The information contained in this guide is not of a contractual nature and may be subject to change without prior notice.

The software described in this guide is sold under a license agreement. The software may be used, copied or reproduced only in accordance with the terms of the agreement.

No part of this guide may be copied, reproduced or transmitted in any form, by any means or for any purpose other than the purchaser’s own use without the written permission of Teklynx Newco SAS.

©2015 Teklynx Newco SAS,

All rights reserved.
# Table of contents

About this manual ................................................................. i

Typographical conventions ................................................... i

About your product ............................................................... i

Connecting to database ......................................................... 1

Overview .............................................................................. 1

Active Query Builder ............................................................ 4

Getting started ....................................................................... 5

Adding an object to the query ................................................. 5

Editing object properties ....................................................... 6

Joining tables ......................................................................... 7

Sorting output fields ............................................................. 8

Defining criteria ...................................................................... 8

Defining a parameterized query .............................................. 9

The Query result grid ............................................................ 9

Database Manager ............................................................... 11

Database Structure Window ............................................... 11

Choose a database from the list of connections ......................... 11

Delete a table in the active database .................................... 12

Edit Database window ........................................................ 13

Database Query window ....................................................... 15

Remove a filter ...................................................................... 17

Modify a filter in SQL .......................................................... 17

The Print window ................................................................... 17
About this manual

Typographical conventions

This manual distinguishes between different types of information using the following conventions:

- Terms taken from the interface itself, such as commands, appear in **bold**.
- Key names appear in all caps. For example: "Press the SHIFT key."
- Numbered lists indicate that there is a procedure to follow.
- When the conjunction -or- appears next to a paragraph, it means there is the choice of another procedure for carrying out a given task.
- When a menu command contains submenus, the menu name followed by the command to select appears in bold. Thus, "Go to **File > Open**" means choose the **File** menu, then the **Open** command.

About your product

Some of the functions described in this manual may not be available in your product.

For the complete list of specific features available in your software, refer to the specification sheet provided with the product.
Connecting to database

Overview

In this chapter we are going to link a label (the container) with a database (the content). To do this, we will use ODBC (Open DataBase Connectivity) or OLE DB connections.

Databases allow you to store data. All data is organized into two-dimensional tables in what is called a relationship. Each row in a table is called a record. The purpose of a record is to manage an object, the properties of which are organized across the different columns of the table in the form of fields.

A database can contain a number of tables. To link the different tables within a given database, we use joins. A concrete example later in this chapter demonstrates how joins are created.

ODBC

ODBC data sources make it possible to access data belonging to a wide range of database management systems. ODBC makes it easy to link an application such as your label design software with a certain number of databases. The software comes with several ODBC drivers. These enable you to access the most common types of databases.

Some of the more common drivers are listed below:

- Microsoft Access Driver (*.mdb)
- Microsoft Excel Driver (*.xls)
- Microsoft FoxPro Driver (*.dbf), etc.

OLE DB

OLE DB is a set of interfaces that gives access to all data, regardless of type, format or location. It provides components such as access interfaces, query drivers, and so on. These components are called "providers".

The example below describes a connection process when a database is not connected to your software.

Installing an ODBC data source

The process described below uses the direct creation mode. If you want, you can use the wizard by selecting Wizard in the context menu.

Connecting to the TKTraining.mdb database

1. In CODESOFT, choose Tools > Administrator ODBC.
2. Click on the System Data Source tab (DSN), and then click Add.

Note: You can define data sources with system Data Sources Names (DSNs). These data sources are unique to a specific computer but not to a specific user. Any user with the necessary rights can access a system DSN.
3. Select Microsoft Access Driver then click Finish.
4. Enter "TK Training Level 2" in the Data source name field.
5. Click Select and select the database TKTraining.mdb database, which can be found in the InstallDir\Samples\Forms\Tutorial directory.
6. Click the Options button. Check the Read Only option. This option makes it possible to open the database at the same time as your labelling software without causing any read/write problems.
7. Click OK in the ODBC Microsoft Excel Setup dialog box.

Importing data

After the database is connected to your software, you have to connect it to your document.

1. Open the PRODUCT_WS3 label.
2. Choose Data source > Database > Create/Modify query.
3. Select TK Training Level 2 in the Select data source list.
4. Select "Fruits" in the Select table list.
   The database fields appear in the Select fields list.
5. Select the "ProdName", "Origin", "Weight", and "Reference" fields.
6. Click the button. It allows the selected records to be sorted into alphabetical or numeric order, ascending, or descending.
7. Select "Reference" as the Sort Key and "Ascending" as the Sort Order.
8. Save the query in the InstallDir\Samples\Forms\Tutorial\PRODUCT_WS4_ODBC.CSQ directory.
9. Click OK.
   The variables are created automatically and are listed in the Database branch in the Data sources view.

To view or print the different values that your object can take, use the navigation bar. You can also print from the Query result window.

Creating variable objects

1. Select the created variables listed in the Database list in the Data sources view, then drag and drop it into the workspace.
2. Select Text in the context menu.

The example below describes a connection process when a database is not connected to your software.

Installing an ODBC data source

The process described below uses the direct creation mode. If you want, you can use the Wizard by selecting Wizard in the context menu.
Connecting to the TKTraining.mdb database

1. In Labelview, choose Tools > Administrator ODBC.
2. Click on the System Data Source tab (DSN) then click Add.

**Note**: You can define data sources with system Data Sources Names (DSNs). These data sources are unique to a specific computer but not to a specific user. Any user with the necessary rights can access a system DSN.

3. Select Microsoft Access Driver then click Finish.
4. Enter "TK Training Level 2" in the Data source name box.
5. Click Select and select the database TKTraining.mdb, which can be found in the InstallDir\Samples\Forms\Tutorial.
6. Click the Options button. Check the Read Only option. This option makes it possible to open the database at the same time as your labelling software without causing any read/write problems.
7. Click OK in the ODBC Microsoft Excel Setup dialog box.

**Importing data**

When the database is connected to your software, you have to connect it to your document.

1. Open the label PRODUCT_WS3.
2. Choose Data source > Database > Create/Modify query.
3. Select TK Training Level 2 in the Select data source list.
4. Select "Fruits" in the Select table list.
   The database fields appear in the Select fields list.
5. Select the "ProdName", "Origin", "Weight", and "Reference" fields.
6. Click the button. It allows the selected records to be sorted into alphabetical or numeric order, Ascending, or Descending.
7. Select "Reference" as the Sort Key and "Ascending" as the Sort Order.
8. Save the query in the InstallDir\Samples\Forms\Tutorial\PRODUCT_WS4_ODBC.CSQ.
9. Click OK.
   The variables are created automatically and are listed in the Database branch in the Data sources view.

To view or print the different values that your object can take, use the navigation bar. You can also print from the Query result window.

**Creating variable objects**

1. Select the created variables listed in the Database branch in the Data sources view, then drag and drop it into the workspace.
2. Select Text in the context menu.
Defining the properties of a Table lookup data source

Command: Data sources > Table lookup > Table lookup name > Properties.

1. Select the data source from the Select data source list.

   **Note:** To create a new data source, click the New data source button. This gