboxes and you can quickly choose predefined field values instead of re-typing them.



Pressing the cursor keys **Up** (in the first line of that edited field) or **Down** (in the last line of that edited field) switches the "focus" to the displayed list window. To switch back to the edited field, scroll the list to its first item and press **Up** or simply click the edited field.

To scroll the list, use the standard cursor keys: **Up**, **Down**, **PgUp**, **PgDn**, **Home**, **End**. If a list has multiple columns you can also use **Left**, **Right** to jump between columns.

To (un)check a given list item, press **Space** or click the check-box. If the "multi-selection" option is disabled for a list, the check-box button of the current list item is always "on" and checking another one always automatically unchecks the previous one.

To accept selected list item(s), press Enter, double click (one of) the selected list item(s) or click the "OK" button displayed above the edited field.

You can display list items in **1 to 99 columns**. Dragging the vertical grid-lines in the list window resizes all these list columns. To resize the list window, drag its bottom-right corner or its edges.

| | | | | | |
|----|--------|---|---|---|-------------------------|
| 3 | ANTON | Antonio Moreno Taquería | Antonio Moreno | Owner | Mataderos 2312 |
| 4 | AROUT | Around the Horn   | Thomas Hardy | Sales Representative | 120 Hanover Sq. |
| 5 | BERGS | Berglunds snabbköp | Christina Berglund | Order Administrator | Berguvsvägen 8 |
| 6 | BLONP | | | | |
| 7 | BOLID | <input type="checkbox"/> Alfreds Futterkiste | <input type="checkbox"/> Chop-suey Chinese | <input type="checkbox"/> France restauration | |
| 8 | BONAP | <input type="checkbox"/> Ana Trujillo Emparedados y helados | <input type="checkbox"/> Comércio Mineiro | <input type="checkbox"/> Franchi S.p.A. | |
| 9 | BOTTM | <input type="checkbox"/> Antonio Moreno Taquería | <input type="checkbox"/> Consolidated Holdings | <input type="checkbox"/> Frankenversand | |
| 10 | BSBEV | <input type="checkbox"/> Around the Horn | <input type="checkbox"/> Die Wandernde Kuh | <input type="checkbox"/> Furia Bacalhau e Frutos do | |
| 11 | CACTU | <input type="checkbox"/> Berglunds snabbköp | <input type="checkbox"/> Drachenblut Delikatessen | <input type="checkbox"/> Galería del gastrónomo | |
| 12 | BLAUS | <input checked="" type="checkbox"/> Blondel père et fils | <input type="checkbox"/> Du monde entier | <input type="checkbox"/> Godos Cocina Típica | |
| 13 | CENTC | <input type="checkbox"/> Bólide Comidas preparadas | <input type="checkbox"/> Eastern Connection | <input type="checkbox"/> Gourmet Lanchonetes | |
| 14 | CHOPS | <input type="checkbox"/> Bon app' | <input type="checkbox"/> Ernst Handel | <input type="checkbox"/> Great Lakes Food Market | |
| 15 | COMMI | <input type="checkbox"/> Bottom-Dollar Markets | <input type="checkbox"/> Familia Arquibaldo | <input type="checkbox"/> GROSELLA-Restaurante | |
| 16 | CONSH | <input type="checkbox"/> B's Beverages | <input type="checkbox"/> FISSA Fabrica Inter. Salchichas S.A. | <input type="checkbox"/> Hanari Carnes | |
| 17 | WANDK | <input type="checkbox"/> Cactus Comidas para llevar | <input type="checkbox"/> Folies gourmandes | <input type="checkbox"/> HILARIÓN-Abastos | stages |
| 18 | DRACD | <input type="checkbox"/> Centro comercial Moctezuma | <input type="checkbox"/> Folk och få HB | <input type="checkbox"/> Hungry Coyote Import Store | |
| 19 | DUMON | | | | |
| 20 | EASTC | | | | |
| 21 | ERNSH | | | | |
| 22 | FAMIA | | | | |
| 23 | FISSA | FISSA Fabrica Inter. Salchichas S.A. | Diego Roel | Accounting Manager | C/ Moralzarzal, 86 |
| 24 | FOLIIG | Folies gourmandes | Martina Rancé | Assistant Sales Agent | 184 chaussée de Tournai |

To create a new list, click the **New List** button and to add new list items, enter them in the **List Items** edit field: one item in one line.

You can also choose to load list items automatically from a record field from any table in the same database file. GS-Base will load all current unique values from that field. The maximum number of list items is the same as the maximum number of records: 256 millions.

If you select the **AutoAppend** option, each new unique value entered in the corresponding field(s) will be added to the list automatically. This option is disabled if the list is loaded from a database field

If you choose the **AutoComplete** option, the edited cell contents will be automatically completed/replaced with the full found matching text from the list after you type the initial characters.

If the **Sort items** option is not selected, the list will display all its elements in the same order as they were added/entered. Otherwise the list items will be sorted before displaying.

Format Field [X]

Font Style Alignment **Drop-down lists**

Drop-down list: Sample List 2. [v] [New List]

☐ Items from field: customers - CustomerID [v] [Rename]

☒ Items as text: [v] [Remove]

London
 Buenos Aires
 Mannheim
 México D.F.
 Bern
 São Paulo
 London
 Stuttgart
 Aachen
 Nantes
 London
 Graz

Columns: 2 [v]

☒ Auto-Append
 ☒ Sort items
 ☒ Auto-Complete

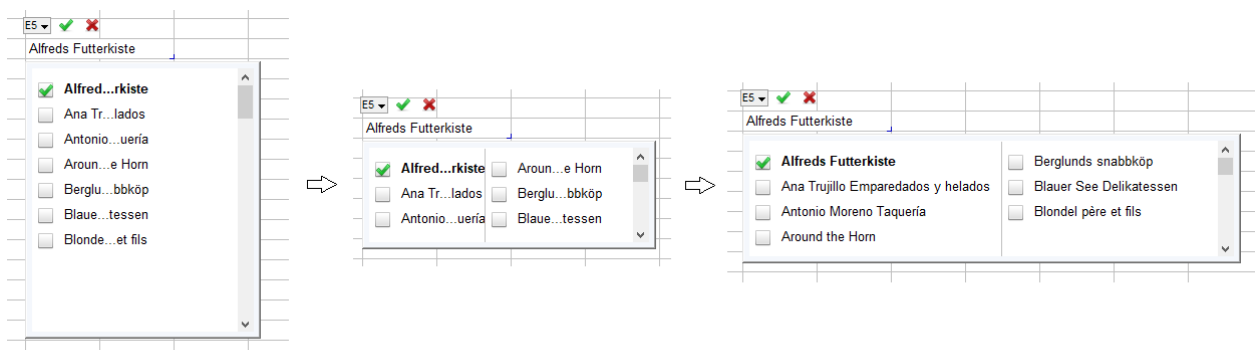
☒ Multiple selection
 Separator: comma [v] ,

[OK] [Cancel] [Help]

One list can be used by any number of fields in any tables in a given database file. You can create up to 100 list in one database file. To move lists between workbooks, you have to copy/paste the data from the **List Items** edit field.

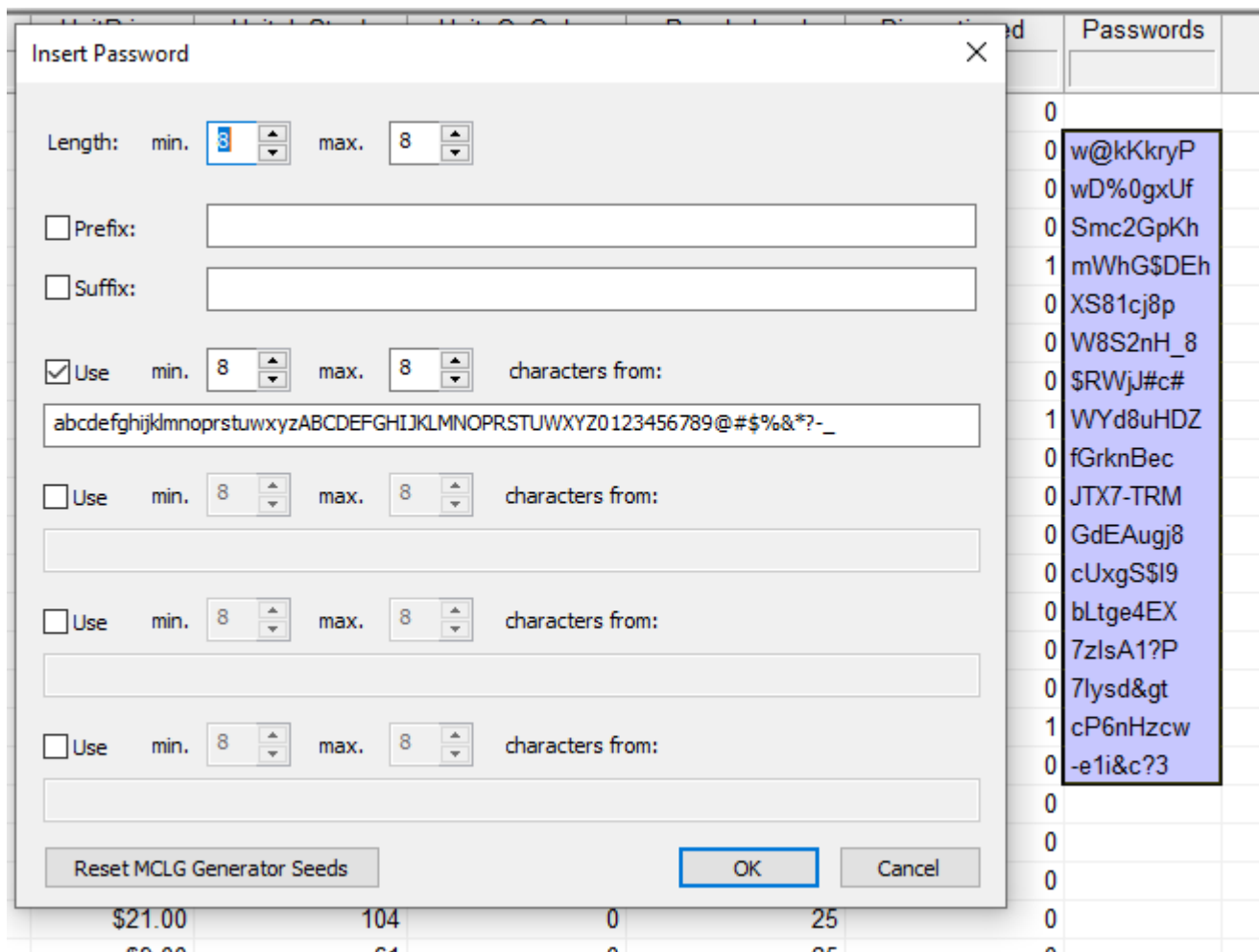
Note:

If you specify the "columns" to be > 1, the splitter lines are displayed only if the list items occupy more than one column. Thus, to resize columns you might need to temporarily shrink the list window, for example:



17 Generating passwords

The "Insert > Password" command enables you to insert unique passwords (either as a series of any number of unique passwords, filling in the currently selected range of table fields or as single unique strings subsequently used in edited text/code/memo fields). There are several options that helps you to increase the probability that the generated passwords are unique. For typical settings, 8-character passwords should be unique within one to several millions generated case-sensitive passwords of one series which is continuously generated with each use of this command till you use the "Reset MCLG Generator Seeds" button. You can use the "Tools > Find Duplicates" command to verify the uniqueness of these passwords. Generated passwords can contain Unicode characters but non-ascii characters should be used with care as some systems (especially various recovery systems) might not support them.



18 Encrypting field contents

To password-protect and encrypt only a part of your database, selected record fields, use the **Edit > Encrypt/Decrypt Field Contents** commands.

(If there are no fields to decrypt, the **Decrypt** command is inactive. If there are already encrypted fields in a given table, the **Encrypt** command is inactive.)

In comparison with the file encryption there are several benefits of using such partial encryption, for example:

- you can publish your database for viewing and updating of the unencrypted (e.g. non-sensitive) parts of the database tables by others and you can be sure that the encrypted table fields can't be changed, misplaced or deleted within the table in any way;
- for any file with partially encrypted data you can still open it without a password to verify the file version, what the table and field names are etc.

To encrypt the data, GS-Base uses the standard Twofish block cipher algorithm which guarantees that it can't be broken and your data can't be access/read without knowing your password.

GS-Base displays in encrypted fields the actual encrypted data in the mime64 encoding form.

For encrypted text fields, the displayed encrypted contents length is proportional to the length of the source text strings.

If you would like to protect this information as well, you can add trailing spaces to these source strings before encrypting.

For encrypted numeric fields, the displayed encrypted values have the same length.

For encrypted LongText/Code/Images/Files fields, the entire contents is encrypted and the "encrypted" label is displayed in table/form cells.

The encrypted part of a given database table can't be modified - GS-Base disables any editing actions for the encrypted fields. If you delete records with encrypted data or insert new records before/between records with encrypted data, GS-Base won't let you save such a modified file. Of course, you're free to edit/delete and save any unencrypted fields and append new records (leaving the encrypted fields empty).

If a given table contains calculated fields, after encrypting one or more of its fields the last calculation results become "frozen" till you decrypt these fields.

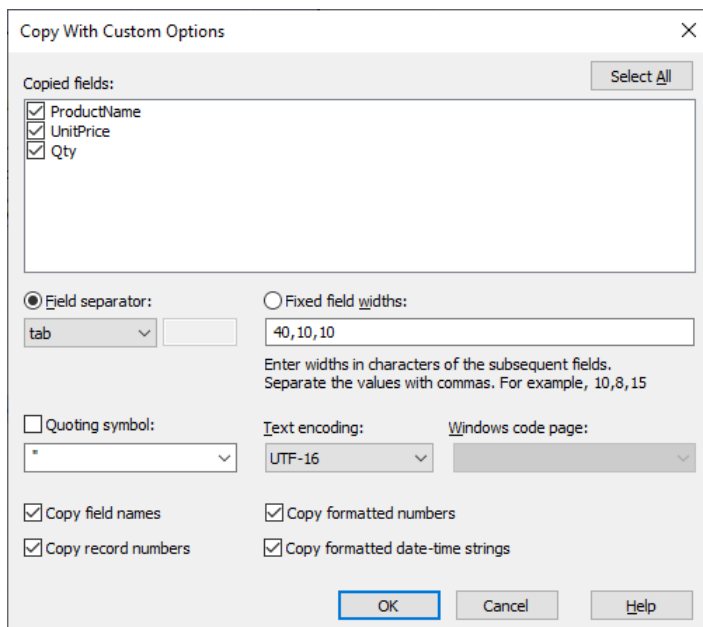
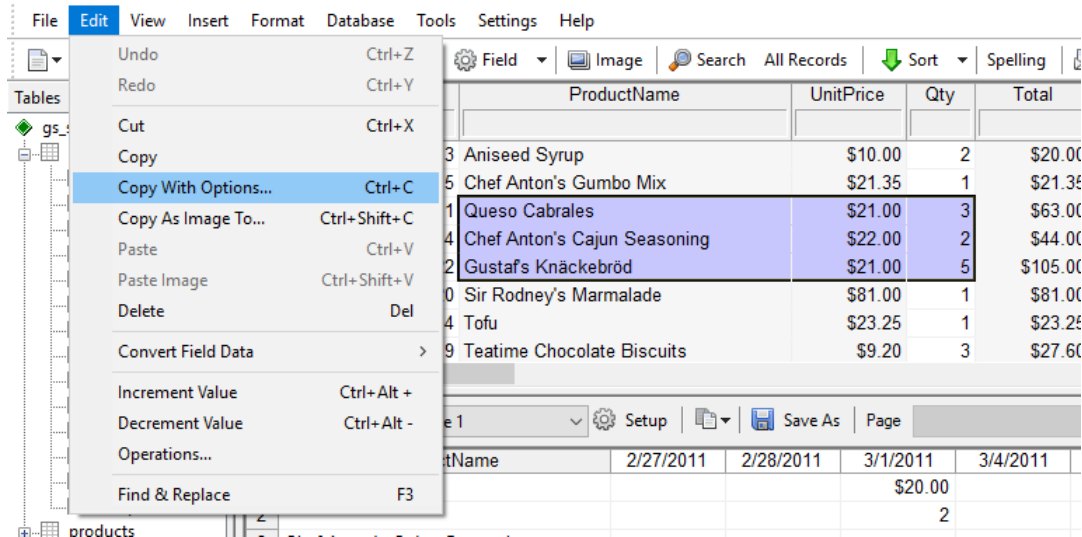
Field encryption can be applied independently to each table, so you can use a different password for each table.

Having some encrypted fields, you can still use file encryption.

19 Using the Copy-With-Options command

The **Copy With Options** command has been added as an alternative to the "Copy" command to enable more flexible copying (of both tables and forms) from GS-Base to other programs. The standard "Ctrl+C" shortcut is assigned interchangeably to one of these commands.

Sample screens with various copy settings and the resulting output in a spreadsheet and a text processor:



| | ProductName | UnitPrice | Qty |
|---|------------------------------|-----------|-----|
| 3 | Queso Cabrales | \$21.00 | 3 |
| 4 | Chef Anton's Cajun Seasoning | \$22.00 | 2 |
| 5 | Gustafs Knäckebröd | \$21.00 | 5 |

Copy With Custom Options

Copied fields: Select All

- ☒ ProductName
- ☒ UnitPrice
- ☒ Qty

☐ Field separator: (none) ☒ Fixed field widths: 40,10,10

Enter widths in characters of the subsequent fields. Separate the values with commas. For example, 10,8,15

☐ Quoting symbol: * Text encoding: UTF-16 Windows code page:

☒ Copy field names ☒ Copy formatted numbers

☒ Copy record numbers ☒ Copy formatted date-time strings

OK Cancel Help

New Text Document - Notepad

| | ProductName | UnitPrice | Qty |
|---|------------------------------|-----------|-----|
| 3 | Queso Cabrales | \$21.00 | 3 |
| 4 | Chef Anton's Cajun Seasoning | \$22.00 | 2 |
| 5 | Gustaf's Knäckebröd | \$21.00 | 5 |

File Edit View Insert Format Database Tools Settings Help

gs_sample2b customers

ProductID: 1
ProductName: Chai
SupplierID: 1
CategoryID: 1
QuantityPerUnit: 10 boxes
UnitPrice: \$18.00
UnitsInStock: 39
UnitsOnOrder: 0
ReorderLevel: 10
Discontinued: 0

Copy With Custom Options

Copied fields: Select All

- ☒ ProductID
- ☒ ProductName
- ☒ SupplierID
- ☒ CategoryID
- ☒ QuantityPerUnit
- ☒ UnitPrice
- ☒ UnitsInStock
- ☒ UnitsOnOrder
- ☒ ReorderLevel
- ☐ Discontinued

☐ Field separator: tab ☒ Fixed field widths: 20,35

Enter widths in characters of the subsequent fields. Separate the values with commas. For example, 10,8,15

☐ Quoting symbol: * Text encoding: UTF-16 Windows code page:

☒ Copy field names ☒ Copy formatted numbers

☒ Copy record numbers ☒ Copy formatted date-time strings

OK Cancel Help

| | |
|------------------|--------------------|
| Record: | 1 |
| ProductName: | Chai |
| SupplierID: | 1 |
| CategoryID: | 1 |
| QuantityPerUnit: | 10 boxes x 20 bags |
| UnitPrice: | \$18.00 |
| UnitsInStock: | 39 |
| UnitsOnOrder: | 0 |
| ReorderLevel: | 10 |

File Edit View Insert Format Database Tools Settings Help

gs_sample2b customers

ProductID: 1
ProductName: Chai
SupplierID: 1
CategoryID: 1
QuantityPerUnit: 10 boxes
UnitPrice: \$18.00
UnitsInStock: 39
UnitsOnOrder: 0
ReorderLevel: 10
Discontinued: 0

Copy With Custom Options

Copied fields: Select All

- ☒ ProductID
- ☒ ProductName
- ☒ SupplierID
- ☒ CategoryID
- ☒ QuantityPerUnit
- ☒ UnitPrice
- ☒ UnitsInStock
- ☒ UnitsOnOrder
- ☒ ReorderLevel
- ☐ Discontinued

☐ Field separator: (none) ☒ Fixed field widths: 20,35

Enter widths in characters of the subsequent fields. Separate the values with commas. For example, 10,8,15

☐ Quoting symbol: * Text encoding: UTF-16 Windows code page:

☒ Copy field names ☒ Copy formatted numbers

☒ Copy record numbers ☒ Copy formatted date-time strings

OK Cancel Help

```
New Text Document - Notepad
File Edit Format View Help
Record: 1

ProductName:      Chai
SupplierID:       1
CategoryID:       1
QuantityPerUnit:  10 boxes x 20 bags
UnitPrice:        $18.00
UnitsInStock:     39
UnitsOnOrder:     0
ReorderLevel:     10
```

20 Using the Increment, Decrement and Operations commands

To add/multiple/divide/subtract values in selected fields automatically

Select a numeric or date/time (a text field with the date/time format) field value or any group of such values within a table and use the **Edit > Increment (Ctrl+Alt+'+')** or **Edit > Decrement (Ctrl+Alt+'-')** command. By default, for numeric fields these commands respectively add 1 and subtract 1. For textual date values they add and subtract one day and for textual time they add and subtract one hour.

The above plain adding and subtracting can be overwritten with the **Edit > Operations** command to perform more complex field value changes.

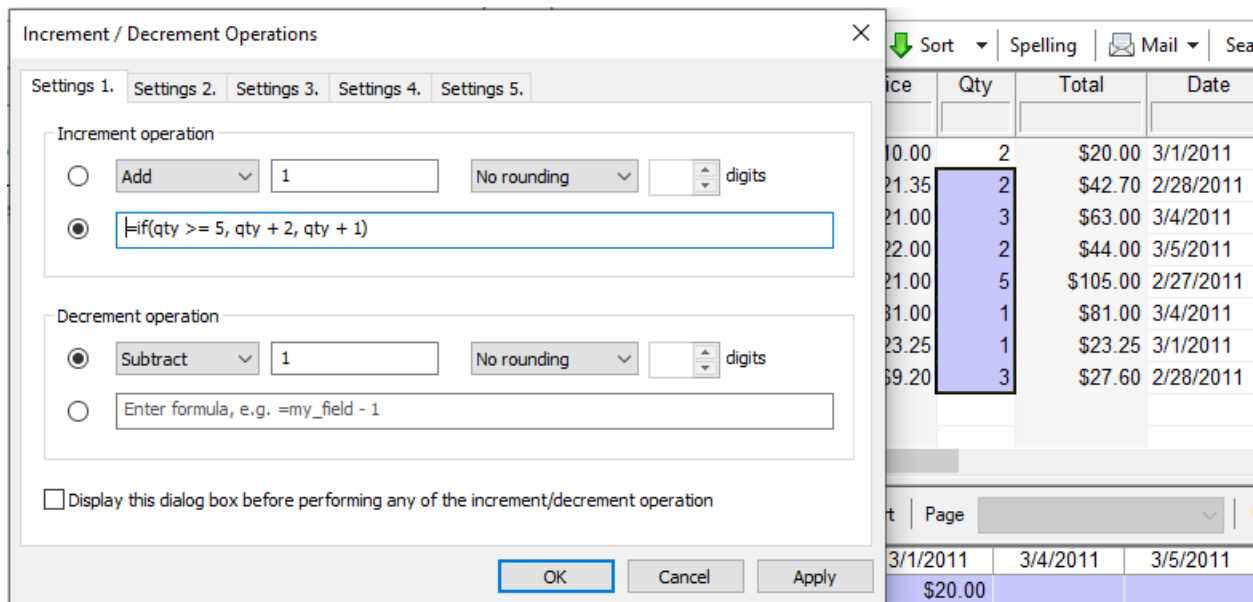
To define operations executed when you use the "Edit > Increment" and "Edit > Decrement" commands

Use the **Edit > Operations** command and specify the incrementing and decrementing operations, which can be adding/subtracting/dividing/multiplying with a fixed value and with a given precision. The operation can be also defined as a formula referring to a given field (or fields).

There are five increment/decrement different pair settings and the active one is determined by the active tab in the **Operations** dialog box.

For example:

1. Define some 'increment' formula-based action on the 'Settings1' tab and make it active.



2. Use the 'increment' command (Ctrl+Alt+'+'). Values in the selected fields will be modified according to the defined formula:

| ds | Sort | Spelling | Mail | Search |
|-----------|------|----------|-----------|--------|
| UnitPrice | Qty | Total | Date | |
| \$10.00 | 2 | \$20.00 | 3/1/2011 | |
| \$21.35 | 3 | \$64.05 | 2/28/2011 | |
| \$21.00 | 4 | \$84.00 | 3/4/2011 | |
| \$22.00 | 3 | \$66.00 | 3/5/2011 | |
| \$21.00 | 7 | \$147.00 | 2/27/2011 | |
| \$81.00 | 2 | \$162.00 | 3/4/2011 | |
| \$23.25 | 2 | \$46.50 | 3/1/2011 | |
| \$9.20 | 4 | \$36.80 | 2/28/2011 | |

21 Dual Views

Use the **View > Dual View** command split the main GS-Base window as two panes that display two database views independently. This enables you to browse and edit two different tables at the same time.

The screenshot shows the GS-Base 16.1 application window with a dual view. The left pane displays a list of customers, and the right pane displays a pivot table for the 'orders' table. The pivot table shows product names and their quantities and prices for different dates.

| ProductID | ProductName | UnitPrice |
|-----------|-------------|-----------|
| 1 | BOLID | \$2 |
| 2 | BOLID | \$2 |
| 3 | BOLID | \$2 |

Pivot Table: Pivot table 1

| ProductName | 2/27/2011 | 2/28/2011 | 3/1/2011 |
|------------------------------|-----------|-----------|----------|
| Aniseed Syrup | | | \$1 |
| Chef Anton's Cajun Seasoning | | | |
| Chef Anton's Gumbo Mix | | \$21.35 | |
| Gustaf's Knäckebröd | \$105.00 | 5 | |
| Queso Cabrales | | | |
| Sir Rodney's Marmalade | | | |
| Teatime Chocolate Biscuits | | \$27.60 | |
| Tofu | | 3 | \$1 |
| Grand Total | \$105.00 | \$48.95 | \$1 |

22 Linking tables in dual views by the 1-n relation

To create a link (the one-to-many relation) between two fields in different tables

1. Use the **View > Dual View** command to open the database in two panes: the left and right panes.
2. With the current table as the "master" table in the left pane, select a desirable "slave/search" table in the right pane, then place the current selections in fields (columns) that are to be linked.
3. Use the **Tools > Set Relation (Ctrl+=)** command to turn the relation on and off.

The linked fields are displayed with the highlighted background color. Scrolling to another record in the "master" table in the left pane causes automatic filtering of the "search" table to find matching values in the linked field in the right pane.

Closing the dual view turns off the active relation.

Executing another the **Tools > Set Relation (Ctrl+=)** command turns off the active relation, no matter which tables are currently selected in the dual view.

23 Merging and joining tables

Merging/Unmerging records

To merge (add) or "unmerge" (mass-delete) records from other databases, use the **Tools > Merge/Unmerge** command. The display dialog box enables you to specify how adding or removing records should be carried out:

Merge/Unmerge Records

File:

Table:

☒ Add all records from the specified table

☐ Add records where
 doesn't exist in the current table

☐ Update records where
 =

☐ Don't import empty field values

☐ Delete records where
 =

☒ Enable undo for this operation

Options 1. - 3. perform field name matching in the current and the merged tables. You can bypass this using this command in the form of a scripts - please see: ...

Options 4. causes deleting records from the current tables where the specified field values match.

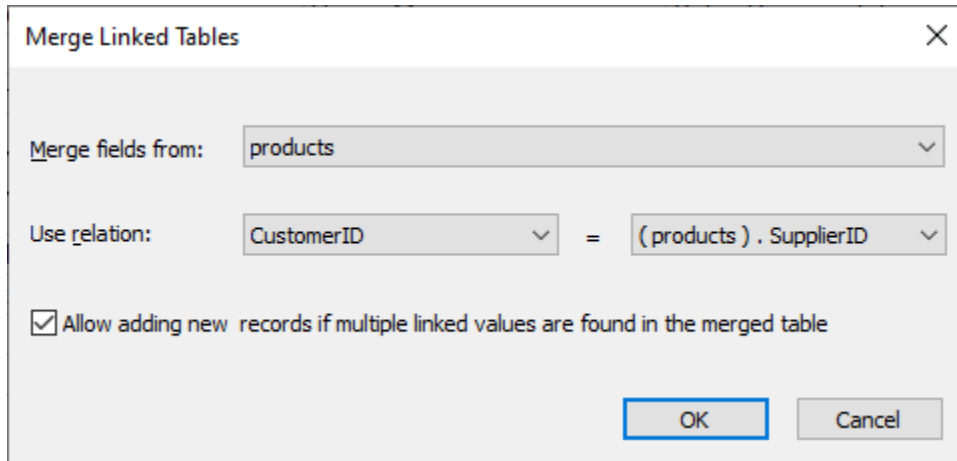
Note: to perform mass-merging tables from a given folder or multiple folders, use a script as shown in the "Merging" scripting examples further in this manual.

Joining column from other tables

The **Tools > Merge Linked Tables** command enables you to add (to the current table) columns/fields from different tables in the same database.

For the current table and for the merged table you need to specify a field that contain values/IDs that link records in both these tables, like "ProductID" fields in the "products" and "orders" tables in the included **sample.zip** database.

Linking fields don't have to use the same name, but must be of the same type, either **Text** or **Number**.



24 Spell-checking and adding Hunspell dictionaries

Spell-checking availability

GS-Base 64-bit can use either the system spell checking functionality provided by Windows 8.x and later or the Hunspell spell checking engine (available for all systems).

GS-Base 32-bit can use Hunspell spell checking only.

The current selection can be changed either in the **Settings > Options > Spelling** dialog box or directly in the **Tools > Spelling** dialog box.

Adding Hunspell dictionaries

To install a Hunspell dictionary for a given language:

1. Go to one of the web sites which distributes Hunspell dictionaries, for example:

<https://addons.mozilla.org/en-US/firefox/language-tools/>

The dictionaries are distributed as zip files typically with *.xpi or *.oxl extensions. After downloading the desirable language file, rename it to a *.zip file.

2. Each downloaded dictionary archive contains two text files with the *.aff and *.dic extensions and the name that (usually) represents the corresponding language ISO codes, e.g. en-US.aff and en-US.dic (English US).

Copy those files to some folder and specify that folder in the GS-Base spelling options. GS-Base will automatically find all installed Hunspell dictionaries and list them in the **Spelling** dialog box next time you execute the **Tools > Spelling** command.

To remove a given Hunspell dictionary, delete its *.aff and *.dic files from the above folder.

Adding custom words to dictionaries

Users can add their own words to existing dictionaries (or - when using the Hunspell spell checking engine - create their own full dictionaries).

The added words are saved in a text file in a folder specified in the GS-Base spelling options. The file extension is *.txt and the file name is the language ISO code corresponding to the dictionary the file is link to. For example, if the dictionary files are

en-US.aff and en-US.dic

then the custom words will automatically (after closing the dialog box) be saved to

en-US.txt .

New words are added after clicking the **Add Word** button when performing spell checking. The *.txt files can be also edited in any text editor to add some larger number of words at once.

The text files containing added words must be saved using the UTF-8 encoding.

Other ISO codes include:

- af-ZA
- sq-AL
- ar-AE
- ar-BH
- ar-DZ
- ar-EG
- ar-IQ
- ar-JO
- ar-KW
- ar-LB
- ar-LY
- ar-MA
- ar-OM
- ar-QA
- be-BY
- bg-BG
- zh-CN
- zh-HK

- zh-MO
- zh-SG
- zh-TW
- hr-HR
- cs-CZ
- da-DK
- nl-NL
- nl-BE
- en-AU
- en-BZ
- en-CA
- en-CB
- en-IE
- en-JM
- en-NZ
- en-PH
- en-ZA
- en-US
- en-GB
- et-EE
- fi-FI
- fr-FR
- fr-BE
- fr-CA
- fr-LU
- fr-CH
- gd-IE
- de-DE
- de-AT
- de-LI
- de-LU

- de-CH
- el-GR
- he-IL
- hi-IN
- hu-HU
- is-IS
- id-ID
- it-IT
- it-CH
- ja-JP
- ko-KR
- lv-LV
- lt-LT
- mk-MK
- ms-MY
- no-NO
- pl-PL
- pt-PT
- pt-BR
- ro-RO
- ru-RU
- sr-SR
- sl-SI
- sk-SK
- es-ES
- es-AR
- es-BO
- es-CO
- es-CR
- es-DO
- es-EC

- es-GT
- es-HN
- es-MX
- es-NI
- es-PA
- es-PE
- es-PR
- es-PY
- es-UY
- es-VE
- sv-SE
- sv-FI
- th-VE
- tr-VE
- uk-UA
- uz-UZ
- vi-VN
- yi-IL
- uk-UA

25 Using Pivot Tables

Pivot tables enable you to quickly and easily obtain and visualize various statistical information about the data stored in flat source record tables. This includes (but is not limited to) counting, summarizing, sorting or finding partial maximum/minimum and mean values.

In GS-Base you can create up to 100 pivot tables for each database table. To create a new pivot table, use the **View > Pivot Table View** command and in the opened pivot table pane click **Pivot Table > Add New Pivot Table** or - if this is the first pivot table - simply click the **Setup** button:

Pivot Table Setup

Table name: Pivot table 1

ID
 ProductID
 UnitPrice
 Qty
 Total
 OrderID

Row fields
 Add >
 < Remove
 ProductName
 Sort ascending

Column fields
 Add >
 < Remove
 Date
 Sort ascending

Data fields
 Add >
 < Remove
 Sum of Total
 Sum of Qty
 Sum

Grand Totals in columns and rows
☐ Page field: ID

Display unavailable data as:
 #N/A!

☒ Subtotals
☐ Repeat rows fields
☐ Empty fields as zeroes

☒ Ignore punctuation
☐ Case sensitive

OK Cancel Help

Quick start

Open the included sample "sample.zip" database and choose the "customers" table. The records contain (among others) the "Country" and "City" fields.

To obtain the number of customers from particular countries:

1. Open the **Pivot Table Setup** dialog.
2. Choose the "Country" database field as the only **Row field**.
3. click **OK**.

| | Country | Grand Total |
|----|--------------------|-------------|
| 1 | Argentina | 3 |
| 2 | Austria | 2 |
| 3 | Belgium | 2 |
| 4 | Brazil | 9 |
| 5 | Canada | 3 |
| 6 | Denmark | 2 |
| 7 | Finland | 2 |
| 8 | France | 11 |
| ⋮ | | |
| 22 | Grand Total | 91 |

To obtain the number of customers from particular countries and to find out how many of them come from given cities in that country:

1. Open the **Pivot Table Setup** dialog.
2. Choose the "Country" database field as the first **Row field**.
3. Choose the "City" database field as the second **Row field**.
4. click **OK**.

| | Country | City | Grand Total |
|----|--------------------|--------------|-------------|
| 1 | Argentina | Buenos Aires | 3 |
| 2 | Argentina Total | | 3 |
| 3 | Austria | Graz | 1 |
| 4 | | Salzburg | 1 |
| 5 | Austria Total | | 2 |
| 6 | Belgium | Bruxelles | 1 |
| 7 | | Charleroi | 1 |
| 8 | Belgium Total | | 2 |
| ⋮ | | | |
| 91 | Grand Total | | 91 |

Open the included sample "sample.zip" database and choose the "orders" table.
The records contain (among others) the "ProductName" and (order) "Date" fields.

To obtain the numbers of sold products and to find out how these numbers looked like in subsequent days:

1. Open the **Pivot Table Setup** dialog.
2. Choose the "ProductName" database field as the only **Row field**.
3. Choose the "Date" database field as the only **Column field**.

4. click **OK**.

| | ProductName | 2011-02-27 | 2011-02-28 | 2011-03-01 | 2011-03-04 | 2011-03-05 | Grand Total |
|---|------------------------------|------------|------------|------------|------------|------------|-------------|
| 1 | Aniseed Syrup | | | 1 | | | 1 |
| 2 | Chef Anton's Cajun Seasoning | | | | | 1 | 1 |
| 3 | Chef Anton's Gumbo Mix | | 1 | | | | 1 |
| 4 | Gustaf's Knäckebröd | 1 | | | | | 1 |
| 5 | Queso Cabrales | | | | 1 | | 1 |
| 6 | Sir Rodney's Marmalade | | | | 1 | | 1 |
| 7 | Teatime Chocolate Biscuits | | 1 | | | | 1 |
| 8 | Tofu | | | 1 | | | 1 |
| 9 | Grand Total | 1 | 2 | 2 | 2 | 1 | 8 |

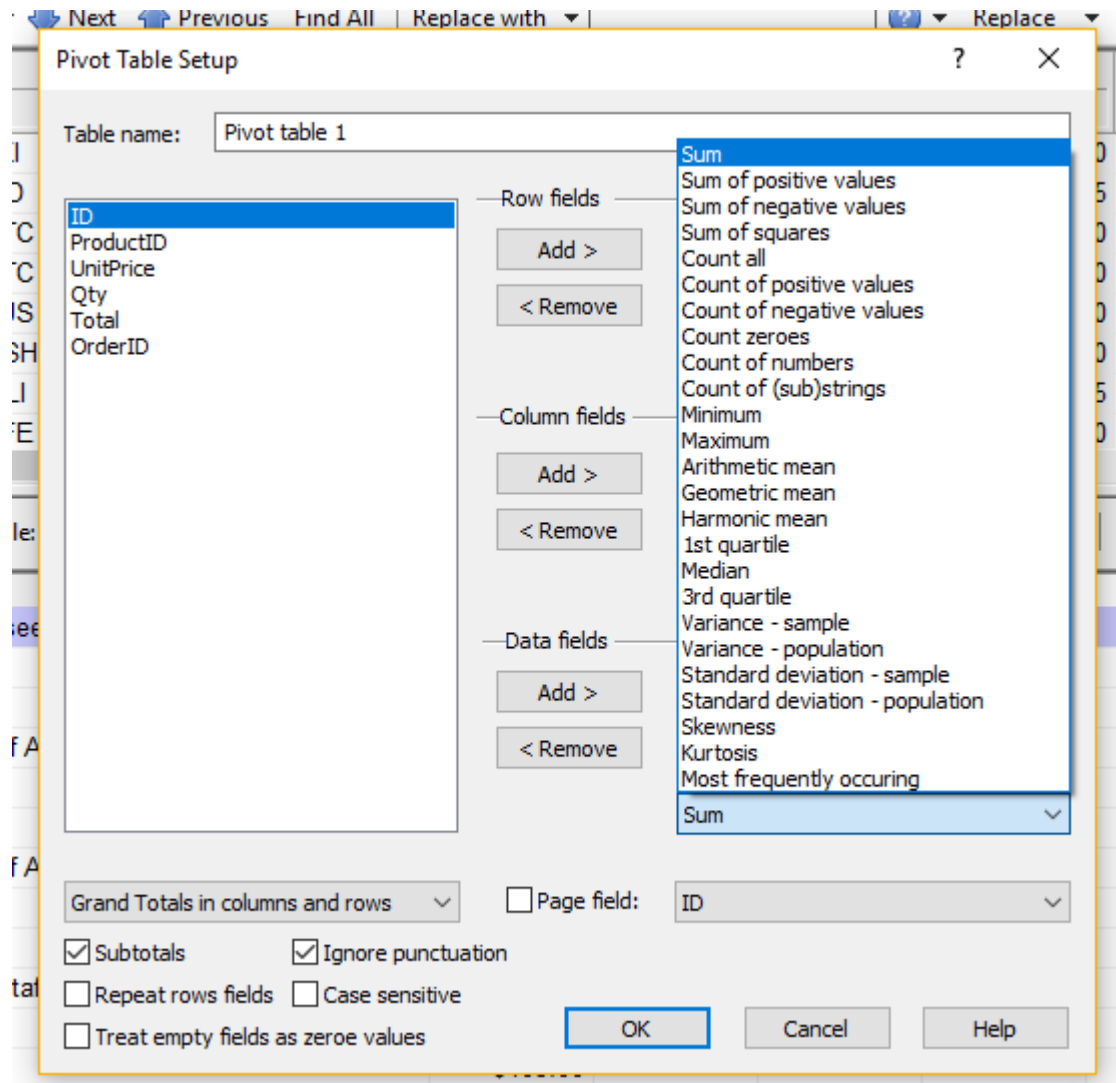
To obtain the numbers of sold products and to find out how both these numbers and the sales totals looked like in subsequent days:

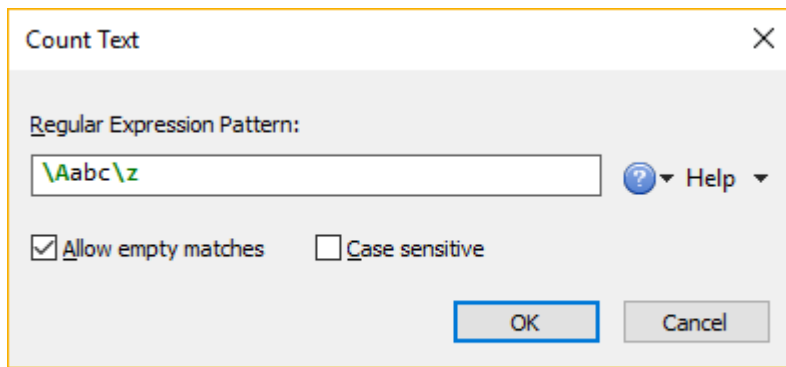
1. Open the **Pivot Table Setup** dialog.
2. Choose the "ProductName" database field as the only **Row field**.
3. Choose the "Date" database field as the only **Column field**.
4. Choose the "Total" database field as the first **Data field** and select the **Count** function for it.
5. Choose the "Qty" database field as the second **Data field** and select the **Sum** function for it.
6. click **OK**.

| | ProductName | 2011-02-27 | 2011-02-28 | 2011-03-01 | 2011-03-04 | 2011-03-05 | Grand Total |
|----|------------------------------|-----------------|----------------|----------------|-----------------|----------------|-----------------|
| 1 | Aniseed Syrup | | | \$20.00 | | | \$20.00 |
| 2 | | | | 2 | | | 2 |
| 3 | Chef Anton's Cajun Seasoning | | | | | \$44.00 | \$44.00 |
| 4 | | | | | | 2 | 2 |
| 5 | Chef Anton's Gumbo Mix | | \$21.35 | | | | \$21.35 |
| 6 | | | 1 | | | | 1 |
| 7 | Gustaf's Knäckebröd | \$105.00 | | | | | \$105.00 |
| 8 | | 5 | | | | | 5 |
| 9 | Queso Cabrales | | | | \$63.00 | | \$63.00 |
| 10 | | | | | 3 | | 3 |
| 11 | Sir Rodney's Marmalade | | | | \$81.00 | | \$81.00 |
| 12 | | | | | 1 | | 1 |
| 13 | Teatime Chocolate Biscuits | | \$27.60 | | | | \$27.60 |
| 14 | | | 3 | | | | 3 |
| 15 | Tofu | | | \$23.25 | | | \$23.25 |
| 16 | | | | 1 | | | 1 |
| 17 | Grand Total | \$105.00 | \$48.95 | \$43.25 | \$144.00 | \$44.00 | \$385.20 |
| 18 | | 5 | 4 | 3 | 4 | 2 | 18 |

26 Pivot Table Functions

GS-Base pivot tables can use of number of "counting" functions:



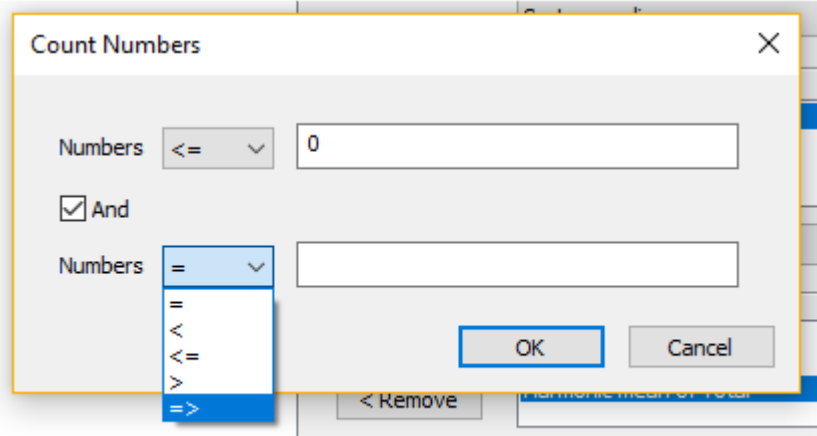


Count Text

Regular Expression Pattern:

☒ Allow empty matches ☐ Case sensitive

? Help



Count Numbers

Numbers

☒ And

Numbers

< Remove

27 Creating Pivot Table Reports

Clicking the "Save" icon in the Pivot Table View enables you to save a new time-stamp report for a given pivot table.

The procedure looks as follows:

1.

The screenshot shows a 'Save Pivot Table Report' dialog box with the following fields:

- New table name:
- Database folder:
- Buttons: OK, Cancel

Below the dialog box is a pivot table report titled 'Pivot Table 1'. The report has the following columns: ProductName, ID, 2/27/2011, 2/28/2011, 3/1/2011, 3/4/2011, 3/5/2011, and Grand Total. The data is as follows:

| | ProductName | ID | 2/27/2011 | 2/28/2011 | 3/1/2011 | 3/4/2011 | 3/5/2011 | Grand Total |
|---|------------------------------|--------|-----------|-----------|----------|----------|----------|-------------|
| 1 | Aniseed Syrup | ALFKI | | | \$20.00 | | | \$20.00 |
| 2 | | | | | 2 | | | 2 |
| 3 | | ALFKII | | | \$50.00 | | | \$50.00 |
| 4 | | | | | 5 | | | 5 |
| 5 | Chef Anton's Cajun Seasoning | EASTC | | | | | \$44.00 | \$44.00 |
| 6 | | | | | | | 2 | 2 |
| 7 | Chef Anton's Gumbo Mix | BOLID | | \$21.35 | | | | \$21.35 |
| 8 | | | | 1 | | | | 1 |
| 9 | Gustaf's Knäckebröd | BLAUS | \$105.00 | | | | | \$105.00 |

2.

The screenshot shows a database table view with the following columns: Date, ProductName, ID, Sum of Total, and Sum of Qty. The data is as follows:

| | Date | ProductName | ID | Sum of Total | Sum of Qty |
|----|------------|------------------------------|--------|--------------|------------|
| 1 | 2011-02-27 | Aniseed Syrup | ALFKI | | |
| 2 | 2011-02-27 | Aniseed Syrup | ALFKII | | |
| 3 | 2011-02-27 | Chef Anton's Cajun Seasoning | EASTC | | |
| 4 | 2011-02-27 | Chef Anton's Gumbo Mix | BOLID | | |
| 5 | 2011-02-27 | Gustaf's Knäckebröd | BLAUS | 105 | 5 |
| 6 | 2011-02-27 | Queso Cabrales | CENTC | | |
| 7 | 2011-02-27 | Sir Rodney's Marmalade | ERNSH | | |
| 8 | 2011-02-27 | Teatime Chocolate Biscuits | VAFFE | | |
| 9 | 2011-02-27 | Tofu | WELLI | | |
| 10 | 2011-02-28 | Aniseed Syrup | ALFKI | | |
| 11 | 2011-02-28 | Aniseed Syrup | ALFKII | | |
| 12 | 2011-02-28 | Chef Anton's Cajun Seasoning | EASTC | | |
| 13 | 2011-02-28 | Chef Anton's Gumbo Mix | BOLID | 21.35 | 1 |
| 14 | 2011-02-28 | Gustaf's Knäckebröd | BLAUS | | |
| 15 | 2011-02-28 | Queso Cabrales | CENTC | | |
| 16 | 2011-02-28 | Sir Rodney's Marmalade | ERNSH | | |
| 17 | 2011-02-28 | Teatime Chocolate Biscuits | VAFFE | 27.6 | 3 |
| 18 | 2011-02-28 | Tofu | WELLI | | |

You can also simply copy the pivot table data to other programs either as image or using a few "Copy" commands from the context menu:

| 8 | VAFFE | 19 | Teatime Chocolate Biscuits | \$9.20 | 3 | \$27.60 | 2/28/2011 |
|--|------------------------------|-----------|----------------------------|----------|----------|----------|-------------|
| | | | | | | | |
| Pivot Table: Pivot table 1 Setup Save As Page Update | | | | | | | |
| | ProductName | 2/27/2011 | 2/28/2011 | 3/1/2011 | 3/4/2011 | 3/5/2011 | Grand Total |
| 1 | Aniseed Syrup | | | \$20.00 | | | \$20.00 |
| 2 | | | | 2 | | | 2 |
| 3 | | | | \$20.00 | | | \$20.00 |
| 4 | Chef Anton's Cajun Seasoning | | | | | \$44.00 | \$44.00 |
| 5 | | | | | | 2 | 2 |
| 6 | | | | | | \$44.00 | \$44.00 |
| 7 | Chef Anton's Gumbo Mix | | | | | | \$21.35 |
| 8 | | | | | | | 1 |
| 9 | | | | | | | \$21.35 |
| 10 | Gustafs Knäckebröd | \$105.00 | | | | | \$105.00 |
| 11 | | 5 | | | | | 5 |
| 12 | | \$105.00 | | | | | \$105.00 |
| 13 | Queso Cabrales | | | | \$63.00 | | \$63.00 |

28 Selecting fields and records, scrolling, shortcut keys

To select a range of fields and/or records in the table view

Do one of the following:

- Use the mouse cursor.
- Press **Shift** and use any of the scrolling keys.
- Press **Ctrl+Space** to select the current field in all records
- Press **Shift+Space** to select the entire current record

To jump to the desirable record or to select a record range

Click the **Record** field on the bottom toolbar or press **Ctrl+G**, then enter the record number or a record range (as two numbers separated by ":") and press **Enter**.

You can also use menus attached to the "Previous"/"Next" buttons to scroll up or down by one record or to some specific record:

- Top / bottom
- Previous / next non-empty field
- Previous / next empty field
- Previous / next identical field value
- Previous / next different field value

Navigation / shortcuts keys

| | |
|---------------------------------------|---|
| Up, Down, Left, Right | scroll by one row/column |
| Ctrl+Up, Ctrl+Down | Go up or down to the next empty field |
| Alt+Ctrl+Up, Alt+Ctrl+Down | Go up or down to the next field with the same content |
| PgUp, PgDn, Alt+PgUp, Alt+PgDn | scroll by one page vertically or horizontally |
| End | last non-empty column |
| Home | first column |
| Ctrl+End | last non-empty row |
| Ctrl+Home | first row |
| | |
| Ctrl+PgUp | previous table (in the print-preview mode: previous page) |
| Ctrl+PgDn | next table (in the print-preview mode: next page) |

Other shortcuts

| | |
|-----------------|----------------|
| Ctrl+B | bold font |
| Ctrl+I | italic font |
| Ctrl+U | underline font |
| Ctrl+K | strikeout font |
| F6 | next pane |
| Shift+F6 | previous pane |

To use the 'auto-scroll' function

Click the worksheet window using the mouse wheel button and choose the desirable scrolling speed and direction by moving the mouse cursor around the clicked point.

29 Formatting

Fonts

The Font dialog box enables you to specify all basic font attributes including its name, size, weight, slant (the italic style), the **underline** and **strikeout** options and the font color.

To revert to the default settings, click the **Reset** button.

Style

You can use the following standard numeric styles and their options:

- Default
(No specific formatting. Numbers with more than 15 digits are displayed in the exponential-scientific format.)
- General
 - decimal places (0 to 14)
 - leading zeros (0 to 14)
 - thousand separator
 - displaying negative numbers in red or in parenthesis
- Currency
 - currency symbol position
 - currency symbol
 - displaying negative numbers in red or in parenthesis
- Accounting
 - currency symbol
- Date
 - date pattern
 - using fixed or system dependent year/month/day order
- Time
 - time/period pattern
- Date/Time
 - date pattern
 - time pattern
 - using fixed or system dependent year/month/day order
 - date/time order
- Percent
 - decimal places (0 to 14)
- Fraction
 - either a fixed denominator value (1 to 9,223,372,036,854,775,807) or a fixed number of denominator digits (1 to 19)
- Scientific
 - decimal places (0 to 14)
 - a fixed exponent value

Alternatively, you can use a **user-defined** numeric format. To do this, simply enter the desirable style pattern in the Style dialog box. If you need to re-use it, you can save it as a new user style with a new name. To learn more about style patterns, please see how the standard styles are defined. For example, the accounting format is defined as:

```
_(\$* #,##0.00_);"(\$"* #,##0.00\);_(\$* \-??_);_(@_)
```

and the fraction format with the fixed denominator (4) can be defined as:

```
?\ ?/\4
```

User-defined formats allow you to specify font colors for numbers that meet some conditions. For example:

```
[green]#.#;[red]\-#.#;[blue]#.#;[gray]@
```

displays positive numbers in a green font, negative numbers in red, blue zeroes and gray text labels,

```
[red][<10]#0.00;[yellow][<=50]#0.0;[green][<400]##0;[magenta][>=400]#00
```

displays numbers less than 10 in a red font, numbers less or equal to 50 in a yellow font, numbers less than 400 in a green font and numbers greater or equal to 400 using a magenta font.

Color values can be expressed as RGB values (for example, [red][<10]#0.00 is the same as [#FF0000][<10]#0.00) or by one of the following color names:

| | | | |
|---------------|------------------|----------------|------------------|
| black | [#000000] | maroon | [#800000] |
| green | [#008000] | olive | [#808000] |
| navy | [#000080] | purple | [#800080] |
| teal | [#008080] | gray | [#808080] |
| silver | [#C0C0C0] | red | [#FF0000] |
| lime | [#00FF00] | yellow; | [#FFFF00] |
| blue | [#0000FF] | fuchsia | [#FF00FF] |
| aqua | [#00FFFF] | | |

Alignment

The Alignment dialog box enables you to specify the following options:

- how to display text strings that exceed the cell width,

- horizontal and vertical indents (the distance measured from the cell contents to the cell boundaries),
- text rotation,
- horizontal (left, center, right) and vertical (top, center, bottom) alignment; unformatted cells display vertically centered, right-aligned numbers and vertically centered, left-aligned text strings.

To revert to the default settings, click the **Reset** button.

Drop-down lists

You can use the Drop-down list dialog box to set the drop-down list format and/or to manage (create, edit, delete) lists in the current workbook.

When editing a field with some drop-down list attached to it, you can quickly choose some predefined cell value instead of re-typing it.

To set that format, simply choose an existing list or click the **New List** button.

To add list items, enter them in the **List Items** edit field: one item in one line.

If you choose the **Append new items** option, each unique value entered in the corresponding field(s) will be added to the list automatically.

If the **Sort items** option is not selected, the list will display all its elements in the same order as they were added/entered.

If the **Multiple selection** option is selected, you can select more than one list item. Use the Ctrl and Shift keys (along with clicking) to make such selections. Multiple item are copied to the corresponding cells as comma-separated lists. Multiple selections are not treated as new items and are not added to the list.

One list can be used by any number of fields in any table. You may create up to 100 lists in one database file. To move lists between files, you must copy/paste the data from the **List Items** edit field.

30 Formatting - Hyperlinks

Use the **Format > Hyperlinks** command to set the hyperlink style for a given field. Clicking such a field or pressing a space will open that link. Hyperlinks can be links to another record/table, www addresses, e-mail addresses, regular files/folders on your local disk etc. You can explicitly specify the link type/protocol adding one of the following prefixes:

- table://
- http://
- https://
- file://
- mailto:

- ftp://
- gopher://
- nntp://
- news://
- wais://
- telnet://
- prospero://
- res://

Links between individual records can be easily created by the **Edit > Copy As Hyperlink** command. After it's copied, you can paste it in any other text field in any record/table. After the first use of that copy command variant, you can use the Ctrl+C shortcut to repeat it.

Links between records can be also entered manually as text. The format is:

some-table-name, record-number

where the record-number is the physical record number in a given table.

For example:

table://orders, 4555 (a link to the 4555th record in the "orders" table)

table://folder1\receipts, 2227 (a link to the 2227th record in the "receipts" table in the folder1 subfolder)

table://11455 (a link to the 11455th record in the same table)

31 Formatting – Images

The **Format > Images** style enables you to enter in a text field any sequence of image names and then display the result in one line in a cell.

The predefined image names (available in every database) include the following buttons, flags and symbols:

- tick
- cross
- flag-green
- flag-red
- flag-white
- star

33 Searching / Filtering Records

To set/modify a field filter

Scroll to a given field and press **Ctrl+F** or **Ctrl+Enter** or click the **Search** button or use the **Tools > Set Search Filter** command.

Filters specified for more than one field at the same are assumed to be merged by the AND logical operator and narrow down the search results.

- The displayed Search dialog box enables you to specify the filter and its type.
 - Regular expressions
 - "Starts with", text patterns with the * and ? masks
 - Equal, Not Equal, Greater than, Smaller than
 - Between
 - AND, OR
 - IsEmpty (also after removing whitespaces)
 - Similar ("fuzzy" searches)
 - Flagged records (with user-defined flags)
 - Formula (including over 300 functions and references to fields in one record)

To search for the currently selected field value, click the "=" button or press **Ctrl + =**.

- The default filter type for text fields is **Regular Expression** and for numeric fields: **Equal**.
- After a given filter type is used for some field, it becomes its default type.
- The default initial filter type can be changed in the **Settings > Options** dialog box.
- **Allow empty matches** and **Match case** are two options used by the **Regular Expression** filter type only.
Empty matches occur for **Regular Expression** like \A\z or a? .

There are three filter types available for numeric and text fields:

1. **Regular Expressions (RegEx)** - the filter string is a data pattern based on standard regular expression syntax. You can click the "?" button to display a list of commonly used expressions. Some examples of regular expressions:

abc - matches substrings "abc",

.bc - matches three character substrings consisting of any character followed by "bc",

\Aabc - matches field contents starting with "abc",

abc\z - matches field contents ending with "abc",

`\Aabc.*123\z` - matches field contents starting with "abc" and ending with "123" with any number of other characters in between,
`abc\d\z` - matches substrings ending with "abc" and one digit,
`^a\d+` - matches field contents containing a line starting with "a" and at least one digit,
`^a\d*` - matches field contents containing a line starting with "a" and zero or more digits,
`[ab]+c` - matches substrings "abc", "aabc", "abbabc" etc. and not "c",
`[ab]*c` - matches substrings "abc", "aabc", "abbabc" etc. and "c",
`[^ab]+c` - matches substrings ending with "c" and not containing "a" or "b",
`\w\d{2,3}` - matches substrings consisting of one letter followed by 2 or 3 digits,
`\Aab\d{2,3}c` - matches field contents beginning with "ab", two or three digits and "c",
`abc|xyz` - matches substrings "abc" or "xyz",
`\A\z` - matches empty fields,

For more information, see PCRE regular expression syntax summary

2. **Plain text (Starts with, Equal, Not equal, ...)** - the filter string represents a plain text string that is compared against the whole field contents. This filter category includes a few variants: "Pattern", "Equal", "Not equal", "Greater than", "Less than", "Between", "And" and "Or".

For the "Starts with", "And" and "Or" variants the filter string can be a prefix of the compared strings and can contain wildcard "?" and "*" characters. Any non-wildcard "?" and "*" characters must be prefixed with a tilde (~).

The remaining variants perform exact text string comparisons. For "Between", "And" and "Or" the entered filter must be a list of respectively two or more elements separated by spaces.

Examples:

`abc` - matches field contents beginning with "abc" ("Starts with"),
`?bc` - matches field contents beginning with any character followed by "bc" ("Starts with"),
`*abc` - matches field contents ending with any character followed by "abc" ("Starts with"),
`*` - matches all non-empty field contents ("Starts with"),
`50.1 100.2` - matches values equal to or between 50.1 and 100.2 ("Between", US regional numeric settings),
`12/1/2012 12/31/2012` - matches dates equal to or between the specified ones ("Between", US regional date settings),
`ab cd *ef` - matches field contents equal to "ab", "cd" or ending with "ef" ("Or")

3. **Formulas** - the filter string represents a formula expression that can make use of all the built-in functions, operators and references to other record fields. The formulas are similar to those used in spreadsheets with a few exceptions:

cell references are replaced by field names,

the range ":" operator is not available,
direct references to other tables via the "!" and "_" operators are not available.

When searching, such a formula is evaluated for each record and a given record is considered to meet the searching criteria if each the formula returns a non-zero value.

To learn more about building formulas, see: [Entering formulas](#)

4. **Flagged records** - searching for records flagged/marked by a given flag, which is a unique combination of field font and background colors.

Long Text, Images/Files and **Code** fields always use **Regular Expressions** to filter records. When filtering **Images/Files** fields, the file names are searched.

Clicking the "Paste" displays a list of recently entered search strings.

To remove all specified search filters, click the **(None)** search key on the toolbar or use the **Tools > Reset Searching** command.

To filter records and search for duplicated field values

Scroll to a field or select a range of fields which should be searched and use the **Tools > Find Duplicates** command.

To find duplicated (whole) records, select an entire row clicking the row heading or pressing **Shift+Space**.

To filter records and display the complement of the current record set

To find the complement of the current record set (that is, all the records that are not included in the current record set), use the **Tools > Find Complement** command.

To perform full-text searches and search for patterns occurring in any text and numeric fields

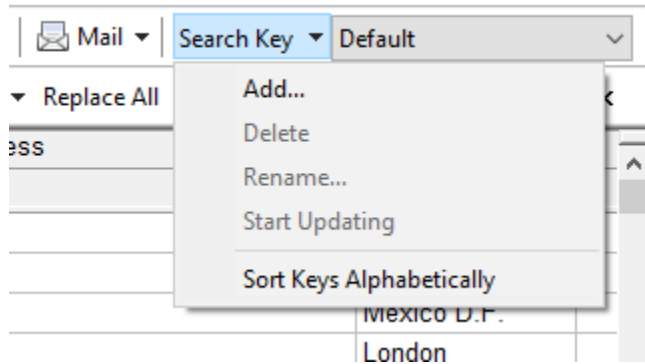
Use the **Edit > Find (F3)** command and in the displayed **Search Toolbar** enter the desirable substring. The **Find Previous/Next** commands simply scroll to the subsequent found table or form cell values. The **Find All** command performs full-text filtering of **Text** and **Numeric** fields (**Long Text** fields are not included).

The **search toolbar** also enables you to replace found substrings. If you want to perform such a find-and-replace action for one field only, select the column clicking its header.

[34 Searching / Saving Current Filters As Named Search Keys](#)

To save the current field filters as a named search key

Click the **Search Keys** toolbar button and use the **Add...** command from the displayed menu.



To apply filters saved as a named search key

Choose the key name from the **Search Keys** combo box on the main toolbar.

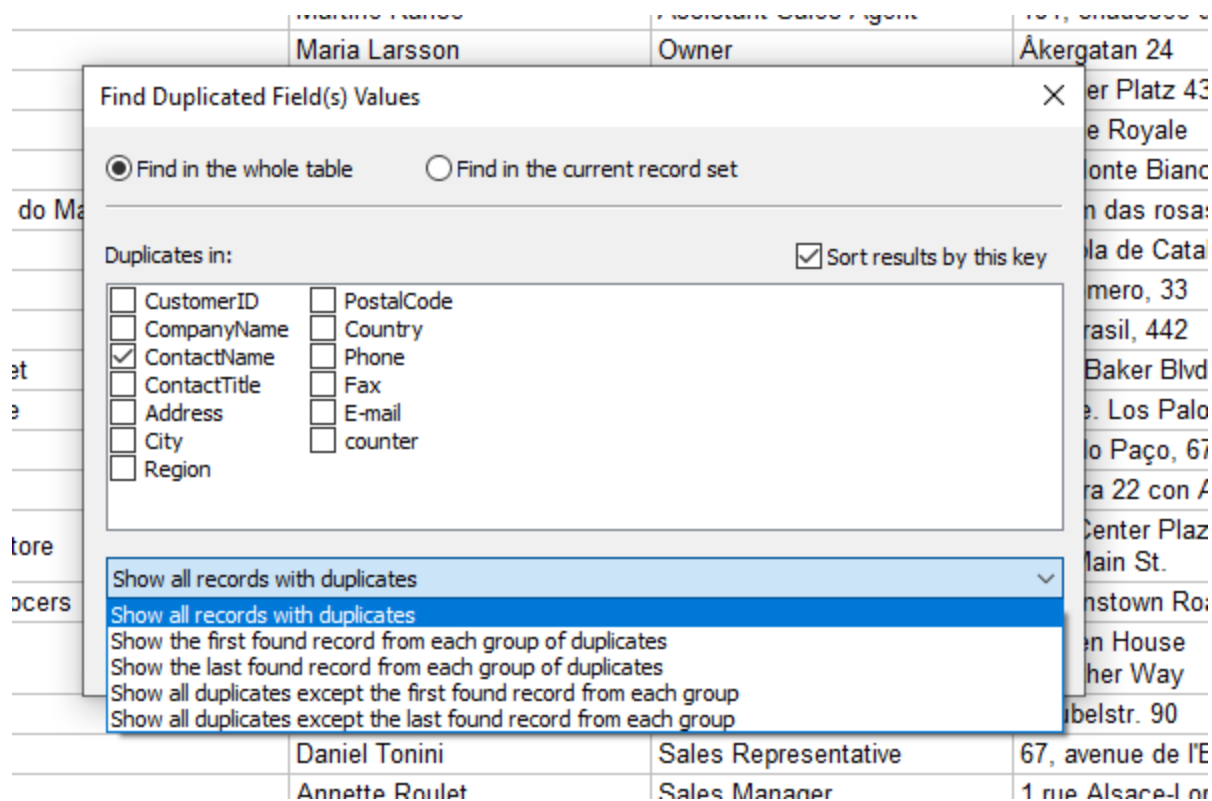
To modify filters saved as a named search key

Select that key then click the **Search Keys** toolbar button and choose the **Start Updating** command. After adding/deleting/modifying desirable filters end the updating mode.

35 Searching - Duplicates

To find duplicated values in one or more column, use the **Tools > Find Duplicates** command. In the displayed dialog box you can specify:

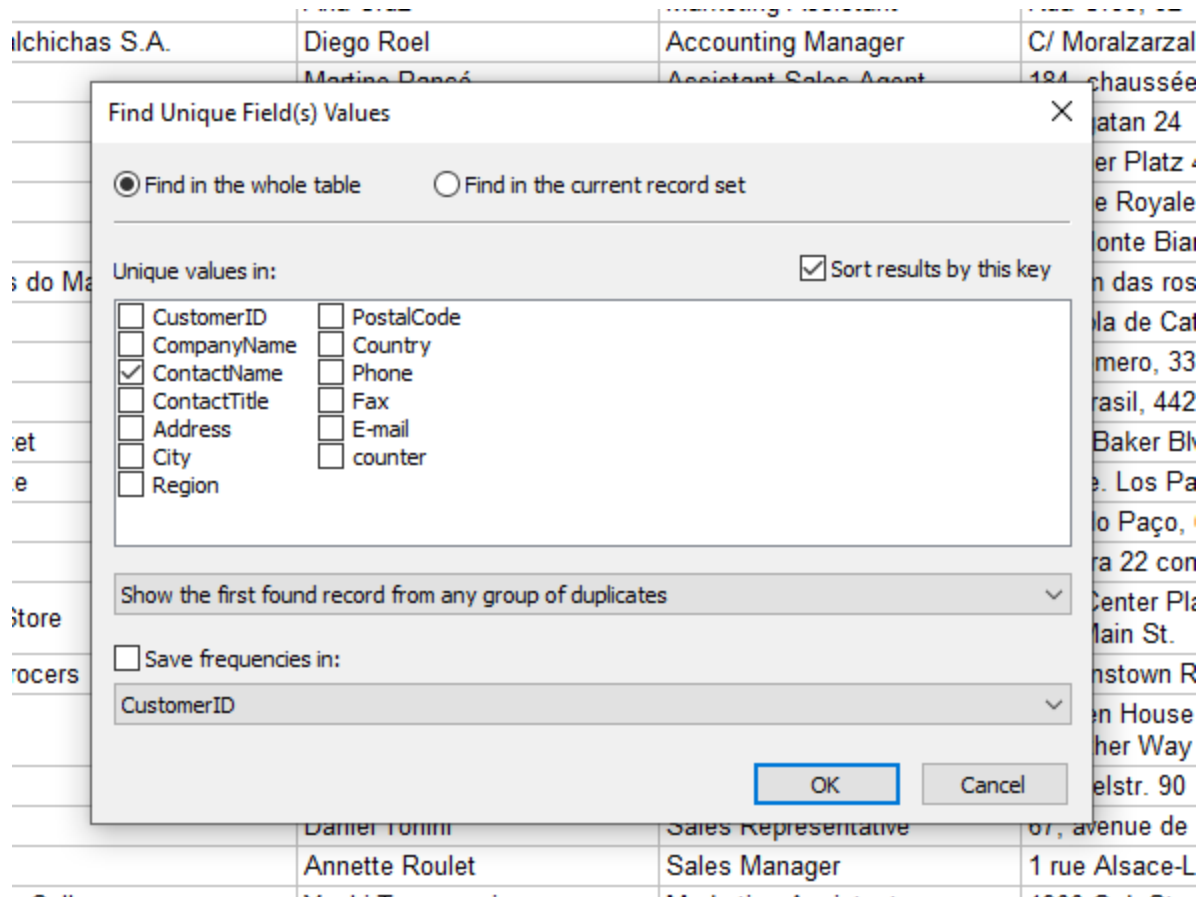
- whether to search for duplicates within the entire table or the current record set
- which record order is to be used for displaying: the physical table order or any sort order
- up to 100 fields to be used as a duplicate key
- how to display the results:
 - the first (entered in the table) records from each group of duplicates,
 - the last (entered in the table) records from each group of duplicates,
 - all records from each group of duplicates except the first one,
 - all records from each group of duplicates except the last one.



36 Searching - Unique field values

To find/list all unique values in one or more column, use the **Tools > Find Unique Values** command. In the displayed dialog box you can specify:

- whether to search for unique values within the entire table or the current record set,
- which record order is to be used for displaying: the physical table order or any sort order,
- up to 100 fields to be used as a duplicate/unique key,
- how to display the results:
 - the first (entered in the table) records from each group of duplicates,
 - the last (entered in the table) records from each group of duplicates,
- a numeric field where the frequencies (number of occurrences of found values) will be saved.



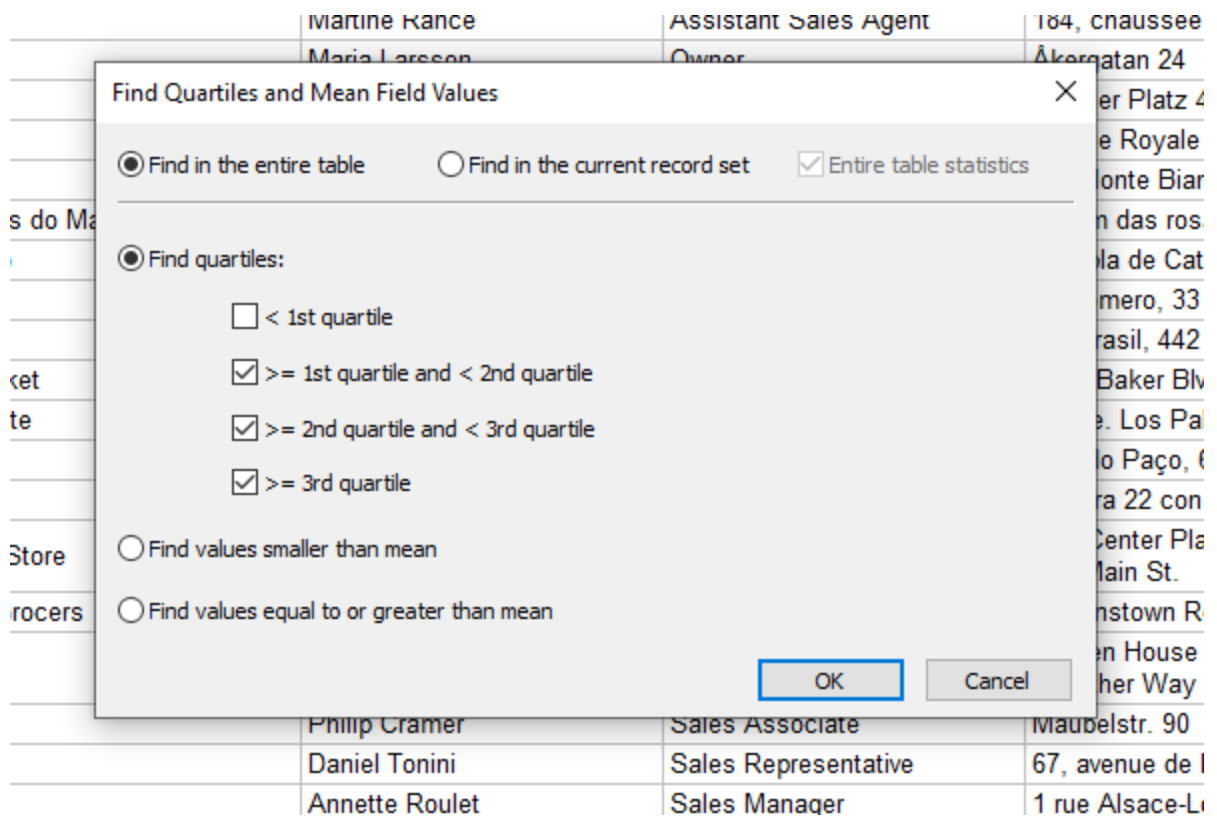
37 Searching - Quartiles, mean values

To find/list field values from a given quartile or below/above the mean value, use the **Tools > Find Quartiles / Mean** command. In the displayed dialog box you can specify:

- whether to search the entire table or the current record set,
- which statistical parameters should be used: for the entire table or the current record set.

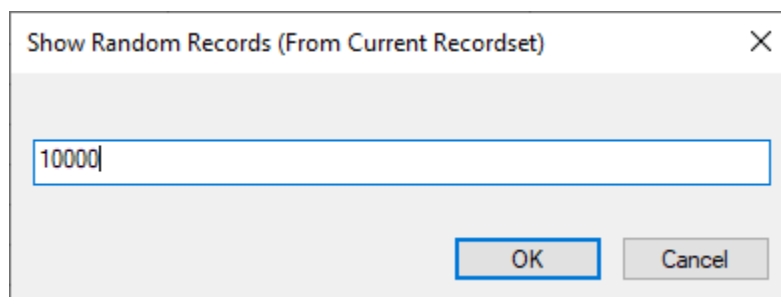
Note: for non-numeric fields the statistics is obtained as follows:

- string lengths for not formatted Text fields,
- numeric date/time values for Text fields formatted as Date/Time,
- the total size in bytes for Long Text and Images/Files fields.



38 Searching - Random records

Use the **Tools > Show Random Records** command to display a given number of randomly selected records from the current recordset.



39 Searching - Selected records

Use the **Tools > Show Selected Records** command to display the selected range of records as the current recordset.

40 Full Text Searches and Replacing

To perform full-text searching use the **Edit > Find (F3)** command and in the **Find** field of the displayed Find & Replace toolbar enter the data pattern or plain substring to look for.

The **Find Previous/Next** commands simply scroll to the subsequent found table or form cell values. The **Find All** command performs full-text filtering of **Text** and **Numeric** fields (**Long Text** fields are not included).

Depending on which view is currently active (a table, form or a memo/object field), the searching procedure can concern:

- all numeric and text fields of all records contained by the (optionally filtered) table
- all numeric and text fields of the record displayed on the form
- the current memo field

There are three search modes that can be specified using the **Options** button menu:

- **Regular Expressions** - the **Find** string is a data pattern based on standard regular expression syntax. Some examples of regular expressions:

abc - matches substrings "abc",

.bc - matches three character substrings consisting of any character followed by "bc",

\Aabc - matches field contents starting with "abc",

abc\z - matches field contents ending with "abc",

\Aabc.*123\z - matches field contents starting with "abc" and ending with "123" with any number of other characters in between,

abc\d\z - matches substrings ending with "abc" and one digit,

^a\d+ - matches field contents containing a line starting with "a" and at least one digit,

^a\d* - matches field contents containing a line starting with "a" and zero or more digits,

[ab]+c - matches substrings "abc", "aabc", "abbabc" etc. and not "c",

[ab]*c - matches substrings "abc", "aabc", "abbabc" etc. and "c",

[^ab]+c - matches substrings ending with "c" and not containing "a" or "b",

\w\d{2,3} - matches substrings consisting of one letter followed by 2 or 3 digits,

\Aab\d{2,3}c - matches field contents beginning with "ab", two or three digits and "c",

abc|xyz - matches field contents containing "abc" or "xyz" - logical OR,

(?=.*abc)(?=.*xyz) - matches field contents containing "abc" and "xyz" - logical AND,

`\A\z` - matches empty fields,

For more information, see PCRE regular expression syntax summary

The **Replace** string can contain:

- (1) absolute references (by number) to capturing subpatterns, eg. `\1`, `\2...`
a capturing subpattern (or a "group") is a part of the pattern enclosed in `()` parenthesis.
- (2) `\l`, `\L`, `\u`, `\U` literals to (binary) switch upper- and lower-case conversion
- (3) `\r`, `\n` - 'line feed' and 'new line' literals (by default, Alt+Enter & Ctrl+Enter insert the `\n` character into the text field).
- (4) `\s` - a single space character.

**Note: to find and replace Unicode/non-ascii characters, use Unicode reg. expressions,
e.g. `\p{L}` instead of `\w`, `\P{L}` instead of `\W` etc.**

Examples:

Find pattern: `cat`
Replace string: `dog`
Result: replaces "cat" with "dog"

Find pattern: `\\`
Replace string: `/`
Result: replaces the "\" characters with "/"

Find pattern: `\A\z`
Replace string: `NULL`
Result: fills empty fields with sthe "NULL" strings

Find pattern: `\ANULL\z`
Replace string:
Result: if a field contains only the "NULL" string, deletes its contents

Find pattern: `\b(\w+)(?:\W+\1\b)+`
Replace string: `\1`
Result: removes duplicate words in fields

Find pattern: `(\b\w)(\w*(\W+|\z))`
Replace string: `\u\1\2`
Result: converts first letters in words to uppercase, other to lowercase

Find pattern: `(\b\w)(\w*(\W+|\z))`
Replace string: `\1`
Result: creates abbreviations consisting of first letters of words

Find pattern: `(\b\w(\d*))(\w*(\W+|\z))`
Replace string: `\1`

Result: creates abbreviations consisting of first letters of words, leaves full numbers

Find pattern: `\b((\d{1,3}\.){3,3})\d{1,3}\b`

Replace string: `\1*`

Result: mask IP address (e.g. 11.12.13.114 to 11.12.13.*)

Find pattern: `(\S*)(\s*)(\S*)(\s*)(\S*)`

Replace string: `\g5\g4\g3\g2\g1`

Result: reverses the order of up to first 3 words in fields

Find pattern: `\R`

Replace string: `\s`

Result: replaces line-breaks with spaces

Find pattern: `ab+`

Database field content: abcdefaabb

Replace string: `x`

Replacement result: xcdefax

Find pattern: `(\.)a+\d{1,3}`

Database field content: abc aa0102

Replace string: `\1`

Replacement result: abc 2

Find pattern: `(ab)`

Database field content: abcdef ghijk abb123

Replace string: `\u\1\l\1xyz\s`

Replacement result: ABabxyz cdef ghijk ABabxyz b123

Find pattern: `\R`

Database field content: abc

def

ghi

Replace string: `\s`

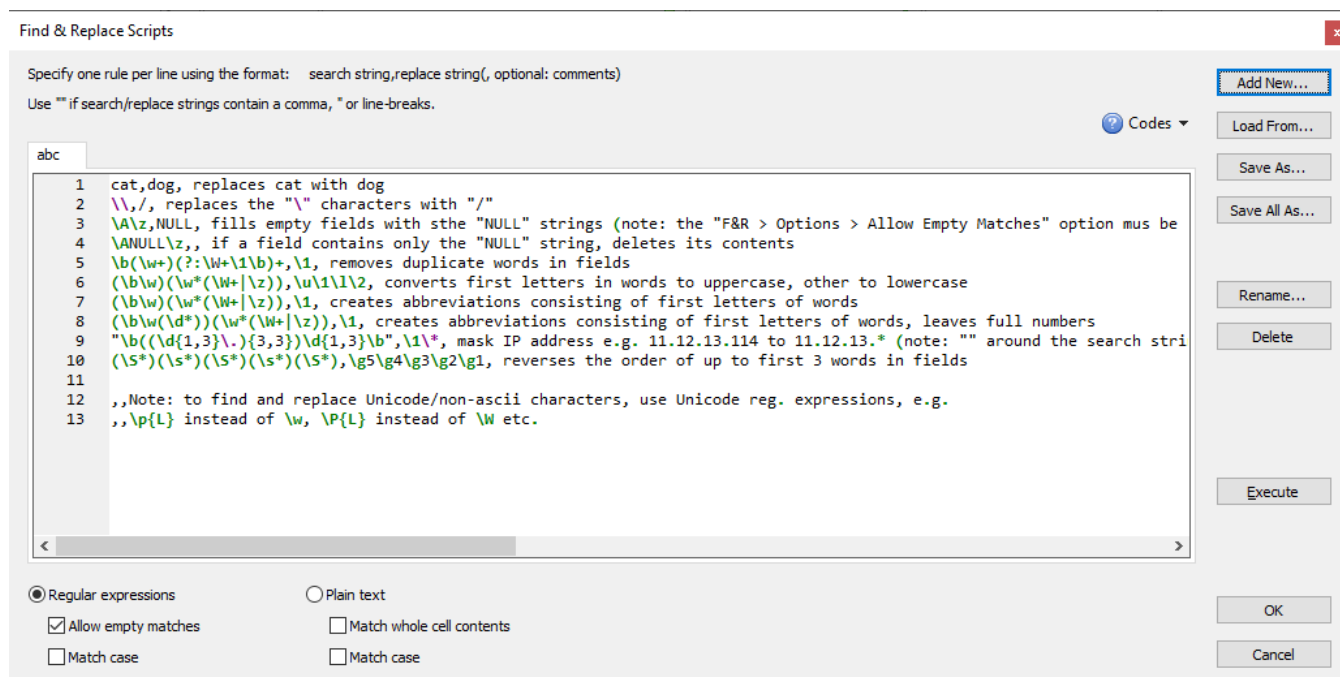
Replacement result: abc def ghi

- **Plain Text - Partial Matching** - the **Find** string represents a plain text string that is compared against the beginning of the field contents. The string can contain wildcard "?" and "*" characters. Any non-wildcard "?" and "*" characters must be prefixed with a tilde (~).
- **Plain Text - Full Content Matching** - the **Find** string represents a plain text string that is compared against the whole field content. The string can contain wildcard "?" and "*" characters. Any non-wildcard "?" and "*" characters must be prefixed with a tilde (~).

If the full text searching is performed for a single memo field data, the search mode automatically defaults to the **Plain Text - Partial Matching** search mode.

The "Scripts" button on the "Find & Replace" toolbar can be used to performed quick mass text replacing in a given table.

Note: the "search" and "replace" strings entered in the window below must be entered like in a csv text file, so text containing commas, line breaks or quoting symbols must be in "" and inner quoting symbols must be doubled.



41 Find-As-You-Type and Find-Similar searching options

If the "Find As You Type" option is selected, entering/deleting characters in the "Find" edit field results in automatic full-text searching for the whole table.

If the "Find In Current Results" or "Find Similar In Current Results" options are selected, searching is performed withing the current record set only and using the subsequent "Find All" commands in these modes means the "AND" searches and causes subsequent narrowing the search results.

The "Find Similar" option matches strings same as spell checking does it to find suggestions for misspelled words.

By default the "Find As You Type" mode and the "Find All" command in the other modes enable searching all fields. Making any selection that spans one or more columns/fields will limit searching to those columns/fields, for example selecting a range 1 row x 3 columns will result in searching within those 3 columns/fields only.

42 Sorting

To sort records

To specify more complex sort keys and to change the default comparison functions, use the **Tools > Sort** command.

When sorting **Long Text**, **Images/Files** and **Code** fields, GS-Base compares the total size of the field contents.

Note: Objects inserted as links (shortcuts) to external files (with the **Insert Shortcut to...** command) will be sorted by size if they were inserted in GS-Base ver. 12.1.3 or later. Unless re-inserted, earlier links will be treated as 1-byte objects.

You can use the **Sort > Use the current sort order as the default order** command to make the current sort order the permanent default table order.

43 Sending e-mail messages

To send messages to addresses entered in a given database field, use the "Tools > Send Mail" command or click the corresponding toolbar button. The messages can be sent either to addresses from all currently filtered records or to addresses included in the currently selected range of records/fields.

The "Send E-mail Messages" dialog box enables you to specify all the usual mailing parameters:

1.

Send E-mail Messages

From: Citadel5 E-mail address: sales@citadel5.com

To: email Bcc addresses:

Subject: GS-Base 17.1 has been released

Files: E:\gsb17.1_scr1.png (147.08 KB)

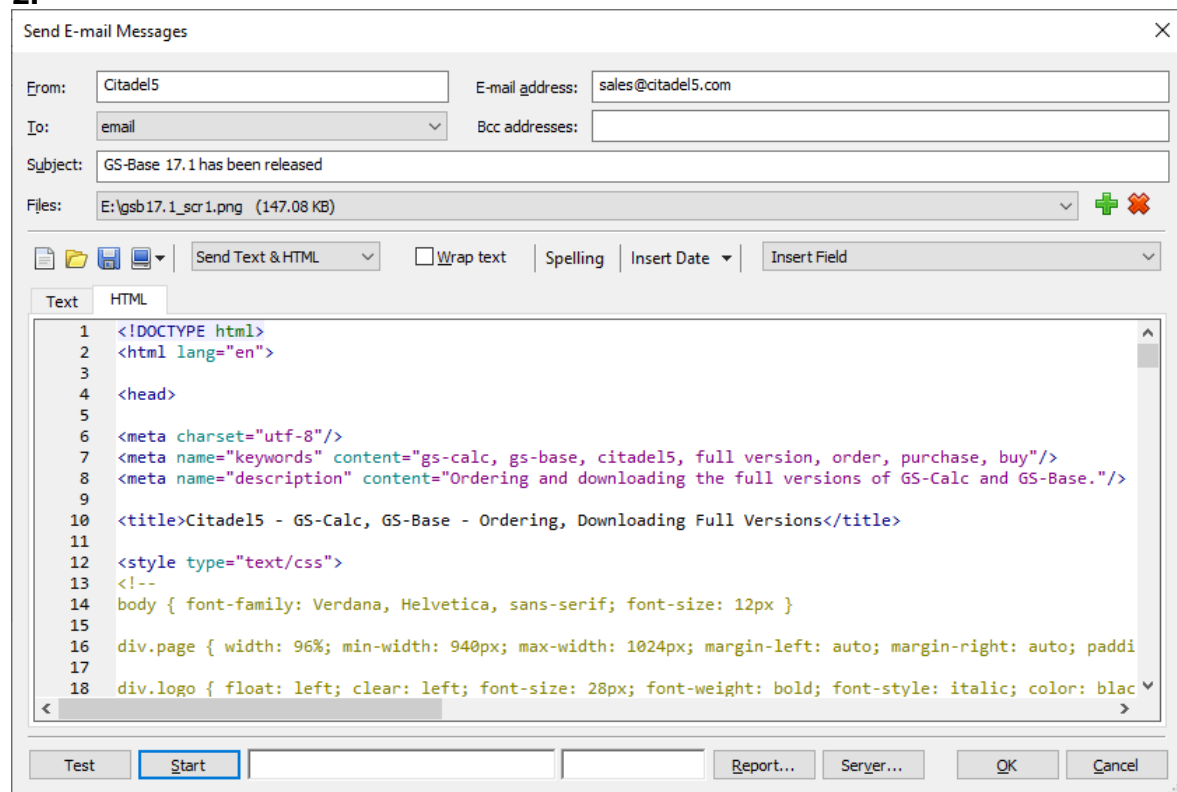
Send Text ☐ Wrap text Spelling Insert Date Insert Field

Text HTML

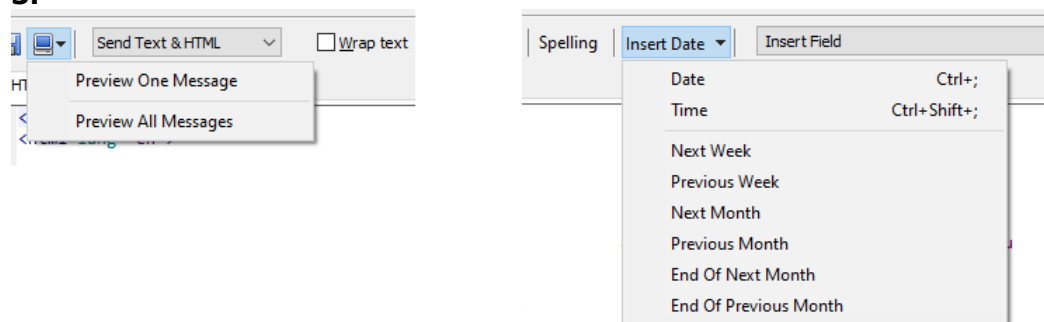
1 Hi {email},
2 |
3 I thought you might want to know that new versions of GS-Calc
4 GS-Base are available for downloading.
5
6 For complete lists of changes in recent versions, please see
7 the forum announcements:
8 GS-Calc: <https://forum-eng.citadel5.com/viewforum.php?f=7>
9 GS-Base: <https://forum-eng.citadel5.com/viewforum.php?f=12>
10
11 To download the full versions, please use your login password
12 and the form at the top of the page [<https://citadel5.com>].
13
14 You're receiving this message because you're a registered user
15 of GS-Calc or/and GS-Base.
16
17 If you don't want to receive such update notifications, please reply

Test Start Report... Server... OK Cancel

2.



3.



Notes:

- The message body can contain embedded record fields. The corresponding field values from subsequent records are inserted in subsequent sent messages. The embedded fields can be references to fields of each type. For "Long Text" fields their unformatted contents will be inserted "in-place". For the "Images/Files" fields the objects they contain will be sent as attachments.
- If attachments are the same or mostly the same for each record the "Files" list can be used. Use the "+" and "x" buttons to add/remove such files.
- You can send messages as a plain text, html or mixed content. The html part must be composed by using the html tags directly and the html/scriting syntax coloring is provided to make it easier.

- The "Preview" commands shows the output as text in the system Notepad program.
- Clicking the "Test" button results in sending one sample message to the defined "Bcc" address.
- You can pause and continue the proces of sending messages at any time.

Setting up the SMTP server

GS-Base sends messages using a STMP server. All it's parameters can be defined in the "Server Information" dialog box:

The "Server Information" dialog box is shown with the following fields and options:

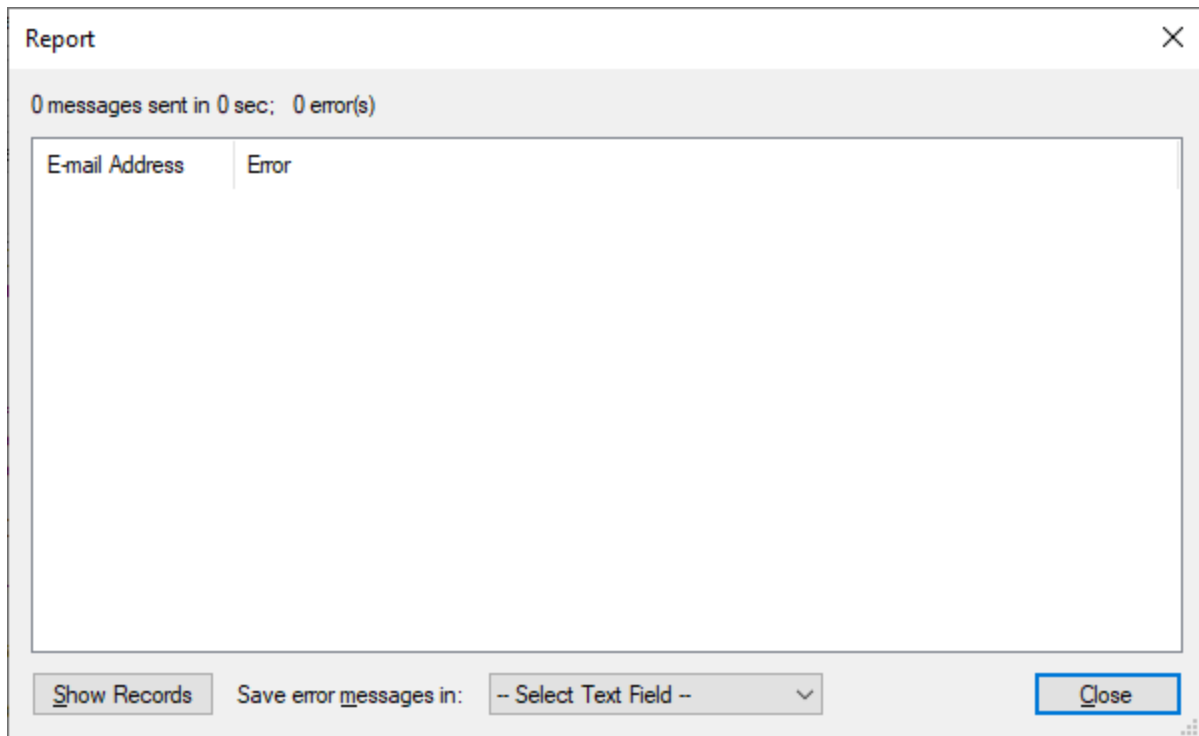
- SMTP server:** A text input field.
- Port:** A spin box set to 587, with a "Default: 587" label.
- Security:** A dropdown menu set to "STARTTLS".
- Server requires authentication:** A checked checkbox.
- User name:** A text input field.
- Password:** A text input field with masked characters (dots).
- Save password:** A checked checkbox.
- Pause:** A checkbox, currently unchecked. Next to it is a spin box set to 0, followed by "sec. every", another spin box set to 5, and "e-mails".
- Display connection security information:** An unchecked checkbox.

Buttons for "OK" and "Cancel" are located in the top right corner.

If the "Display connection security information" is on, after clicking the "Start" GS-Base will show these parameters first prior to sending anything and you choose to terminate the connection.

Checking sending errors

If sending a message for given record couldn't be completed by the STMP server, GS-Base saves the errors response codes and shows them after you click the "Report" button.



If the "Display connection security information" is on, after clicking the "Start" GS-Base will show these parameters first prior to sending anything and you can choose to terminate the connection.

44 Verifying URLs entered in database fields

You can check/verify URLs entered in database fields. GS-Base saves their current status (responses) and the last modification date (if it's available). You can filter the status codes yourself using field filters or you can let GS-Base verify the returned codes and automatically list all errors/broken URLs.

- You can pause and continue the verification process.
- These can be both http and https URLs, optionally with the ending ":number" port specification.
- Choosing cancel to close that dialog box causes retaining the previous results in the "status" and "modification date" fields.

Text, Images/Files or **Code** fields.

To specify which field should be printed and to turn on/off printing one-page forms, click the **Form** tab.

Letter:

GS-Base will print a user-defined *.rtf form containing any amount of text and graphics with embedded references to text and numeric record fields and with embedded formulas performing calculations. A field reference is specified by a field name in double-curly braces. A formula additionally must begin with "=".

One copy of such a document is printed for each record of the current record set.

Clicking the **Edit in External Application** command you can edit the form in the application associated with the *.rtf format. Changes are saved automatically after choosing the **Save** command in that application.

Labels:

You must define the required label format. To do this, click the **Labels** tab, then click the **New Labels** button and enter the name of your new label format.

Then specify the number of columns and rows of labels on one page, vertical and horizontal label spacing and insert data fields you want to print on each label.

Data fields can be simple database field names (references) or formulas (see: Entering formulas).

If the resulting text overflows the field rectangle:

- data entered as formulas is always wrapped and
- data entered as plain database field references is either wrapped or truncated, depending on the current format of that field.

If you want the formula data to span two or more lines, insert the char(13) or char(10) (new line/line feed) functions between field values and/or strings in that formula, for example: =Name & char(13) & City .

When editing labels you can move or resize the fields. To select a group of fields press **Shift** and click subsequent fields or press down the left mouse button and drag the mouse cursor.

To set the default height of the data field rectangle, right-click it.

To rename an existing label format, press **Shift** and click **New Labels** button.

Select the desirable **Page Range** option. You can print all found records, selected range of records or particular pages.

To print selected pages

Choose the **Pages** option in the **Print > General** dialog box and specify the list of pages to print.

The list can contain single pages or page ranges separated by commas. For example: 1,2,4-8,10

To print more than one page on a single paper sheet

Specify the number of printed pages in the **Pages per sheet** fields in the **Print > Layout** dialog box. For example, if you choose two columns and three rows, one sheet will contain (up to) six pages.

To include a table name, page number, date etc. in the printed pages headers and footers

In the **Print > Layout** dialog box, enter the header/footer text inserting the following special codes:

| | |
|----------------------|---|
| &p | Prints the current page number |
| &p+number | Prints the current page number plus 'number' |
| &p-number | Prints the current page number minus 'number' |
| &n | Prints the total number of pages |
| &m | Prints the current paper sheet number (different than &p when printing many pages on one sheet) |
| &f | Prints the full file path |
| &a | Prints the name of the printed table |
| &d | Prints the current date |
| &t | Prints the current time |
| &g | Prints the file modification date |
| &z | Prints the file modification time |
| && | Prints a single ampersand |

To format the header/footer text

In the **Print > Layout** dialog box, enter the header/footer text inserting the following special codes:

| | |
|---------------|---------------------------------|
| &l | Left-aligns the following text |
| &c | Centers the following text |
| &r | Right-aligns the following text |

To change the view scale in the print-preview window

Click the **Zoom In** button and pressing the right mouse button select a rectangle that should be zoom in.

46 Password protection and encryption

To password-protect and encrypt the data in the database zip file, use the **File > Protect** command and specify a password consisting of at least 6 characters.

To encrypt the data, GS-Base currently uses the Twofish CFB algorithm. All encryption parameters are saved in the **META-INF/manifest.txt** text stream in the database zip file.

If the database zip file is encrypted and you still want to manipulate some its data manually without GS-Base, using Windows Explorer, you must not alter the **META-INF/manifest.txt** file in any way or you'll risk loosing the data.

Files added by users manually to the zip that are not used/imported by GS-Base remain unencrypted and are moved to the "unused/" zip subfolder.

If a database is encrypted and protected, there is a green shield symbol displayed on the **status bar**.

Regardless of the above file-level protection you can still use individual field encryption in tables.

47 Saving files: PDF files

To save data in the PDF format:

- Use the **File > Save Table As PDF** and **File > Save All Tables As PDF** commands. The former saves the current table and the latter - all tables of the database. You can change the page layout and other content options using the **File > Page Settings** command.

Note: When saving a PDF file GS-Base maps all fonts used in a workbook to the standard set of 14 PostScript Type 1 fonts which are required to be present in every PDF viewing application and in every operating system.

To specify the advanced PDF save options, use the **File > Advanced PDF Save Options** command. GS-Base enables you to choose the language/code page that should be used when saving the current record set to a PDF file and to specify the custom character encoding list to correctly display all glyphs for that language.

If you don't choose any language before saving, GS-Base will use the default Windows system ANSI code page when converting the Unicode text to the ANSI text required by the PDF format.

For example, the encoding list for the **iso-8859-1** code page can be:

```
32 /space /exclam /quotedbl /numbersign /dollar /percent /ampersand /quotesingle
/parenleft /parenright /asterisk /plus /comma /hyphen /period /slash /zero /one
/two /three /four /five /six /seven /eight /nine /colon /semicolon /less /equal
/greater /question /at /A /B /C /D /E /F /G /H /I /J /K /L /M /N /O /P /Q /R /S
/T /U /V /W /X /Y /Z /bracketleft /backslash /bracketright /asciicircum /underscore
/grave /a /b /c /d /e /f /g /h /i /j /k /l /m /n /o /p /q /r /s /t /u /v /w /x /y /z
/braceleft /bar /braceright /asciitilde /bullet /Euro /bullet /quotesinglbase /florin
/quotedblbase /ellipsis /dagger /daggerdbl /circumflex /perthousand /Scaron
/guilsinglleft /OE /bullet /Zcaron /bullet /bullet /quoteleft /quoteright /quotedblleft
/quotedblright /bullet /endash /emdash /tilde /trademark /scaron /guilsinglright
/oe /bullet /zcaron /Ydieresis /space /exclamdown /cent /sterling /currency /yen
/brokenbar /section /dieresis /copyright /ordfeminine /guillemotleft /logicalnot
/hyphen /registered /uni203E /degree /plusminus /twosuperior /threesuperior
```

/acute /mu /paragraph /periodcentered /cedilla /onesuperior /ordmasculine
 /guillemotright /onequarter /onehalf /threequarters /questiondown /Agrave /Aacute
 /Acircumflex /Atilde /Adieresis /Aring /AE /Ccedilla /Egrave /Eacute /Ecircumflex
 /Edieresis /Igrave /Iacute /Icircumflex /Idieresis /Eth /Ntilde /Ograve /Oacute
 /Ocircumflex /Otilde /Odieresis /multiply /Oslash /Ugrave /Uacute /Ucircumflex
 /Udieresis /Yacute /Thorn /germandbls /agrave /aacute /acircumflex /atilde /adieresis
 /aring /ae /ccedilla /egrave /eacute /ecircumflex /edieresis /igrave /iacute
 /icircumflex /idieresis /eth /ntilde /ograde /oacute /ocircumflex /otilde /odieresis
 /divide /oslash /ugrave /uacute /ucircumflex /udieresis /yacute /thorn /ydieresis

The encoding list the **Windows-1250** code page could be:

32/space 40/parenleft/parenright 44/comma/hyphen/period/slash/zero/one/two/three/four
 /five/six/seven/eight/nine/colon 65/A/B/C/D/E/F/G/H/I/J/K/L/M/N/O/P 82/R/S/T/U
 87/W/X/Y/Z 97/a 99/c/d/e/f/g/h/i/j/k/l/m/n/o/p 114/r/s/t/u 119/w 121/y/z 140/Sacute
 143 /Zacute 156/sacute 159/zacute 163/Lslash 165/Aogonek 175/Zdotaccent 179/Islash
 185/aogonek 191/zdotaccent 202/Eogonek 198/Cacute 209/Nacute 211/Oacute
 230/cacute 234/eogonek 241/nacute 243/oacute

48 Opening and saving text files

To open a text file

Use the **File > Open** command and choose **Text** from the **File of type** list or drag and drop a given text file into the main GS-Base window.
 Next, specify additional text file options:

| | |
|---------------------------------------|---|
| Field separator | A single character used to separate fields in one line. |
| Fixed field widths | A list of fixed field width values. For example: 10,30,15,40 If not all widths are specified, the remaining fields are assumed to contain 1024 characters. |
| Quoting symbol | A symbol used to delimit fields containing separator characters. Fields containing quoting symbols, separators or new line characters must be delimited by quoting symbols. The inner quoting symbols must be doubled. If you need to open a text file that doesn't conform to this standard, consider specifying some unique quoting symbol that doesn't occur within that file. |
| Encoding | Specifies the text encoding for a given text file: UTF-8, UTF-16, ANSI 8-bit, ISO/OEM. The initial value is set based on the file initial bytes. |
| First row contains field names | Specifies whether text strings from the first line of the file should be treated as field names. If you clear this option when reading a text file, GS-Base will use simple "field_####" names. A field names must begin with a letter. |

Convert text to numbers

If you check this option, a column containing **only** text strings that can be converted to numbers (without formatting) will be treated as a numeric database field. Otherwise all such columns will be imported as text fields (and will be sorted and searched as plain text fields).

You can also change the field type after you import the file using the **Database > Field Properties** command.

Convert text to dates

If you check this option, text strings representing dates in one of the local system formats will be converted to generic date/time strings.

As selecting this option may significantly slow down opening a text file, it's recommended that you use the **Edit > Convert Field Data > Text To Standard Date String** command instead (for selected fields, after the file is opened).

To save a text file

Use the **File > Save Recordset As** command and choose **Text** from the **File of type** list. Next, specify the same additional text file options as above and the following ones:

Save 'Image/File' objects to a zip file Save 'Long Text' field contents a zip file

If you check this option(s), the contents of GS-Base binary **Image/File** and **Long Text** fields will be saved to an additional zip file [text_file_name].zip. The corresponding fields in the saved text table will contain file links referring to that zip structure. For example, if some record in the original GS-Base database has some "Memo" **Long Text** field and some long text in it, the saved text table some_text_file.txt instead of the "Memo" contents will contain the file path:

```
some_text_file/r[record_number]\f[field_number]    (like  
some_text_file/r00000002f00003\text.txt)
```

If you re-import the some_text_file.txt text table, unzip the some_text_file.zip file and use the **Format > Hyperlink** command to format that file link text field, clicking that link will open an application associated with the *.txt and showing that former GS-Base **Long Text** field contents. The same applies to images and other file types.

Note that depending on your database size the above may result in creating a zip file with a very large number streams/files and native Windows tools (e.g. the File Explorer) will require a lot of time to unzip it.

Leaving this option unchecked results in saving the text file only with labels describing the binary contents.

Note: When saving a text file, GS-Base 14.5.3 and earlier versions save only filtered records from the active table and GS-Base 14.7 and later versions saves the whole table using the default (unsorted) record order. To save only the current filtered and sorted recordset, you have to use the **Save Recordset As** command.

To automatically resize the column widths to fit the field contents in the opened file, select the **Options > General > Auto-fit column widths in imported files** option. Keep in

mind that for very large files this may slow down the process of importing. To optimize the resizing in the **Options > General** dialog box increase the declared number of processor cores that GS-Base can use.

49 Opening and saving MySQL *.sql files

To open a MySQL *.sql file

Use the **File > Open** command and choose **MySQL (*.sql)** from the **File of type** list or drag and drop a given *.sql file into the main GS-Base window.

Data from *.sql files can also be accessed via the **Add Table** and **Merge Records** commands.

The *.sql dump file must use the **INSERT INTO** command and must include the preceding table definitions.

The preferable method of saving data in binary **BLOB** type fields is hexadecimal (ascii text). If data are saved in **BLOB** fields directly as binary streams GS-Base decodes the following escape sequences:

\0, \n, \r, \, ', ", \z

To save a MySQL *.sql file

If the open database is already in the *.sql format, the **Save** command causes saving it back to the same file using the same, original table (field, keys) definitions. Characters sets are converted to **utf8mb4**.

If the current format is not MySQL *.sql, use the **File > Save Database Copy As** or **File > Save Recordset As** commands and choose **MySQL (*.sql)** from the **File of type** list. In this case the newly created MySQL table definitions will follow the rules specified in **Notes** below.

Notes:

- When editing existing MySQL *.sql files and saving them in this format GS-Base preserves MySQL table (field, keys) definitions except that used characters are converted to **utf8mb4**.
New fields added to such a file in GS-Base subject to the rules listed below.
- MySQL numeric fields become the **Number** fields in GS-Base.
The **Number** GS-Base fields are exported to new MySQL *.sql files as **DOUBLE** fields.
- MySQL **CHAR** and **VARCHAR** fields shorter than 8162 bytes become **Text** fields in GS-Base.
GS-Base **Text** fields are saved as MySQL **VARCHAR(255)** but no truncation is performed for data within the range 256-8162.
- MySQL **VARCHAR** longer than 8162 bytes and all MySQL **[...]TEXT** fields are used in GS-Base as **Long Text** fields.
GS-Base **Long Text** fields are saved as MySQL **LONGTEXT** fields.

- The binary MySQL **BLOB** field contents from *.sql files not created by GS-Base are represented in GS-Base (in **Images/Files** fields) by files with the "sql_blob.bin" name.
Data from GS-Base **Images/Files** fields containing just one such object will be saved back to the MySQL file as the same, original binary MySQL contents.
If a GS-Base **Images/Files** field contains more than one inserted object or the object name is not "sql_blob.bin", the resulting MySQL **BLOB** field will contain binary data representing the list of these objects in the following format:
(1) 36-character GS-Base GUID string: **EBDA2E02-5502-43C4-AF43-44EEF3375275**
(2) a repeating list of the elements:
 - 5 ascii characters representing the object/file name length,
 - the file name,
 - 12 ascii characters representing the object/file data length,
 - the file contents
- Files saved by GS-Base requires the MySQL system **max_allowed_packet** variable to be at least (approx.) 64MB. This is the default value for MySQL 8.0.
If some of your saved MySQL single BLOB/Text fields contains more data this variable must be increased accordingly.
- If a file in the MySQL *.sql format is opened, the Database Explorer tree also displays the SQL fields definitions along with the corresponding GS-Base field types.

To automatically resize the column widths to fit the field contents in the opened file, select the **Options > General > Auto-fit column widths in imported files** option. Keep in mind that for very large files this may slow down the process of importing. To optimize the resizing in the **Options > General** dialog box increase the declared number of processor cores that GS-Base can use.

50 Opening and saving html files

To open a html file

Use the **File > Open** command and choose **HTML** from the **File of type** list. Similarly to saving described below, GS-Base will load the first table that has the "id" attribute set to "gsbase", e.g.: <TABLE id="gsbase">

To save records to a html file

Use the **File > Save Recordset As** command and choose **HTML** from the **File of type** list. Next, specify additional html file options:

Template

Specifies an optional HTML file that should be used as a template. The file must contain a TABLE element with the ID attribute set to 'gsbase', e.g.

```
<TABLE id="gsbase"> ...
```

GS-Base will empty that table and fill it with the specified records. All attributes of the TR, TH and TD tags are removed; you should

specify column formatting using the COLGROUP and COL tags instead.

If you don't provide any template, GS-Base will create and save a basic/minimal HTML document containing a table filled with records.

Note that **Image/File** fields are not saved.

Saved fields

Specifies which fields should be saved.

Save duplicated field values as

Specifies a text string that should be used to replace the same values occurring in the same field and subsequent records.

Encoding

Specifies text encoding. Currently the only valid value is UTF-8.

First row contains field names

If you check this, the first row of the saved table will contain field names as TH elements.

51 Opening and saving dBase/FoxPro/Clipper files

To open an xBase file

Use the **File > Open** command and choose **dBaseIII, dBaseIV, FoxPro 2.x or Clipper** from the **File of type** list or drag and drop a given xBase file into the main GS-Base window.

Next, specify additional xBase file options:

Encoding Specifies the text encoding for a given xBase file: ANSI 8-bit or ISO/OEM.

dBase/FoxPro/Clipper records marked as "deleted" are displayed with the "Selected" flag (the default "0" flag for each GS-Base database).

To save an xBase file

Use the **File > Save Recordset As** command and choose **dBaseIII, dBaseIV, FoxPro 2.x or Clipper** from the **File of type** list.

Note: When saving a text file, GS-Base 14.5.3 and earlier versions save only filtered records from the active table and GS-Base 14.7 and later versions saves the whole table using the default (unsorted) record order. To save only the current filtered and sorted recordset, you have to use the **Save Recordset As** command.

Records marked with the the "Selected" flag (the default "0" flag for each GS-Base database) will be saved to dBase/FoxPro/Clipper databases as "deleted" records.

To automatically resize the column widths to fit the field contents in the opened file, select the **Options > General > Auto-fit column widths in imported files** option. Keep in mind that for very large files this may slow down the process of importing. To optimize the

resizing in the **Options > General** dialog box increase the declared number of processor cores that GS-Base can use.

52 Opening and saving Excel workbooks

To save a database (or just the current filtered table) as an Excel workbook

Use the **File > Save Database Copy As** or the **File > Save Record Set As** command and choose the **Excel *.xlsx** format or the **Excel 97-2003 *.xls** file format from the **File of type** list.

In the **Save Excel File** dialog box specify the following options:

Save Excel XLSX File - sample.xlsx

☒ Split 1M+ row tables into worksheets using name sequences: sheet, sheet(1),..., sheet(n)

☐ Split 1M+ row tables into files using name sequences: file, file(1).xlsx,..., file(n).xlsx

☐ Truncate 1M+ row tables

☒ User-defined worksheet row limit (1... 1048576) 1048576

☒ Field names in the first row in the first worksheet

☒ Field names in the first rows in all worksheets

☒ Restore GS-Base tree subfolders using this path separator: >

☒ Save Long Text/Images/Files/Code fields to the zip file: n/a

☐ Save Long Text/Images/Files/Code content type in cells

OK Cancel Help

- **Split 64K+ row tables using name sequences**

or

Split 1M+ row tables using name sequences

Tables with records exceeding the Excel row limit will be split into multiple Excel worksheets.

For example, if a database contains three tables "customers", "orders", "products"

and the "orders" table has more than 1M/64K records, then the saved Excel file will contain the following worksheets:

"customers", "orders", "orders(1)", "orders(2)",..., "products".

- **Split 1M+ row tables into files using name sequences** (for Excel *.xlsx files)

or

Split 64K+ row tables into files using name sequences (for Excel 97-2003 *.xls files)

If this option is selected and if the number of records in one or more of the saved tables exceeds the Excel row limit, GS-Base will split such tables and save multiple Excel workbooks:

file_name.xlsx (or file_name.xls)
file_name(1).xls (or file_name(1).xls)
file_name(2).xls (or file_name(2).xls)
...
file_name(n).xls (or file_name(n).xls)

If there are any previously saved files with larger indices (e.g. file_name(n+1).xls and on), they will be deleted.

If you open the file_name.xls file again in GS-Base, the remaining partial workbooks will be automatically (internally) loaded to form the original GS-Base tables.

- **Save field names in the first row in all first worksheet**

If this option and the 1st split option are selected, only the first worksheet from each split sequence will contain table field names in the first row.

If this option and the 2nd split option are selected, the first row in each worksheet in each split Excel workbook will contain the corresponding table field names.

- **Save field names in the first row in subsequent split worksheets**

In addition to the above, worksheets that were split within one workbook will also contain that top row with table field names.

- **Enable restoring the GS-Base folder structure [...]**

If this option is selected and if you created folders in the GS-Base database explorer pane for some tables, the Excel sheet names will be formed as whole such table tree paths which will enable GS-Base to re-created the original tree folder structure when you open the saved Excel *.xls workbook back in GS-Base.

As the "\" path separator can not be used in Excel names, it's replaced with ">".

- Binary field options:
 - **Save Long Text/Images/Files/Code fields to the zip file [...]**

The contents of the above fields will be saved to the

file_name.**gsb-bin.zip**

file. You need to keep the saved Excel *.xls workbook(s) and that file together (in the same folder). The corresponding sheet cells in the *.xls workbook will contain links to objects in that zip. To access them, unzip it to the

file_name.**gsb-bin**

folder. Clicking a sheet cell formatted as a hyperlink (in GS-Base: "Format > Hyperlink") will open the field contents (e.g. text, image or a folder with multiple images etc.). You can edit/update/delete them and distribute your file_name.xls workbook either with the unzipped or zipped folder:

file_name.**gsb-bin**

- **Save only Long Text/Images/Files/Code field content description in sheet cells**

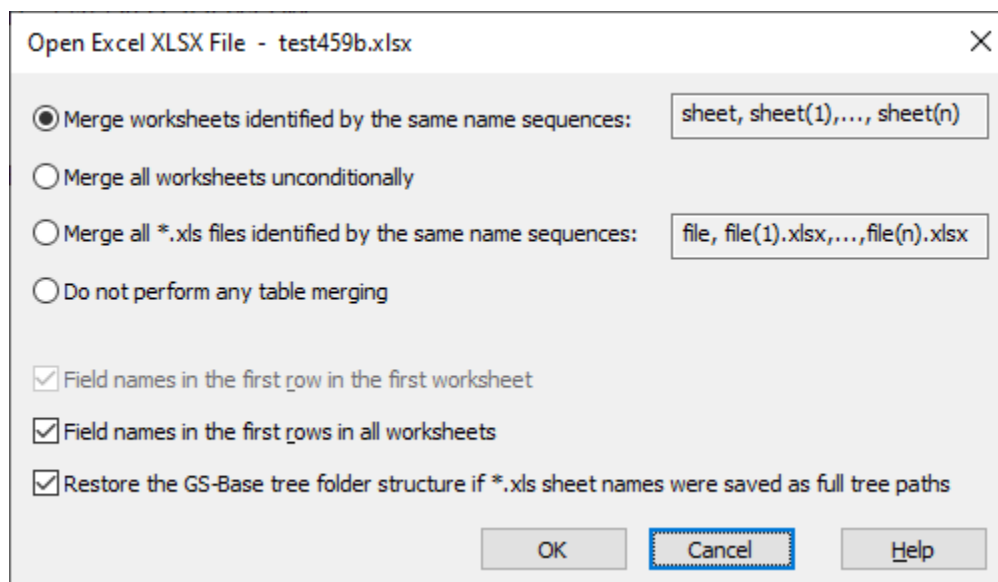
The description include the object names, number, total field size and recent modification date.

Note: After saving a document in a different format you may need to use the **File > Reload** command to refresh the view.

To open an Excel workbook

Use the **File > Open** command (or alternatively any of the external table merging commands) and choose the **Excel *.xlsx** format or the **Excel 97-2003 *.xls** format from the **File of type** list.

In the **Open Excel File** dialog box specify the following options:



- **Merge worksheets identified by name sequences**

It's the reverse of the 1st split option in the above "Save" dialog box.

- **Merge all worksheets worksheets unconditionally**

If this option is selected, all worksheets are assumed to be a sequence to be merged, regardless of their names.

- **Merge tables from previously split multiple Excel *.xls files [...]**

See the mirror option described above in the "Save" dialog box.

- **Field names in the first row in all first worksheet**

If this option is selected, the first row of the first worksheets from each merged sequence has to include the corresponding record field names.

- **Field names in the first row in subsequent split worksheets**

In addition to the above, all worksheets in each merged worksheet sequence also has to include that top row with field names.

- **Restore the GS-Base tree structure [...]**

See the mirror option described above in the "Save" dialog box.

Note: if subsequent worksheets in the same merged sequence have more columns, empty columns are added automatically to the preceding worksheets.

53 Settings

All GS-Base global settings are stored in the settings.xml files. Their location depends on how you install the application. If you choose the portable installation, the settings file will be saved directly to the installation folder. Otherwise they're store in the local application data folder (e.g. C:\Users\username\AppData\Local\GS-Base).

You can use the **Settings > Save - Load Profile** commands to save or load different settings profile files.

The following data can be specified in the **Settings > Options** dialog:

General Settings

Undo/Redo level

The number of actions that can be undone via the **Edit > Undo-Redo** commands.
A very high undo level causes greater memory usage if some very large copy/paste operations

are performed.

The drag-and-drop operation is treated as a two-step action so the minimum undo level is 2.

Valid range: <2, 100>

Default: 20

Number of CPUs

The number of processor cores that GS-Base is allowed to use when updating calculated fields (**Tools > Update All Calculated Fields**) and when filtering records (excluding full-text searches in memo fields).

Valid range: <1, 100>

Default: 2

Maximum number of records

Specifies the maximum numbers of records a single table can contain. For each running instance GS-Base initially allocates

"maximum numbers of records" * 4 bytes.

If your databases don't contain large numbers of records, you can choose some small limit to save memory.

This setting doesn't affect the database file format. After increasing the limit, any existing database can be expanded further up to the new limit. Available values:

- 64 K (65,536)
- 1 M (1,048,576)
- 12 K (12,582,912 **default**)
- 32 M (33,554,432)
- 64 M (67,108,864)
- 128 M (134,217,728)
- 256 M (268,435,456)

If a given database/files contains more records than the current GS-Base limit allows, a respective warning message is displayed.

Default text field filter

Specifies the default filter type for text fields. This filter type will be initially set in the Search dialog box.

Default: Regular Expression

Default pivot function

Specifies a function that will be used by each new or existing pivot table that doesn't have any data field defined.

The default **count** function results in counting all subsequent values of the specified row field(s).

Default: Count

AutoSave

Specifies the period of (user) inactivity after which GS-Base will automatically save the open database (if it's modified).

Default: Never

AutoClose

Specifies the period of (user) inactivity after which GS-Base will automatically close the open database. Modified databases are saved before closing.

Default: Never

After pressing Enter

Specifies whether (and how) to scroll the current table cell after you finish editing and press **Enter**.

Default: No action

Default file extension

Specifies the default file extension (either *.zip or *.gsb). GS-Base uses that extension when creating and saving new databases/files. Existing files are handled the same way, regardless of their extensions and the changes of that parameter.

Choosing the ***.gsb** extension enables you to register it in the Windows registry and open databases double-clicking them. The only disadvantage is that you can't view the zip contents using File Manager until you rename it to a *.zip file.

You can register the *.gsb file type in Windows and associate it with GS-Base using the **Settings > Register GS-Base *.gsb File Type** command. The corresponding **Settings > Unregister...** command removes all registry entries created during the registration.

Note: To use those commands you must start GS-Base as an administrator (e.g. right-click the GS-Base desktop shortcut and choose the "Run As Administrator" command).

Choosing the ***.zip** extension enables you to use File Manager to view/edit/browse text tables contained in a given database. In general, it might be helpful only if you're planning to perform some additional text (pre-)processing of these tables with external software or if you're planning to drag-and-drop text files/tables e.g. to quickly create some initial collections of tables manually, without GS-Base.

The disadvantage is that if a database contains a large number of inserted objects/images/memos, Windows will considerably slow down when indexing such zip files. Also, you can't open *.zip files in GS-Base double-clicking such files.

Default: *.zip

Start folder

Specifies the folder displayed in the **Open File** dialog box when you open it for the first time (after launching GS-Base).

Default: Empty

Relative shortcut path

If you insert some link *.lnk files into **Images/Files** fields and later change the location of the target objects that the links are connected to, you can "overwrite" such broken links globally specifying that path.

Default: Empty

No-deflate file types

Specifies the list of file types (stored in **Images/Files** fields) that will be saved in the database zip file without compression. The list should contain file types that contain already compressed data. This can improve loading/saving times without worsening the overall compression factor. Enter the extensions in the ***.ext1;*.ext2;...** form.
Default: *.png;*.jpg;*.gif;*.pdf;*.zip;*.ods

Show lists of files attached to databases

If you manually drag-and-drop any files to a given database zip file, GS-Base will detect such files and show the list of them.

If the option is off, GS-Base will silently load and save back such unused files.

Default: On

Show progress when opening/saving files

If this option is checked, GS-Base will be displaying a progress dialog box when opening or saving all files. The progress dialog box contains information about the opening/saving process and enables your to cancel it at any time.

Default: On

Auto-fit column widths in imported files

If this option is checked, after opening a text, xBase or html file GS-Base will resize (used) columns to fit the widths to the field/column contents.

Selecting this option will slow down opening very large files. To speed up this resizing procedure, increase the **Number of CPUs** value.

Default: Off

Auto-fit row heights in imported files

If this option is checked, after opening a text, xBase or html file GS-Base will resize (used) rows to fit the heights to the multiline field/column contents. Checking this option is useful only if some field data may contain new-line characters (CR/LF). If **Auto-fit column widths** is not selected, this option is disabled.

Selecting this option will slow down opening very large files. To speed up this resizing procedure, increase the **Number of CPUs** value.

Default: Off

Windows opacity

Specifies the opacity of all dialog boxes displayed by GS-Base. Decreasing this value means making the windows more transparent.

Valid range: <10, 100>

Default: 100% (no transparency)

Default font

Specifies the default font name. The font specified here will be used by database tables, forms, binary edit windows and pivot tables.

Default: Arial

Table/Form Editing Settings

Use date time picker when editing dates

Check this option to use the date-time picker control instead of the plain edit control. This options applies to text fields that are formatted using one of the standard **Date** styles (not custom style patterns) and that don't use the drop-down list functionality.
Default: On

Edit LongText/Files fields in new windows

If this option is **unchecked**, trying to edit (e.g. by pressing Enter, double-clicking or pressing any character/number key) a **Long Text** or **Images/Files** field will open the LongText or Images/Files view pane to enable users to view/edit the contents.

If this option is **checked**, trying to edit a **Long Text** or **Images/Files** field will open an application associated with a given file/contents type. Choosing the "Save" command in that application will automatically update the data in GS-Base.
Depending on how that edit action is initiated, the following additional functionality is supported:

- pressing a digit key opens either the text or the n-th (1...9) object for editing
- pressing a letter key prints either the text or the n-th (a...z) object

Default: Off

Display LongText & Files field contents

If this option is unchecked, the table/form fields related to the **Long Text** and **Images/Files** fields will be displaying text containing the number of files, the total size and the last modification date/time. (If the **Images/Files** contains some shortcuts to external files, only the number of files is displayed as the label.)

If this option is checked, the table/form fields related to the **Long Text** and **Images/Files** fields will be displaying the contained images or file icons with optional labels containing (similarly optional) file names, sizes and the last modification dates/times.

Default: Off

Display description of images and files

Specifies whether images and files (icons) in tables/forms should be displayed along with their general file information which includes the file name, size and the last modification date/time.
Default: On

Display images as thumbnails

Specifies whether images and files (icons) in tables/forms should be displayed as thumbnails of the specified size.
Default: Off

LongText/Files Editing Settings

Strip off formatting in edited LongText fields

Text entered in the **Long Text** fields is saved in the plain UTF-8 *.txt text format rather than in the rich text *.rtf format.

Using plain text may noticeably reduce memory usage if there is a very large number of **Long Text** entries in the database zip file.

Changing this option affect newly entered or edited data only. Older entries remain in the previous format until they are edited.

Default: Off

Wrap text in pane

Text displayed in the **Long Text** fields is wrapped according to the width of the view pane.

Default: On

Automatically scroll bottom in LongText fields

By default, after loading LongText fields the beginning of the text displayed.

If you check this option, the text cursor will be automatically positioned at the bottom of the text.

Default: Off

Display description of images and files

Specifies whether images and files (icons) in **Images/Files** fields should be displayed along with their general file information which includes the file name, size and the last modification date.

Default: On

Display images as thumbnails

Specifies whether images and files (icons) in **Images/Files** fields should be displayed as thumbnails of the specified size.

Default: Off

Spelling Settings

Please see Spell-checking and adding Hunspell dictionaries

54 Managing the list of "recently-used" files

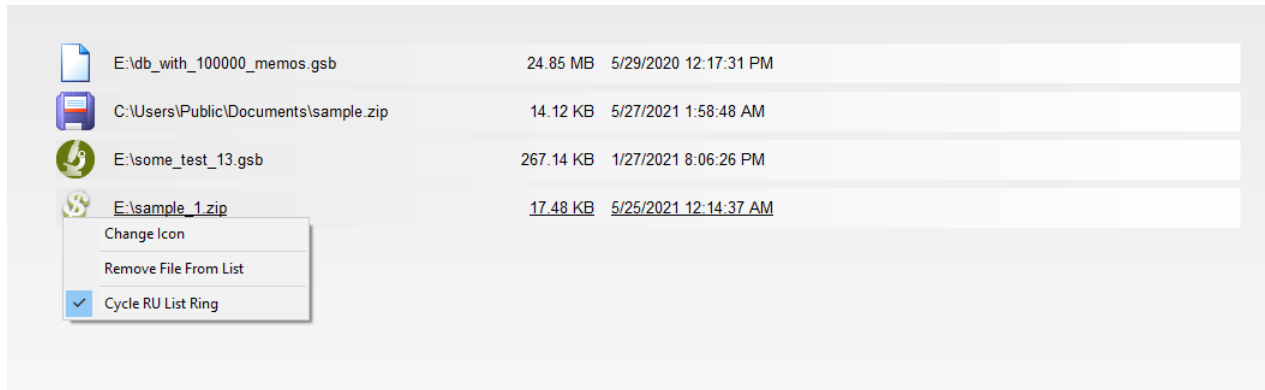
After closing the database window GS-Base displays the list of recently used databases.

To open a given file, click the corresponding link or use the cursor keys and press Space. Alternatively, drag and drop a given file (or files) into that window.

To change the icon associated with a given file, right-click the current link icon.

To delete a file from the RU list, right-click it and use the **Remove** command from the context menu.

If you want to preserve the displaying order of the RU file list right-click the list and unchecked the **Cycle RU List Ring** option. Otherwise the recently opened database will be moved to the top.



55 JScript and VBScript scripting

Sample JScripts scripts

- Creating and saving a new database
- Adding and removing records, updating calculated fields
- Adding, removing and renaming tables and subfolders
- Importing tables from GS-Base databases, text and Excel files
- Merging records from all text and Excel files in a folder
- Formatting record fields
- Setting column/field widths and row heights
- Browsing the folders/tables tree
- Searching and sorting
- Predefined searching
- File password protection
- Enabling sharing databases and text, Excel and other files
- Opening and saving text files
- Error handling

List of all properties and functions

- File/database opening
- Saving databases
- Saving databases as new files
- Saving recordsets
- Importing tables databases, text and Excel files
- Performing JOIN operations for tables in the same database file
- Merging/adding records from other files
- Switching the active table
- Record counters
- Searching
- Adding, modifying and removing fields
- Browsing the database folder and tables tree structure

Editing records
Inserting series
Changing column widths and row heights
Setting record flags
Setting database passwords for GS-Base *.gsb/*.zip files
File information and access
Updating all calculation formulas
Input/output UI functions
Window functions
Obtaining the last error details

To make scripting accessible, you need to perform one-time GS-Base scripting registration using the "Settings > Register GS-Base Scripting" command in the admin mode. (Right-click the GS-Base shortcut or file and choose "Run As Administrator", register, close GS-Base). This registers GS-Base in Windows registry as a program that can be scripted not only in GS-Base, but also using external programming tools. (The "Unregister(...) " command removes the added entries.)

You can create your scripts either as global scripts saved in the program settings and available to all databases or you can create scripts stored in a given GS-Base database and available only after you open that database.
To create these scripts use the "File > Application Scripts" and "File > Database Scripts" commands.

The "(...) Scripts" dialog box enables you to organize your scripts in subfolders, test them to locate errors, copy/import/export them etc.

Sample "Scripts" screen.

Notes:

- If a global script is executed when there is no open database, it should start with one of the database Open(...)/New(..) functions
- If a database is already loaded, an executed script, either global or local, automatically refers to that database and using the Open(...) functions with the same database path has no effect.

55.1 Creating and saving a new database

```
//// a new database with a the "table1" table  
GSBase.NewDatabase("table1");
```

```
// insert another "table1" in the root ("") folder at the bottom;
```

```

// as "table1" already exists, the uniqueName will contain a unique
// name table1(1)...table1(n) that GS-Base will create and use instead;
var uniqueName = GSBase.InsertDatabaseItem("", 1, "table1");

var field = GSBase.CreateFieldParams();

// add one Text field to the currently selected table;
//
// adding and importing a table make it 'selected' automatically;
// deleting tables may result in a new selection;
// SetActiveTable() and GetActiveTable() set and retrieve the selection
//
field.name = "field1";
// T - text field
// N - numeric field
// M - long text / Memo
// O - objects (images, files etc.)
// C - code field (long text with specific syntax highlighting)
field.type = 'T';
GSBase.AppendField(field);

//add one Number field with range validation
//
field.Reset();
field.name = "field2";
field.type = 'N';
field.formula = "=(field2 > 1) * (field2 < 10)";
// 1 - calculation formula/calculated field
// 2 - validation formula
// 3 - conversion formula
// 4 - default value
// 5 - incremented maximum
field.formulaType = 2;
GSBase.AppendField(field);

//add one calculated Number field
//
field.Reset();
field.name = "field3";
field.type = 'N';
field.formula = "=field2 * 10";
field.formulaType = 1;
GSBase.AppendField(field);

//insert one more Text field at the beginning of the record
field.Reset();
field.name = "field4";
field.type = 'T';
GSBase.InsertField(1, field);

//add one Code field that uses the "cpp" syntax highlighting
field.Reset();
field.name = "field5";
field.type = 'C';
//subtypes same as in the "Field Setup": "cpp", "assembler", "php"...
field.subtype = "cpp";
GSBase.AppendField(field);

```

```

//choose to use the standard zip (zip32) file format for the new database;
the default value for new files is "true" ("use zip64");
for existing database files their the default value is the one that was used
previously;

//GSBase.zip64 = true;
GSBase.zip64 = false;

//save a new database (with one table and no records so far);
GSBase.SaveDatabaseAs("e:\\test_dbase.gsb");
GSBase.Close();

```

55.2 Adding and removing records, updating calculated fields

```

var password = "fhE4!lko"
GSBase.OpenDatabase("e:\\test_dbase.gsb", password);

//all editing actions always refer to the currently selected table;
//once a table is selected (and the database is saved), it remains selected
till you change this;
//
if (GSBase.GetActiveTable() != "table1")
    GSBase.SetActiveTable("table1");

//as the table contains the calculated "field3", for performance reasons turn
off
//recalculation of each row which would otherwise occur after each of the
10,000 modifications below;
//to restore the default automatic updating use "automatic" or re-open the
database;
GSBase.updateMode = "manual";

//NOTE: InsertText() and InsertNumber() do exactly what plain entering data
does
//which means they set up Undo information and refreshes the screen which
makes them slow. If you need
//to fill millions of fields instantly, use the Insert(...)Series() functions
family instead.

//NOTE: for best performance, when inserting a large series of data, always
fill the table fields
//in the "top to bottom" (and "left to right") order. The "bottom to top"
order may
//be considerably slower.

//insert the following date string in the 1st record field in 100 records;
var today = "2021-01-15";

```

```

for (i = 1; i <= 100; ++i)
{
    GSBase.InsertText(i, 1, today);
}

//insert some numbers in the 3rd record field in the first 100 records
for (i = 1; i <= 100; ++i)
{
    GSBase.InsertNumber(i, 3, 2.0 + i%8);
}

//update all calculated fields in this table using 4 processor cores
GSBase.UpdateTable(4);

//get the sum of the 4th field for the entire current record set
var counter = GSBase.GetRecordSetCount();
var sum = 0;
for (i = 1; i <= counter; ++i)
    sum = sum + GSBase.GetNumber(i, 4);

//clear the 3rd field in records 11 to 21
GSBase.ClearRange(11, 3, 21, 3);

//update all "field3" calculated fields as the "manual" update mode was set
earlier
GSBase.UpdateTable(4);

//remove record 1
GSBase.RemoveRecords(1, 1);
//remove records 2 to 3
GSBase.RemoveRecords(2, 3);
//remove record 11
GSBase.RemoveRecords(11, 11);

// save using the current file format and file format options
GSBase.Save();
GSBase.Close();

```

55.3 Adding, removing and renaming tables and folders

```

GSBase.OpenDatabase("e:\\test_dbase.gsb", "");

//notes:
//insert a new folder "folder1\\" in the root folder at the bottom;
//if "folder1" already exists at that level, the uniqueName will contain a
unique
//name folder1(1)...folder1(n) that GS-Base will create and use instead;
var uniqueName = GSBase.InsertDatabaseItem("", 1, "folder1\\");

```

```

//insert folder2 at the top
GSBase.InsertDatabaseItem("", 0, "folder2\\");

GSBase.InsertDatabaseItem("\\folder1\\", 1, "folder1\\");
GSBase.InsertDatabaseItem("folder1", 1, "folder1\\");

//rename the "text_abc(2)" table in the "\folder2" folder to "test_abc_b";
GSBase.RenameDatabaseItem("\\folder2\\test_abc(2)", "test_abc_b");
//delete the entire "folder2" folder
GSBase.DeleteDatabaseItem("\\folder2\\");

//delete the "\folder1\test_abc_b" table
GSBase.DeleteDatabaseItem("\\folder1\\test_abc_b");

GSBase.Save();
GSBase.Close();

```

55.4 Importing tables from GS-Base databases, text files and Excel workbooks

```

//import the "product" table from the sample.zip database and insert it in
the root folder
GSBase.ImportTable("e:\\sample.zip", "products", "", "\\");

//import to the "folder1" subfolder
GSBase.ImportTable("c:\\sample2.zip", "products2", "pass4563word!",
"folder1");

//import the first table from file_a.xlsx and insert it in the root folder
//first row in file_a.xlsx contains field names
GSBase.ImportExcelTable("e:\\excel_data\\file_a.xlsx", "", 1, "");
//import the first table from file_c.xlsx and insert it in the root folder
//first row in file_a.xlsx doesn't contain field names
GSBase.ImportExcelTable("e:\\excel_data\\file_c.xlsx", "", 0, "");
//import the first table from file_a.xlsx and insert it in the root folder
//first row in file_a.xlsx doesn't contain field names
GSBase.ImportExcelTable("e:\\excel_data\\file_i.xlsx", "", 0, "");

//create text file parameters
var textParams = GSBase.CreateTextParams();
//use "|" as the field separator ("," is the default one)
textParams.separator = "|";

//import a new table from the "text_abc.txt" text file and place it in the
"folder2" folder
GSBase.ImportTextTable("c:\\test_abc.txt", "", textParams, "\\folder2\\");
//import it again (the name of the imported table will be modified to
"text_abc(1)")

```

```

GSBase.ImportTextTable("c:\\test_abc.txt", "", textParams, "\\folder2\\");
//import it again (the name of the imported table will be modified to
"text_abc(2)")
GSBase.ImportTextTable("c:\\test_abc.txt", "some_name", textParams,
"\\folder2");

GSBase.Save();
GSBase.Close();

```

55.5 Merging records from all text and Excel files in a folder

```

GSBase.OpenDatabase("e:\\test_dbase.gsb", "");

var mergeParams = GSBase.CreateMergeParams();
// merge record e.g. from report2000.txt, report2001.txt, ..., report2023.txt
mergeParams.path = "e:\\gsb19_test\\report20???.txt";
// don't perform field names matching, add fields "as is"
// note: field types must match, text fields can't be copied to numeric
fields
mergeParams.matchFieldNames = false;

var textParams = GSBase.CreateTextParams();
textParams.separator = "|";
textParams.encoding = "windows";

GSBase.MergeRecordsFromTextFile(mergeParams, textParams);

// if there are calculated fields, update all records using 4 processor cores
GSBase.UpdateTable(4);

// you can also only perform more complex merging using the "mergeType"
property:
// mergeParams.mergeType=0 - merge records unconditionally (default value)
// mergeParams.mergeType=1 - merge records only with new values of the
mergeParams.slaveIndex field from merged files
// mergeParams.mergeType=2 - update records in the main table where the
specified fields in the main/master table and in the merge tables are the
same, mergeParams.masterIndex = mergeParams.slaveIndex
// mergeParams.mergeType=3 - delete records from the main table where the
specified fields in the main/master table and in the merge tables are the
same, mergeParams.masterIndex = mergeParams.slaveIndex

// For Excel and other file format use:
// MergeRecords(mergeParams)
// MergeRecordsFromTextFile(mergeParams, textParams)
// MergeRecordsFromExcelFile(mergeParams, namesInFirstRow)
// MergeRecordsFromXBaseFile(mergeParams, xBaseParams)

```

```
// MergeRecordsFromMySQLFile (mergeParams)
```

Record merging functions

```
GSBase.Save();  
GSBase.Close();
```

55.6 Formatting fields

```
GSBase.OpenDatabase("e:\\test_dbase.gsb", 0);  
  
//select the "table1" table in the root folder  
GSBase.SetActiveTable("\\table1");  
  
//create formatting settings  
var format = GSBase.CreateFormatParams();  
  
//set the currency format: variable/automatic number of decimals and the  
exponent value  
//parameters:  
//1. decimals: 0 - 14 | "auto"  
//2. currency position: "$1.1" | "$ 1.1" | "1.1$" | "1.1 $"  
//3. currency symbol: "$", "GBP" etc.  
//4. true - use curly braces for negative values  
//5. true - use red color for negative values  
//  
format.SetCurrencyFormat("2", "$1.1", "gbp", true, true);  
  
// other style examples:  
//  
//set the scientific style: 5 decimal digits and the fixed "07" exponent  
//  
//----- format.SetScientificFormat("5", "07");  
//  
//set the scientific style: variable/automatic number of decimals and the  
exponent value  
//  
//----- format.SetScientificFormat("auto", "auto");  
//  
//  
//set the accounting style  
//parameters:  
//1. decimals: 0 - 14 | "auto"  
//2. currency symbol: "$", "GBP" etc.  
//  
//----- format.SetAccountingFormat("2", "gbp");  
//  
//
```

```

//set the fractional format
//parameters:
//1. if the 2nd parameter is "false", (1) is a denominator value 2, 3, ...,
n;
//   if the 2nd parameter is "true", (1) is a fixed number of denominator
digits 1...14
//2. ...
//
//----- format.SetFractionFormat(2, true);
//
//
//set the general number format
//parameters:
//1. decimals: 0 - 14 | "auto"
//2. leading zeroes: 0 - 14
//3. true - use curly braces for negative values
//4. true - use red color for negative values
//5. true - use the thousand separator
//
//----- format.SetGeneralNumberFormat("auto", 5, false, false, true);
//

GSBase.SetFieldFormat(3, format);

//set the date format
//parameters:
//1. a date pattern: same as in the "Format > Style" window
//2. 1 - always switch to the current Windows system day/month order
("m/d..." | "d/m...")
format.SetDateFormat("m/d/yyyy", 1);
GSBase.SetFieldFormat(1, format);

//clear the previously set information
format.Reset();

format.fontSize = 14;
format.boldFont = true;
GSBase.SetFieldFormat(3, format);

GSBase.Save();
GSBase.Close();

```

55.7 Setting column/field widths and row heights

```

GSBase.OpenDatabase("e:\\test_dbase.gsb", 0);

if (GSBase.GetActiveTable() != "table1")
    GSBase.SetActiveTable("table1");

```

```

//get the 1st column/field width in screen pixels
var width = GSBase.GetColumnWidth(1);

//set a new width
width += 50;
GSBase.SetColumnWidth(1, width);

//fit the 3rd column/field width to the data in that column/field
GSBase.FitColumnWidth(3, 3);

//get the 9th row/record height in screen pixels: typically it should be 25px
var height = GSBase.GetRowHeight(9);

//set a new height
height += 20;
GSBase.SetRowHeight(9, height);

//check the "auto-height" state of the 9th row
var autoHeight = GSBase.GetAutoRowHeight(9);

//restore on the "auto-height" state for the 9th row
GSBase.SetAutoRowHeight(9, 9, true);

//should be true now
autoHeight = GSBase.GetAutoRowHeight(9);

//it should be 25px again
height = GSBase.GetRowHeight(9);

GSBase.Save();
GSBase.Close();

```

55.8 Browsing the folders/tables tree

```

GSBase.OpenDatabase("e:\\test_dbase.gsb", "");

//1st method

//get the total number of tables and folders in the main/root folder
(including nested folders)
var counter = GSBase.GetDatabaseItemCount();

//iterate through all table/folders
for (i = 1; i <= counter; ++i)
{
    var path = GSBase.GetDatabaseItem(i);
    if (path.length && path.charAt(path.length - 1) == '\\')

```

```

        {
            //folder
            var ret = GSBase.MessageBox("folder - " + path, "ok", 1,
"information");
        }
        else
        {
            //table
            var ret = GSBase.MessageBox("table - " + path, "ok", 1,
"information");
        }

        var ret = GSBase.MessageBox(path, "ok", 1, "information");
    }

//2nd method

//get the first "child" element in the specified folder, e.g. the main/"root"
folder
//var path = GSBase.GetFirstDatabaseItem("\\folder2(1)\\");
var path = GSBase.GetFirstDatabaseItem("\\");

//iterate through direct "child" elements of the specified folder (not
expanding nested folders)
while (path.length)
{
    if (path.charAt(path.length - 1) == '\\')
    {
        //folder
        var ret = GSBase.MessageBox("folder - " + path, "ok", 1,
"information");
    }
    else
    {
        //table
        var ret = GSBase.MessageBox("table - " + path, "ok", 1,
"information");
    }
    path = GSBase.GetNextDatabaseItem(path);
}

GSBase.Close();

```

55.9 Searching and sorting

```

GSBase.OpenDatabase("e:\\test_dbase.gsb", "");

//select the "table1" table in the root folder

```

```

GSBase.SetActiveTable("\\products");

var field = GSBase.CreateFieldParams();

//find the "ProductName" and "UnitPrice" fields;
//filter "ProductName" and sort "UnitPrice"

//reset the previous sorting indices - resetting should be used before
calling "put_sortIndex()"
GSBase.ResetSorting();

var iname = 0;
var iprice = 0;

for (i = 1; i <= GSBase.GetFieldCount() && (!iname || !iprice); ++i)
{
    GSBase.GetField(i, field);
    if (!iname && field.name == "ProductName")
    {
        //set the "\\Ai" RegEx filter for "ProductName" (=search for
names starting with "I");
        //searching is performed automatically if a call to the
"SetField" changes the "filter" value;
        //note: in this version the filter type is always "RegEx"
        field.filter = "\\Ai";

        GSBase.SetField(iname = i, field);
    }
    if (!iprice && field.name == "UnitPrice")
    {
        //set the 1 as the sorting index for "UnitPrice";
        //sorting is performed automatically after a call to
"SetField" if the "sorting" index is modified;
        //to create a compound sorting index, use subsequent numbers
(2, 3...) for further fields;
        //note: using an index not in that strictly incremented manner
causes an error;

        field.sortIndex = 1;

        // 'A' - ascending order, 'D' - descending order
        field.sortOrder = 'A';

        GSBase.SetField(iprice = i, field);
    }
}

GSBase.Save();
GSBase.Close();

```

55.10 Predefined searching

```
GSBase.OpenDatabase("e:\\test_dbase.gsb", "");

//select the "table1" table in the root folder
GSBase.SetActiveTable("\\products");

//find duplicates in the 5th field
GSBase.FindDuplicates(5, 5, 1, false, false);

//check the results
var counter1 = GSBase.GetRecordTotalCount();
var counter2 = GSBase.GetRecordSetCount();

//check the results
counter1 = GSBase.GetRecordTotalCount();
counter2 = GSBase.GetRecordSetCount();

//find records with the flag "1"
GSBase.FindFlagged(1);

// ...

//find the records not included in the current record set
GSBase.FindComplement();

// ...

//clear all filters and display all records
GSBase.FindAll();

// ...

GSBase.Close();
```

55.11 File password protection

```
//set a password
GSBase.OpenDatabase("e:\\test_dbase.gsb", "");

GSBase.SetFilePassword(true, "blowfish", "", "rocc4545");

GSBase.Save();
GSBase.Close();

//open a password-protected database
GSBase.OpenDatabase("e:\\test_dbase.gsb", "rocc4545");
//...
//GSBase.Save();
GSBase.Close();
```

```
//remove password protection
GSBase.OpenDatabase("e:\\test_dbase.gsb", "rocc4545");

GSBase.SetFilePassword(false, "blowfish", "rocc4545", "");

GSBase.Save();
GSBase.Close();
```

55.12 Enabling sharing databases and text, Excel and other files

```
var fpath = GSBase.ReleaseFile()
GSBase.MessageBox("File closed: " + fpath, "ok", 1, "information");

//You can keep on viewing/working with this file in GS-Base as usual (as it's
in RAM).
//It can be edited in other programs.

//At any moment you can execute the following script, which will check for
updates:

if (GSBase.AttachFile(fpath) == 1)
{
    // AttachFile returned 1, so the file was modified
    // after ReleaseFile(), you can ask whether reload
    // it now or later
    GSBase.Reload();
}
else
{
    // no changes detected, so leave the file accessible
    // for any other programs.
    GSBase.ReleaseFile();
}
```

55.13 Opening and saving text files

```
// 1. Exporting the current record set to a text file
// -----
```

```

GSBase.OpenDatabase("e:\\test_dbase.gsb", "");

//select the "table1" table in the root folder
GSBase.SetActiveTable("\\products");

//create text file parameters
var textParams = GSBase.CreateTextParams();

//use ";" as the field separator; the default value is ","
textParams.separator = ";";

//use of separators can be turned off; the default value is true
//textParams.useSeparator = false;

//use "'" as the quoting symbol; the default value is ""
textParams.quotingSymbol = "'";

//use of quoting symbols can be turned off; the default value is true
//textParams.useQuoting = false;

//save field names in the first row; the default value is true
textParams.fieldNames = true;

//change the text encoding: "utf8" | "windows" | "dos"; the default value is
"utf8"
textParams.encoding = "utf8";

//export the current table to a text file;
//"dbase" remains the originally opened database and can edited further as
usual
GSBase.SaveRecordSetAsText("e:\\test_b.txt", textParams);

// 2. Opening and saving a text file
// -----

//re-use the above "txt" settings and add new ones

//if a column contains textual representations of numbers, try converting
them to a number field
textParams.parseNumbers = true;

//if a column contains textual representations of dates in various formats,
convert these strings
//to the generic "DT" W3 text representation of dates in GS-Base (please see
the "data types" help topic for details).
textParams.parseDates = true;

GSBase.OpenTextFile("e:\\test_b.txt", textParams);

var fcounter = GSBase.GetFieldCount();

var field = GSBase.CreateFieldParams();

GSBase.GetField(1, field);
//

```

```

// ...perform any editing, field changes etc.
//

//save the edited text file
GSBase.Save();
GSBase.Close();

// 3. Opening a text file and saving it as a database
// -----

//re-use the above "txt"
textParams.parseNumbers = false;

GSBase.OpenTextFile("e:\\test_b.txt", txt);

//SaveDatabaseAs changes "textFile" to a database with the path given below;
//the text file is closed
GSBase.SaveDatabaseAs("e:\\sample_b.gsb");

//
// ...perform any editing, field actions etc. with "e:\\sample_b.gsb"
//

GSBase.Close();

```

55.14 Error handling

```

GSBase.OpenDatabase("e:\\test_dbase.gsb", "");

//
// ...
//

// if a COM function returns an error other than E_OUTOFMEMORY,
// additional error information can be obtained via the "lastError" property;
var code = GSBase.lastError;

// 21 // Out of memory while creating/editing worksheet data
// 22 // A database must contain at least one table
// 37 // Invalid password.
// 53 // Not allowed field type change - e.g. conversion from "Files/Images"
// to "Number"
// 61 // Can't open the specified file
// 62 // Can't open or create the specified file
// 63 // Error while closing the specified file
// 64 // Error while repositioning a file pointer
// 65 // Error while reading from a file
// 66 // Error while writing to a file

```

```

// 67 // Error while deleting a file
// 68 // Error while checking the file size/info
// 69 // Error while allocating a file read/write buffer
// 70 // Can't open the specified file. File in use.
// 71 // Error while decrypting a file.
// 72 // Error while encrypting a file.
// 73 // Error while reading binary fields."
// 79 // Can't find the manifest file or its content is incorrect
// 80 // The file format requires a newer GS-Base version.
// 119 // Too many zip streams in a standard zip (zip32) file
// 120 // Inconsistent zip stream state
// 121 // Corrupted zip stream data
// 122 // Out of memory while processing a zip stream
// 123 // Unexpected end of zip stream
// 125 // Unknown zlib error
// 131 // Some data in the file can't be converted to numeric field values.

GSBase.Close();

```

55.15 File/database opening functions

| | |
|--|--|
| NewDatabase(tableName) | |
| OpenDatabase(path, password) | If the database is not encrypted, password should be ""/null. |
| OpenTextFile(path, showDialogBox, textParams) | showDialogBox = 1 to show the standard "Open Text File" dialog box. textParams - an object created with GSBase.CreateTextParams(). |
| OpenExcelFile(path, showDialogBox, mergeType, options) | showDialogBox = 1 to show the standard "Open Excel File" dialog box. mergeType - represents the 0...3 "merge" radio buttons from the "Open Excel File" dialog box. options - a combination of 1, 2, 4 (1 - field names in the 1st worksheet, 2 - field names in every worksheet, 4 - restore GS-Base table/folder tree paths). |
| OpenHtmlFile(path, fnames) | fnames = 1 if there are field names in the 1st row of the html table (note: the table must have the "id" attribute set to "gsbase"). |
| OpenXBaseFile(path, showDialogBox, xBaseParams) | showDialogBox = 1 to show the standard "Open xBase File" dialog box. |

| | |
|---------------------|--|
| | xBaseParams - an object created with GSBBase.CreateXBaseParams(). |
| OpenMySQLFile(path) | |
| Close() | Closing is also automatic when subsequent "open" methods are used. |
| Reload() | Reloads the opened file using originally used file format settings |

params = GSBBase.CreateTextParams() - settings corresponding to the "Open Text File" dialog box options

| | |
|--------------------------|-------------------------------------|
| params.useSeparator | 0, 1 |
| params.separator | any character |
| params.fixedFieldWidths | for example, "10,20,30,15" |
| params.encoding | "utf8", "utf16", "windows" |
| params.useQuoting | 0, 1 |
| params.quotingSymbol | any character |
| params.fieldNames | 0, 1 (field names in the first row) |
| params.parseNumbers | 0, 1 |
| params.parseDates | 0, 1 |
| params.saveLongTextAsZip | 0, 1 |
| params.saveObjectsAsZip | 0, 1 |
| params.autoFitColumns | 0, 1 |

params = GSBBase.CreateXBaseParams() - settings corresponding to the "Open xBase File" dialog box options

| | |
|-----------------|--|
| params.format | "dbaseIII", "dbaseIV", "foxpro", "clipper" |
| params.encoding | "windows", "dos" |

| | |
|--|------------------------------------|
| params.GetFieldCount() | |
| params.SetField(index, name, type, length, decimals) | type: "C", "N", "F", "L", "D", "M" |
| params.AddField(name, type, length, decimals) | |
| params.params.InsertField(index, name, type, length, decimals) | |
| params.GetFieldName(index) | index: 1...number of fields |
| params.GetFieldType(index) | |
| params.GetFieldLength(index) | |
| params.GetFieldDecimals(index) | 0, 1 |
| params.DeleteField(index) | |

55.16 Saving recordsets

| | |
|--|---|
| SaveRecordSetAs(path, password) | |
| SaveRecordSetAsText(path, textParams) | If the file is not to be encrypted, password should be ""/null. |
| SaveRecordSetAsExcel(path, split, fnames, saveZip) | |
| SaveRecordSetAsXBase(path, format, xBaseParams) | |
| SaveRecordSetAsMySQL(path) | |
| SaveRecordSetAsPDF(path) | |
| zip64 | 0, 1 - specify whether ZIP64 or the older standard Windows ZIP32 should be used for *.gsb and *.xlsx files. |

Note: Unlike "SaveDatabaseAs" methods, recordset saving methods do not change the current database path or state.

55.17 Saving databases

| | |
|---------------------------------------|---|
| Save() | Saves the current file using the current file format and its current file format settings |
| SaveTextFile(textParams) | Enables you to modify settings when saving a previously opened text files. |
| SaveExcelFile(split, fnames, saveZip) | |
| SaveXBaseFile(xBaseParams) | |
| SaveMySLQFile() | |

55.18 Saving databases as new files

| | |
|---|---|
| SaveDatabaseAs(path) | |
| SaveDatabaseAsExcel(path, int split, fnames, saveZip) | |
| SaveDatabaseAsMySQL(path) | |
| zip64 | 0, 1 - specify whether ZIP64 or the older standard Windows ZIP32 should be used for *.gsb and *.xlsx files. |

Note: The above methods change the current database path, saving a new copy of a given database.

55.19 Importing tables databases, text and Excel files

| | |
|--|--|
| <code>ImportTable(filePath, tablePath, password, folder)</code> | If <code>tablePath=""</code> , the first found table from "filePath" is imported The target "folder" must exist or the parameter must be empty. If <code>folder=""</code> , the imported table is inserted in the main/root folder |
| <code>ImportTextTable(textFilePath, tableName, textParams, folder)</code> | If <code>tableName=""</code> , the inserted table name will be the same as the text file name The target "folder" must exist or the parameter must be empty. If <code>folder=""</code> , the imported table is inserted in the main/root folder |
| <code>ImportExcelTable(excelFilePath, tableName, namesInFirstRow, folder)</code> | |

55.20 Performing JOIN operations for tables in the same database file

| | |
|--|--|
| <code>MergeTable(masterField, tablePath, slaveField, allowDuplicates)</code> | <code>masterField</code> - 1-based index of the field in the current table <code>tablePath</code> - full path of the joined table <code>slaveField</code> - 1-based index of the related field in the joined table <code>allowDuplicates</code> - 0, 1: specifies whether duplicated values in the joined table should result in adding multiple joined records in the result table |
|--|--|

55.21 Merging/adding records from other files

| | |
|--|--|
| <code>MergeRecords(mergeParams)</code> | Merge records from another *.gsb database <code>mergeParams</code> - an object created with <code>GSBase.CreateMergeParams()</code> . |
| <code>MergeRecordsFromTextFile(mergeParams, textParams)</code> | <code>textParams</code> - an object created with <code>GSBase.CreateTextParams()</code> |

| | |
|---|---|
| MergeRecordsFromExcelFile(mergeParams, namesInFirstRow) | |
| MergeRecordsFromXBaseFile(mergeParams, xBaseParams) | xBaseParams - an object created with GSBASE.CreateXBaseParams() |
| MergeRecordsFromMySQLFile(mergeParams) | |

params = GSBASE.CreateMergeParams() - settings corresponding to the record merging options

| | |
|------------------------|---|
| params.path | full file path; the file name contain wildcard characters to enable mass merge of files from a given folder, for example, "e:\\folder*.csv", "f:\\folder*20??.gsb", "c:\\report*.xlsx" |
| params.table | full table path for *.gsb, *.xlsx and *.sql files; if it's empty the first table in the specified file is used. |
| params.masterField | 1-based index of the field in the current table; used only if the mergeType <> 0 |
| params.slaveField | 1-based index of the related field in the merged table; used only if the mergeType <> 0 |
| params.mergeType | 0 - add all records, 1 - add records where the "slaveField" values don't exist in the main table 2 - update records in the main table where the "slaveField" values exist in the main table 3 - delete records from the main table where the "slaveField" values exist in the main table |
| params.matchFieldNames | 0, 1; used only if the mergeType=0 and causes adding records unconditionally, without matching field names in both tables |
| params.ignoreEmpty | 0, 1; used only if the mergeType=2 and causes not using empty fields |
| params.enableUndo | 0, 1; specifies whether the UI Undo command should be enabled for merging; it's 0 by default as the undo information can multiple the memory usage for these actions |

55.22 Switching the active table

| | |
|----------------------|------------------------|
| SetActiveTable(path) | path - full table path |
| GetActiveTable() | |

55.23 Record counters

| |
|-----------------------|
| GetRecordTotalCount() |
| GetRecordSetCount() |
| GetFieldCount() |

55.24 Searching

| | |
|---|--|
| FindDuplicates(fieldFrom, fieldTo, searchAll, type, sortResults) | fieldFrom and fieldTo are 1-based field indices. Field in this range are treated as the search key. searchAll: 0, 1 - if it's 1, GS-Base will search for duplicates in the current filtered record set; otherwise the entire table is searched. type=0 - show all duplicates, type=1 - show first records from each group of duplicates, type=2 - show the last record from each group of duplicates, type=3 - all duplicates except the first from each group, type=4 - all duplicates except the last one from each group sortResults: 0,1 - sort the obtained records using the above key. |
| FindUnique(fieldFrom, fieldTo, searchAll, type, sortResults, useCounters, counterIndex) | fieldFrom and fieldTo are 1-based field indices. Field in this range are treated as the search key. searchAll: 0, 1 - if it's 1, GS-Base will search for duplicates in the current filtered record set; otherwise the entire table is searched. type=0 - if there are duplicates, show first records |

| | |
|---|---|
| | from each group of duplicates, type=1 - if there are duplicates, show the last record from each group of duplicates, useCounters: 0, 1 - use 1 to specify a field where the number of occurrences will be saved, counterIndex: 1-based index of a numeric field, where the number of occurrences will be saved, sortResults: 0,1 - sort the obtained records using the above key. |
| FindQuartile(field, searchAll, statAll, type) | The "type" parameter can be a mix of the following values: 1: 1st, 2: 2nd, 4: 3rd, 8: 4th, 16: below mean, 32: above mean statAll: 0, 1 - if it's 1, the statistics will be calculated for the entire table, otherwise for the current recordset. |
| FindRandom(n) | n randomly selected records from the current recordset |
| FindFlagged(flag) | flg: a flag index from 1 to the number of created flags |
| FindComplement() | Shows all record not included in the current recordset |
| FindAll() | Shows all records. |

Note: regular field filter are set using the SetField() function.

55.25 Adding, modifying and removing fields

| | |
|---------------------------------|---|
| AppendField(fieldParams) | fieldParams - an object created with GSBase.CreateFieldParams() |
| InsertField(field, fieldParams) | |
| AppendField(fieldParams) | |
| RemoveField(field) | |

| | |
|------------------------------|--|
| GetField(field, fieldParams) | |
| SetField(field, fieldParams) | SetField() triggers filtering and/or sorting if the respective data was set in the fieldParams |
| ConvertField(field, type) | |
| ResetSorting() | Clears any currently defined single or multiple sort key |

params = GSBase.CreateFieldParams() - settings corresponding to the record field definition

| | |
|------------------|--|
| params.name | |
| params.type | "T" - textual "N" - numeric "M" - LongText "C" - Code "O" - Images/Files |
| params.subType | used for the "C" field type: "cpp", "cpp/rc", "cpp/idl(odl)", "c#", "assembler", "java", "jscript", "vbscript/vb.net", "html", "xml", "flash", "php", "python", "perl", "powershell", "sql". Choosing the subtype causes applying the correct language syntax and keyword coloring. |
| params.hlink | 0, 1 |
| params.sortIndex | 1 for single key sorting or 1-based index for multi-key sorting |
| params.sortOrder | "A" or "D" |
| params.formula | e.g. "=some_numeric_field_name + 10" |

| | |
|--------------------|---|
| params.formulaType | 1 - calculation formula/calculated field 2 - validation formula 3 - conversion formula 4 - default value 5 - incremented maximum |
| params.filter | sets the current filter expression; if it changes, the SetField() method will trigger searching |
| params.filterType | 1 - regex 2 - plain text pattern 3 - equal 4 - not equal 5 - greater than 6 - less than 7 - between 8 - and 9 - or 10 - is empty 11 - is similar 12 - flag index |
| params.matchCase | 0, 1 |
| params.matchWords | 0, 1; match whole words for full text plain searching |
| params.Reset() | Clear all set options and flags |

55.26 Browsing the database folder and tables tree structure

| | |
|--------------------------------|--|
| GetDatabaseItemCount() | The total number of tables and folders in a database file. |
| GetDatabaseItem(index) | Return full tables and folders paths |
| GetFirstDatabaseItem(folder) | |
| GetNextDatabaseItem(prevName) | |
| RenameDatabaseItem(path, name) | |
| DeleteDatabaseItem(path) | |

55.27 Editing records

NOTE: InsertText() and InsertNumber() do exactly what plain entering data does which means they set up Undo information and refreshes the screen which makes them slow. If you need to fill millions of fields instantly, use the Insert(...)Series() functions family instead.

| |
|--|
| RemoveRecords(recordFrom, recordTo) |
| InsertRecords(recordFrom, recordTo) |
| InsertText(record, field, text) |
| GetText(record, field) |
| InsertNumber(record, field, number) |
| GetNumber(record, field) |
| Clear(record, field) |
| ClearRange(record1, field1, record2, field2) |

Note: all record numbers are relative to the current, filtered and sorted (or not) record set. They are equal to physical cardinal record numbers in a table only if they are not active searches or sort keys.

55.28 Inserting series

| | |
|--|--|
| InsertSeries(field, recordFrom, recordTo, copy) | copy=0 - numeric or date/time sequence copy=1 - copying the top item within the selected range See: Inserting series |
| InsertRecurrenceSeries(field, recordFrom, recordTo, formula) | See: Inserting series |

| | |
|---|-----------------------|
| InsertCustomSeries(field, recordFrom, recordTo, series) | See: Inserting series |
| InsertRandomSeries(field, recordFrom, recordTo, distribution, param1, param2) | See: Inserting series |

Note: all record numbers are relative to the current, filtered and sorted (or not) record set. They are equal to physical cardinal record numbers in a table only if they are not active searches or sort keys.

55.29 Changing column widths and row heights

| | |
|---|-----------|
| GetColumnWidth(field) | |
| SetColumnWidth(field, width) | |
| FitColumnWidth(fieldFrom, fieldTo) | |
| GetRowHeight(record) | |
| SetRowHeight(record, height) | |
| GetNumber(record, field) | |
| GetAutoRowHeight(record) | |
| SetAutoRowHeight(recordFrom, recordTo, val) | val: 0, 1 |

Note: all record numbers are relative to the current, filtered and sorted (or not) record set. They are equal to physical cardinal record numbers in a table only if they are not active searches or sort keys.

55.30 Formatting fields

| | |
|----------------------|--|
| CreateFormatParams() | |
|----------------------|--|

| | |
|-------------------------------------|--|
| GetFieldFormat(field, formatParams) | field - 1-based field index, formatParams - an object created with GSBase.CreateFormatParams() |
| SetFieldFormat(field, formatParams) | |

params = GSBase.CreateFormatParams() - format/style settings

| | |
|---|--|
| params.SetGeneralNumberFormat(decimals, zeroes, brackets, inRed, separators, scaling) | 1. decimals: 0 - 14 "auto" 2. leading zeroes: 0 - 14 3. true - use curly braces for negative values 4. true - use red color for negative values 5. true - use the thousand separator For example: params.SetGeneralNumberFormat("auto", 5, false, false, true); |
| params.SetCurrencyFormat(decimals, position, symbol, brackets, inRed, scaling) | 1. decimals: 0 - 14 "auto" 2. currency position: "\$1.1" "\$ 1.1" "1.1\$" "1.1 \$" 3. currency symbol: "\$", "GBP" etc. 4. true - use curly braces for negative values 5. true - use red color for negative values For example: params.SetCurrencyFormat("2", "\$1.1", "gbp", true, true); |
| params.SetAccountingFormat(decimals, symbol, scaling) | 1. decimals: 0 - 14 "auto" 2. currency symbol: "\$", "GBP" etc. For example: params.SetAccountingFormat("2", "gbp"); |
| params.SetDateFormat(pattern, systemOrder) | 1. a date pattern: same as in the "Format > Style" window 2. 1 - always switch to the current Windows system day/month order ("m/d..." "d/m...") For example: params.SetDateFormat("m/d/yyyy", 1) |
| params.SetTimeFormat(pattern) | |
| params.SetDateTimeFormat(datePattern, timePattern, systemOrder, timeFirst)(index, name, type, length, decimals) | |

| | |
|--|--|
| params.SetPercentFormat(decimals, scaling) | <p>decimals - the number of decimal places; this can be "auto" or any number from 0 to 14. Default value: "auto"</p> <p>scaling - the display factor as a power of 1000; this can be any number from 0 to 5. Default value: 0</p> |
| params.SetFractionFormat(denominator, digits) | <p>1. if the 2nd parameter is "false", (1) is a denominator value 2, 3, ..., n if the 2nd parameter is "true", (1) is a fixed number of denominator digits 1...14 2. ... For example: params.SetFractionFormat(2, true);</p> |
| params.SetScientificFormat(decimals, exponent) | <p>For example: 5 decimal digits and the fixed "07" exponent params.SetScientificFormat("5", "07") variable/automatic number of decimals and the exponent value params.SetScientificFormat("auto", "auto");</p> |
| params.GetFieldDecimals(index) | 0, 1 |
| params.DeleteField(index) | |
| params.fontName | |
| params.fontSize | 0, 1 |
| params.boldFont | |
| params.italicFont | |
| params.underlineFont | |
| params.strikeoutFont | |
| params.fontColor | |
| params.horzAlignment | |
| params.vertAlignment | |
| params.wrapText | |

| | |
|-------------------|--|
| params.shrinkText | |
| params.Reset | |

55.31 Setting record flags

| | |
|---|---|
| GetRecordFlag(record) | |
| SetRecordFlag(recordFrom, recordTo, flag) | flag index - 1 to the number of defined flags |

55.32 Setting database passwords for GS-Base *.gsb/*.zip files

| | |
|--|--|
| SetFilePassword(enable, cryptMethod, oldPassword, newPassword) | enable: 0, 1 cryptMethod: "blowfish" oldPassword: empty if settings a new password newPassword: empty if removing password protection |
|--|--|

Note: The industry-level encryption include the entire database contents and all fields (textual, numeric, LongText, Code, Images/Objects). Information included in JScript and VBscript scripts stored either as GS-Base global settings or scripts added to files using the "File > Database Scripts" command is not encrypted.

55.33 File information and access

| | |
|-------------------------|---|
| GetFileInfo(path, type) | type=1 - returns the file size, type=2 - return the last modification date in the format YYYY-MM-DD |
| ReleaseFile() | Closes, detaches the current database file and return its file path. This enables updating records by another processes/users. Using "Save()" |

| | |
|------------------|--|
| | before attaching the file back causes displaying the "File Save As" dialog box. |
| AttachFile(path) | Attach a given file to the current database loaded in RAM and returns 0 if the file has been modified after the most recent use of the Release() function. |

55.34 Updating all calculation formulas

| | |
|------------------------|--|
| UpdateTable(procCores) | Update calculation formulas in all records. By default, record editing actions cause updating the modified records only. |
| updateMode | Can be: "default", "automatic", "manual". Changing this value from "automatic" to "manual" can be helpful if there are many "block" editing actions (field editing, conversion, clearing ranges) and only one final UpdateTable() will be sufficient instead of multiple time-consuming automatic updates. |

55.35 Input/output methods

| | |
|--|---|
| MessageBox(message, buttons, button, icon) | <p>The "button" parameter must be one of the following strings:</p> <ul style="list-style-type: none"> • ok • ok-cancel • retry-cancel • yes-no • yes-no-cancel • abort-retry-ignore <p>The "button" parameter is an index 1-3 of the default message box button.</p> <p>The "icon" parameter must be one of the following strings:</p> <ul style="list-style-type: none"> • exclamation • warning • information |
|--|---|

| | |
|---|---|
| | <ul style="list-style-type: none"> • question • error <p>Returns a text string representing the name of the clicked/pressed button:</p> <ul style="list-style-type: none"> • abort • cancel • continue • ignore • no • ok • retry • tryagain • yes |
| InputDialog(title, isPassword, initValue) | |
| Sleep(unsigned long time) | |

55.36 Window methods

MaximizeAppWindow()

MinimizeAppWindow()

RestoreAppWindow()

55.37 Obtaining the last error details

lastError

56 GS-Base - methods and properties

```

[
    dual,
    helpstring("IXBaseParams Interface"),
    pointer_default(unique)
]
interface IXBaseParams : IDispatch
{
    [propget, id(1), helpstring("property format")] HRESULT
format([out, retval] BSTR *pFormat);
    [propput, id(1), helpstring("property format")] HRESULT
format([in] BSTR format);

    [propget, id(2), helpstring("property encoding")] HRESULT
encoding([out, retval] BSTR *pEncoding);
    [propput, id(2), helpstring("property encoding")] HRESULT
encoding([in] BSTR encoding);

    [id(3), helpstring("method GetFieldCount")] HRESULT
GetFieldCount([out, retval] int *pCounter);

    [id(4), helpstring("method SetField")] HRESULT SetField([in]
int index, [in] BSTR name, [in] BSTR type, [in] int length, [in] int
decimals);
    [id(5), helpstring("method AddField")] HRESULT AddField([in]
BSTR name, [in] BSTR type, [in] int length, [in] int decimals);
    [id(6), helpstring("method InsertField")] HRESULT
InsertField([in] int index, [in] BSTR name, [in] BSTR type, [in] int length,
[in] int decimals);

    [id(7), helpstring("method GetFieldName")] HRESULT
GetFieldName([in] int index, [out, retval] BSTR *pName);
    [id(8), helpstring("method GetFieldType")] HRESULT
GetFieldType([in] int index, [out, retval] BSTR *pType);
    [id(9), helpstring("method GetFieldLength")] HRESULT
GetFieldLength([in] int index, [out, retval] int *pLength);
    [id(10), helpstring("method GetFieldDecimals")] HRESULT
GetFieldDecimals([in] int index, [out, retval] int *pDecimals);

    [id(11), helpstring("method DeleteField")] HRESULT
DeleteField([in] int index);
};

[
    dual,
    helpstring("ITextParams Interface"),
    pointer_default(unique)
]
interface ITextParams : IDispatch
{
    [propget, id(1), helpstring("property separator")] HRESULT
separator([out, retval] BSTR *pSeparator);
    [propput, id(1), helpstring("property separator")] HRESULT
separator([in] BSTR separator);

    [propget, id(2), helpstring("property useSeparator")] HRESULT
useSeparator([out, retval] BOOL* pUseSeparator);

```

```

        [propput, id(2), helpstring("property useSeparator")] HRESULT
useSeparator([in] BOOL useSeparator);

        [propget, id(3), helpstring("property quotingSymbol")] HRESULT
quotingSymbol([out, retval] BSTR *pSymbol);
        [propput, id(3), helpstring("property quotingSymbol")] HRESULT
quotingSymbol([in] BSTR symbol);

        [propget, id(4), helpstring("property useQuoting")] HRESULT
useQuoting([out, retval] BOOL* pUseSymbol);
        [propput, id(4), helpstring("property useQuoting")] HRESULT
useQuoting([in] BOOL useSymbol);

        [propget, id(5), helpstring("property encoding")] HRESULT
encoding([out, retval] BSTR* pEncoding);
        [propput, id(5), helpstring("property encoding")] HRESULT
encoding([in] BSTR encoding);

        [propget, id(6), helpstring("property fieldNames")] HRESULT
fieldNames([out, retval] BOOL* pValue);
        [propput, id(6), helpstring("property fieldNames")] HRESULT
fieldNames([in] BOOL value);

        [propget, id(7), helpstring("property parseNumbers")] HRESULT
parseNumbers([out, retval] BOOL *pValue);
        [propput, id(7), helpstring("property parseNumbers")] HRESULT
parseNumbers([in] BOOL value);

        [propget, id(8), helpstring("property parseDates")] HRESULT
parseDates([out, retval] BOOL *pValue);
        [propput, id(8), helpstring("property parseDates")] HRESULT
parseDates([in] BOOL value);

        [propget, id(9), helpstring("property fixedFieldWidths")]
HRESULT fixedFieldWidths([out, retval] BSTR *pColumnWidths);
        [propput, id(9), helpstring("property fixedFieldWidths")]
HRESULT fixedFieldWidths([in] BSTR columnWidths);

        [propget, id(10), helpstring("property saveObjectsAsZip")]
HRESULT saveObjectsAsZip([out, retval] BOOL *pValue);
        [propput, id(10), helpstring("property saveObjectsAsZip")]
HRESULT saveObjectsAsZip([in] BOOL value);

        [propget, id(11), helpstring("property saveLongTextAsZip")]
HRESULT saveLongTextAsZip([out, retval] BOOL *pValue);
        [propput, id(11), helpstring("property saveLongTextAsZip")]
HRESULT saveLongTextAsZip([in] BOOL value);

        [propget, id(12), helpstring("property autoFitColumns")]
HRESULT autoFitColumns([out, retval] BOOL* pValue);
        [propput, id(12), helpstring("property autoFitColumns")]
HRESULT autoFitColumns([in] BOOL value);
    };

    [
        dual,
        helpstring("IFormatParams Interface"),

```

```

        pointer_default(unique)
    ]
    interface IFormatParams : IDispatch
    {
        [id(1), helpstring("method SetGeneralNumberFormat")] HRESULT
        SetGeneralNumberFormat([in] BSTR decimals, [in] BYTE zeroes, [in] BOOL
        brackets, [in] BOOL inRed, [in] BOOL separators);
        [id(2), helpstring("method SetCurrencyFormat")] HRESULT
        SetCurrencyFormat([in] BSTR decimals, [in] BSTR position, [in] BSTR symbol,
        [in] BOOL brackets, [in] BOOL inRed);
        [id(3), helpstring("method SetAccountingFormat")] HRESULT
        SetAccountingFormat([in] BSTR decimals, [in] BSTR symbol);
        [id(4), helpstring("method SetDateFormat")] HRESULT
        SetDateFormat([in] BSTR pattern, [in] BOOL systemOrder);
        [id(5), helpstring("method SetTimeFormat")] HRESULT
        SetTimeFormat([in] BSTR pattern);
        [id(6), helpstring("method SetDateTimeFormat")] HRESULT
        SetDateTimeFormat([in] BSTR datePattern, [in] BSTR timePattern, [in] BOOL
        systemOrder, [in] BOOL timeFirst);
        [id(7), helpstring("method SetPercentFormat")] HRESULT
        SetPercentFormat([in] BSTR decimals, [in] int scaling);
        [id(8), helpstring("method SetFractionFormat")] HRESULT
        SetFractionFormat([in] long denominator, [in] BOOL digits);
        [id(9), helpstring("method SetScientificFormat")] HRESULT
        SetScientificFormat([in] BSTR decimals, [in] BSTR exponent);

        [propget, id(10), helpstring("property fontName")] HRESULT
        fontName([out, retval] BSTR *pVal);
        [propput, id(10), helpstring("property fontName")] HRESULT
        fontName([in] BSTR newVal);
        [propget, id(11), helpstring("property fontSize")] HRESULT
        fontSize([out, retval] int *pVal);
        [propput, id(11), helpstring("property fontSize")] HRESULT
        fontSize([in] int newVal);

        [propget, id(12), helpstring("property language")] HRESULT
        language([out, retval] BSTR *pLanguage);
        [propput, id(12), helpstring("property language")] HRESULT
        language([in] BSTR language);

        [propget, id(13), helpstring("property boldFont")] HRESULT
        boldFont([out, retval] BOOL *pVal);
        [propput, id(13), helpstring("property boldFont")] HRESULT
        boldFont([in] BOOL newVal);
        [propget, id(14), helpstring("property italicFont")] HRESULT
        italicFont([out, retval] BOOL *pVal);
        [propput, id(14), helpstring("property italicFont")] HRESULT
        italicFont([in] BOOL newVal);
        [propget, id(15), helpstring("property underlineFont")]
        HRESULT underlineFont([out, retval] BOOL *pVal);
        [propput, id(15), helpstring("property underlineFont")]
        HRESULT underlineFont([in] BOOL newVal);
        [propget, id(16), helpstring("property strikeoutFont")]
        HRESULT strikeoutFont([out, retval] BOOL *pVal);
        [propput, id(16), helpstring("property strikeoutFont")]
        HRESULT strikeoutFont([in] BOOL newVal);
    }
}

```

```

        [propget, id(17), helpstring("property fontColor")] HRESULT
fontColor([out, retval] BSTR *pColor);
        [propput, id(17), helpstring("property fontColor")] HRESULT
fontColor([in] BSTR color);

        [propget, id(18), helpstring("property horzAlignment")]
HRESULT horzAlignment([out, retval] BSTR *pHorzAlign);
        [propput, id(18), helpstring("property horzAlignment")]
HRESULT horzAlignment([in] BSTR horzAlign);
        [propget, id(19), helpstring("property vertAlignment")]
HRESULT vertAlignment([out, retval] BSTR *pVertAlign);
        [propput, id(19), helpstring("property vertAlignment")]
HRESULT vertAlignment([in] BSTR vertAlign);

        [propget, id(20), helpstring("property wrapText")] HRESULT
wrapText([out, retval] BOOL *pValue);
        [propput, id(20), helpstring("property wrapText")] HRESULT
wrapText([in] BOOL value);

        [propget, id(21), helpstring("property shrinkText")] HRESULT
shrinkText([out, retval] BOOL *pValue);
        [propput, id(21), helpstring("property shrinkText")] HRESULT
shrinkText([in] BOOL value);

        [id(22), helpstring("method Reset")] HRESULT Reset();
};

[
    dual,
    helpstring("IFieldParams Interface"),
    pointer_default(unique)
]
interface IFieldParams : IDispatch
{
        [propput, id(1), helpstring("property field name")] HRESULT
name([in] BSTR pVal);
        [propget, id(1), helpstring("property field name")] HRESULT
name([out, retval] BSTR* pVal);

        [propput, id(2), helpstring("property field type")] HRESULT
type([in] BSTR newVal);
        [propget, id(2), helpstring("property field type")] HRESULT
type([out, retval] BSTR *newVal);

        [propput, id(3), helpstring("property field subtype")] HRESULT
subtype([in] BSTR stype);
        [propget, id(3), helpstring("property field subtype")] HRESULT
subtype([out, retval] BSTR* pStype);

        [propput, id(4), helpstring("property hyperlink")] HRESULT
hlink([in] BOOL pVal);
        [propget, id(4), helpstring("property hyperlink")] HRESULT
hlink([out, retval] BOOL* pVal);

        [propput, id(5), helpstring("property sortIndex")] HRESULT
sortIndex([in] unsigned short newVal);

```

```

        [propget, id(5), helpstring("property sortIndex")] HRESULT
sortIndex([out, retval] unsigned short* newVal);

        [propput, id(6), helpstring("property sortOrder")] HRESULT
sortOrder([in] BSTR order);
        [propget, id(6), helpstring("property sortOrder")] HRESULT
sortOrder([out, retval] BSTR* pOrder);

        [propput, id(7), helpstring("property formula")] HRESULT
formula([in] BSTR newVal);
        [propget, id(7), helpstring("property formula")] HRESULT
formula([out, retval] BSTR* newVal);

        [propput, id(8), helpstring("property formulaType")] HRESULT
formulaType([in] unsigned short type);
        [propget, id(8), helpstring("property formulaType")] HRESULT
formulaType([out, retval] unsigned short* pType);

        [propput, id(9), helpstring("property filter")] HRESULT
filter([in] BSTR newVal);
        [propget, id(9), helpstring("property filter")] HRESULT
filter([out, retval] BSTR* newVal);

        [propput, id(10), helpstring("property filterType")] HRESULT
filterType([in] int newVal);
        [propget, id(10), helpstring("property filterType")] HRESULT
filterType([out, retval] int* newVal);

        [propput, id(11), helpstring("property matchWords")] HRESULT
matchWords([in] BOOL pVal);
        [propget, id(11), helpstring("property matchWords")] HRESULT
matchWords([out, retval] BOOL* pVal);

        [propput, id(12), helpstring("property matchCase")] HRESULT
matchCase([in] BOOL pVal);
        [propget, id(12), helpstring("property matchCase")] HRESULT
matchCase([out, retval] BOOL* pVal);

        [id(13), helpstring("method Reset")] HRESULT Reset();
};

[
    dual,
    helpstring("IMergeParams Interface"),
    pointer_default(unique)
]
interface IMergeParams : IDispatch
{
        [propput, id(1), helpstring("property path pattern")] HRESULT
path([in] BSTR val);
        [propget, id(1), helpstring("property path pattern")] HRESULT
path([out, retval] BSTR* val);

        [propput, id(2), helpstring("property table name")] HRESULT
table([in] BSTR val);
        [propget, id(2), helpstring("property table name")] HRESULT
table([out, retval] BSTR* val);

```

```

        [propput, id(3), helpstring("property master field index")]
HRESULT masterField([in] int field);
        [propget, id(3), helpstring("property master field index")]
HRESULT masterField([out, retval] int* field);

        [propput, id(4), helpstring("property slave field index")]
HRESULT slaveField([in] int field);
        [propget, id(4), helpstring("property slave field index")]
HRESULT slaveField([out, retval] int* field);

        [propput, id(5), helpstring("property merge type")] HRESULT
mergeType([in] int val);
        [propget, id(5), helpstring("property merge type")] HRESULT
mergeType([out, retval] int* val);

        [propput, id(6), helpstring("property matchFieldNames")]
HRESULT matchFieldNames([in] int val);
        [propget, id(6), helpstring("property matchFieldNames")]
HRESULT matchFieldNames([out, retval] int* val);

        [propput, id(7), helpstring("property ignore empty fields")]
HRESULT ignoreEmpty([in] BOOL val);
        [propget, id(7), helpstring("property ignore empty fields")]
HRESULT ignoreEmpty([out, retval] BOOL* val);

        [propput, id(8), helpstring("property allow undo")] HRESULT
enableUndo([in] BOOL val);
        [propget, id(8), helpstring("property allow undo")] HRESULT
enableUndo([out, retval] BOOL* val);
    };

    [
        dual,
        helpstring("IApplication Interface"),
        pointer_default(unique)
    ]
    interface IApplication : IDispatch
    {
        [id(1), helpstring("method CreateTextParams")] HRESULT
CreateTextParams([out, retval] ITextParams** ppTextParams);
        [id(2), helpstring("method CreateXBaseParams")] HRESULT
CreateXBaseParams([out, retval] IXBaseParams** ppTextParams);

        [id(3), helpstring("method NewDatabase")] HRESULT
NewDatabase([in] BSTR table);
        [id(4), helpstring("method OpenDatabase")] HRESULT
OpenDatabase([in] BSTR path, [in] BSTR password);
        [id(5), helpstring("method OpenTextFile")] HRESULT
OpenTextFile([in] BSTR path, [in] BOOL showDialogBox, [in] ITextParams*
pTextParams);
        [id(6), helpstring("method OpenExcelFile")] HRESULT
OpenExcelFile([in] BSTR path, [in] BOOL showDialogBox, [in] int merge, [in]
int fnames);
        [id(7), helpstring("method OpenHtmlFile")] HRESULT
OpenHtmlFile([in] BSTR path, [in] int fnames);
    };

```

```

[id(8), helpstring("method OpenXBaseFile")] HRESULT
OpenXBaseFile([in] BSTR path, [in] BOOL showDialogBox, [in] IXBaseParams*
pXBaseParams);

[id(9), helpstring("method OpenMySQLFile")] HRESULT
OpenMySQLFile([in] BSTR path);

[id(10), helpstring("method Reload")] HRESULT Reload();

[id(11), helpstring("method SaveRecordSetAs")] HRESULT
SaveRecordSetAs([in] BSTR path, [in] BSTR password);
[id(12), helpstring("method SaveRecordSetAsText")] HRESULT
SaveRecordSetAsText([in] BSTR path, [in] ITextParams* params);
[id(13), helpstring("method SaveRecordSetAsExcel")] HRESULT
SaveRecordSetAsExcel([in] BSTR path, [in] int merge, [in] int fnames, [in]
int saveZip);
[id(14), helpstring("method SaveRecordSetAsXBase")] HRESULT
SaveRecordSetAsXBase([in] BSTR path, [in] BYTE format, [in] IXBaseParams*
params);
[id(15), helpstring("method SaveRecordSetAsMySQL")] HRESULT
SaveRecordSetAsMySQL([in] BSTR path);
[id(16), helpstring("method SaveRecordSetAsPDF")] HRESULT
SaveRecordSetAsPDF([in] BSTR path);

[id(17), helpstring("method Save")] HRESULT Save();
[id(18), helpstring("method SaveTextFile")] HRESULT
SaveTextFile([in] ITextParams* params);
[id(19), helpstring("method SaveExcelFile")] HRESULT
SaveExcelFile([in] int merge, [in] int fnames, [in] int saveZip);
[id(20), helpstring("method SaveXBaseFile")] HRESULT
SaveXBaseFile([in] IXBaseParams* params);
[id(21), helpstring("method SaveMySLQFile")] HRESULT
SaveMySLQFile();

[id(22), helpstring("method SaveDatabaseAs")] HRESULT
SaveDatabaseAs([in] BSTR path);
[id(23), helpstring("method SaveDatabaseAsExcel")] HRESULT
SaveDatabaseAsExcel([in] BSTR path, [in] int merge, [in] int fnames, [in] int
saveZip);
[id(24), helpstring("method SaveDatabaseAsMySQL")] HRESULT
SaveDatabaseAsMySQL([in] BSTR path);

[id(25), helpstring("method Close")] HRESULT Close();

[id(26), helpstring("method ImportTable")] HRESULT
ImportTable([in] BSTR filePath, [in] BSTR tablePath, [in] BSTR password, [in]
BSTR folder, [out, retval] BSTR* uniqueName);
[id(27), helpstring("method ImportTextTable")] HRESULT
ImportTextTable([in] BSTR textFilePath, [in] BSTR tableName, [in]
ITextParams* pTextParams, [in] BSTR folder);
[id(100), helpstring("method ImportExcelTable")] HRESULT
ImportExcelTable([in] BSTR textFilePath, [in] BSTR tableName, [in] int
fnames, [in] BSTR folder);

[id(91), helpstring("method CreateMergeParams")] HRESULT
CreateMergeParams([out, retval] IMergeParams** ppMergeParams);

```

```

        [id(28), helpstring("method MergeTable")] HRESULT
MergeTable([in] int masterField, [in] BSTR tablePath, [in] int slaveField,
[in] BOOL allowDuplicates);
        [id(29), helpstring("method MergeRecords")] HRESULT
MergeRecords([in] IMergeParams* mergeParams);
        [id(92), helpstring("method MergeRecords")] HRESULT
MergeRecordsFromTextFile([in] IMergeParams* mergeParams, [in] ITextParams*
params);
        [id(93), helpstring("method MergeRecords")] HRESULT
MergeRecordsFromExcelFile([in] IMergeParams* mergeParams, [in] int fnames);
        [id(94), helpstring("method MergeRecords")] HRESULT
MergeRecordsFromXBaseFile([in] IMergeParams* mergeParams, [in] IXBaseParams*
params);
        [id(95), helpstring("method MergeRecords")] HRESULT
MergeRecordsFromMySQLFile([in] IMergeParams* mergeParams);

        [id(30), helpstring("method SetActiveTable")] HRESULT
SetActiveTable([in] BSTR path);
        [id(31), helpstring("method GetActiveTable")] HRESULT
GetActiveTable([out, retval] BSTR* path);

        [id(32), helpstring("method GetRecordCount")] HRESULT
GetRecordTotalCount([out, retval] __int64* counter);
        [id(33), helpstring("method GetRecordSetCount")] HRESULT
GetRecordSetCount([out, retval] __int64* counter);
        [id(34), helpstring("method GetRecordCount")] HRESULT
GetFieldCount([out, retval] unsigned short int* counter);

        [id(35), helpstring("method FindDuplicates")] HRESULT
FindDuplicates([in] unsigned short int fieldFrom, [in] unsigned short int
fieldTo, [in] BOOL searchAll, [in] BYTE type, [in] BOOL sortResults);
        [id(36), helpstring("method FindUnique")] HRESULT
FindUnique([in] unsigned short int fieldFrom, [in] unsigned short int
fieldTo, [in] BOOL searchAll, [in] BYTE type, [in] BOOL sortResults, [in]
BOOL useCounters, [in] unsigned short int counterIndex);
        [id(37), helpstring("method FindQuartile")] HRESULT
FindQuartile([in] unsigned short int field, [in] BOOL searchAll, [in] BOOL
statAll, [in] BYTE type);
        [id(38), helpstring("method FindRandom")] HRESULT
FindRandom([in] __int64 counter);
        [id(39), helpstring("method FindFlagged")] HRESULT
FindFlagged([in] unsigned short int flag);
        [id(40), helpstring("method FindComplement")] HRESULT
FindComplement();
        [id(41), helpstring("method FindAll")] HRESULT FindAll();

        [id(42), helpstring("method CreateFieldParams")] HRESULT
CreateFieldParams([out, retval] IFieldParams** fieldParams);
        [id(43), helpstring("method GetField")] HRESULT GetField([in]
unsigned short field, [in/*out, retval*/] IFieldParams* fieldParams);
        [id(44), helpstring("method SetField")] HRESULT SetField([in]
unsigned short field, [in] IFieldParams* fieldParams);
        [id(45), helpstring("method ConvertField")] HRESULT
ConvertField([in] unsigned short field, [in] BYTE type);
        [id(46), helpstring("method ResetSorting")] HRESULT
ResetSorting();

```

```

[id(47), helpstring("method AppendField")] HRESULT
AppendField([in] IFieldParams* fieldParams);
[id(48), helpstring("method InsertField")] HRESULT
InsertField([in] unsigned short field, [in] IFieldParams* fieldParams);
[id(49), helpstring("method RemoveField")] HRESULT
RemoveField([in] unsigned short field);

[id(50), helpstring("method GetDatabaseItemCount")] HRESULT
GetDatabaseItemCount([out, retval] __int64* pCounter);
[id(51), helpstring("method GetDatabaseItem")] HRESULT
GetDatabaseItem([in] __int64 index, [out, retval] BSTR* path);
[id(52), helpstring("method GetFirstDatabaseItem")] HRESULT
GetFirstDatabaseItem([in] BSTR folder, [out, retval] BSTR* name);
[id(53), helpstring("method GetNextDatabaseItem")] HRESULT
GetNextDatabaseItem([in] BSTR prevName, [out, retval] BSTR* nextName);
[id(54), helpstring("method RenameDatabaseItem")] HRESULT
RenameDatabaseItem([in] BSTR path, [in] BSTR name, [out, retval] BSTR*
uniqueName);
[id(55), helpstring("method InsertDatabaseItem")] HRESULT
InsertDatabaseItem([in] BSTR folder, [in] BOOL bottom, [in] BSTR name, [out,
retval] BSTR* uname);
[id(56), helpstring("method DeleteDatabaseItem")] HRESULT
DeleteDatabaseItem([in] BSTR path);

[id(57), helpstring("method RemoveRecords")] HRESULT
RemoveRecords([in] __int64 recordFrom, [in] __int64 recordTo);
[id(58), helpstring("method InsertRecords")] HRESULT
InsertRecords([in] __int64 recordFrom, [in] __int64 recordTo);
[id(59), helpstring("method InsertText")] HRESULT
InsertText([in] __int64 record, [in] unsigned short field, [in] BSTR text);
[id(60), helpstring("method GetText")] HRESULT GetText([in]
__int64 record, [in] unsigned short field, [out, retval] BSTR* text);
[id(61), helpstring("method InsertNumber")] HRESULT
InsertNumber([in] __int64 record, [in] unsigned short field, [in] double
number);
[id(62), helpstring("method GetNumber")] HRESULT
GetNumber([in] __int64 record, [in] unsigned short field, [out, retval]
double* number);
[id(63), helpstring("method ClearField")] HRESULT Clear([in]
__int64 record, [in] unsigned short field);
[id(64), helpstring("method ClearRange")] HRESULT
ClearRange([in] __int64 record1, [in] unsigned short field1, [in] __int64
record2, [in] unsigned short field2);

[id(96), helpstring("method InsertSeries")] HRESULT
InsertSeries([in] unsigned short field, [in] __int64 record1, [in] __int64
record2, [in] BOOL copy);
[id(97), helpstring("method InsertRecurrenceSeries")] HRESULT
InsertRecurrenceSeries([in] unsigned short field, [in] __int64 record1, [in]
__int64 record2, BSTR formula);
[id(98), helpstring("method InsertCustomSeries")] HRESULT
InsertCustomSeries([in] unsigned short field, [in] __int64 record1, [in]
__int64 record2, [in] __int64 series);
[id(99), helpstring("method InsertRandomSeries")] HRESULT
InsertRandomSeries([in] unsigned short field, [in] __int64 record1, [in]

```

```

__int64 record2, [in] unsigned short distribution, [in] double param1, [in]
double param2);

[id(65), helpstring("method GetColumnWidth")] HRESULT
GetColumnWidth([in] unsigned short field, [out, retval] unsigned short*
pWidth);

[id(66), helpstring("method SetColumnWidth")] HRESULT
SetColumnWidth([in] unsigned short field, [in] unsigned short width);

[id(67), helpstring("method FitColumnWidth")] HRESULT
FitColumnWidth([in] unsigned short fieldFrom, [in] unsigned short fieldTo);

[id(68), helpstring("method GetRowHeight")] HRESULT
GetRowHeight([in] __int64 record, [out, retval] unsigned short* pHeight);
[id(69), helpstring("method SetRowHeight")] HRESULT
SetRowHeight([in] __int64 record, [in] unsigned short height);

[id(70), helpstring("method GetAutoRowHeight")] HRESULT
GetAutoRowHeight([in] __int64 record, [out, retval] BOOL* pVal);
[id(71), helpstring("method SetAutoRowHeight")] HRESULT
SetAutoRowHeight([in] __int64 recordFrom, [in] __int64 recordTo, [in] BOOL
val);

[id(72), helpstring("method CreateFormatParams")] HRESULT
CreateFormatParams([out, retval] IFormatParams** ppFormatParams);
[id(73), helpstring("method GetFieldFormat")] HRESULT
GetFieldFormat([in] unsigned short field, [in] IFormatParams* pFormatParams);
[id(74), helpstring("method SetFieldFormat")] HRESULT
SetFieldFormat([in] unsigned short field, [in] IFormatParams* pFormatParams);

[id(75), helpstring("method GetRecordFlag")] HRESULT
GetRecordFlag([in] __int64 record, [out, retval] int* flag);
[id(76), helpstring("method SetRecordFlag")] HRESULT
SetRecordFlag([in] __int64 recordFrom, [in] __int64 recordTo, [in] int flag);

[id(77), helpstring("method SetFilePassword")] HRESULT
SetFilePassword([in] BOOL enable, [in] BSTR cryptMethod, [in] BSTR
oldPassword, [in] BSTR newPassword);

[id(78), helpstring("method GetFileInfo")] HRESULT
GetFileInfo([in] BSTR path, [in] BYTE type, [out, retval] BSTR* value); // 1
- size, 2 - mod. date
[id(79), helpstring("method ReleaseFile")] HRESULT
ReleaseFile([out, retval] BSTR* value);
[id(80), helpstring("method AttachFile")] HRESULT
AttachFile([in] BSTR path, [out, retval] int* modified);

[id(81), helpstring("method UpdateTable")] HRESULT
UpdateTable([in] BYTE procCores);
[propget, id(82), helpstring("property updateMode")] HRESULT
updateMode([out, retval] BSTR* pMode);
[propput, id(82), helpstring("property updateMode")] HRESULT
updateMode([in] BSTR mode);

[id(83), helpstring("method MessageBox")] HRESULT
MessageBox([in] BSTR message, [in] BSTR buttons, [in] int button, [in] BSTR
icon, [out, retval] BSTR* retValue);

```

```

        [id(84), helpstring("method InputBox")] HRESULT InputBox([in]
BSTR title, [in] BOOL password, [in] BSTR initValue, [out, retval] BSTR*
retValue);
        [id(85), helpstring("method Sleep")] HRESULT Sleep([in]
unsigned long time);

        [id(86), helpstring("method MaximizeAppWindow")] HRESULT
MaximizeAppWindow();
        [id(87), helpstring("method MinimizeAppWindow")] HRESULT
MinimizeAppWindow();
        [id(88), helpstring("method RestoreAppWindow")] HRESULT
RestoreAppWindow();

        [propget, id(89), helpstring("property lastError")] HRESULT
lastError([out, retval] BYTE* pCode);
        [propput, id(89), helpstring("property lastError")] HRESULT
lastError([in] BYTE code);

        [propget, id(90), helpstring("property zip64")] HRESULT
zip64([out, retval] BYTE* pCode);
        [propput, id(90), helpstring("property zip64")] HRESULT
zip64([in] BYTE code);

```

57 Support

If you have any questions, comments or suggestions, please visit the support forum or e-mail Citadel5 directly.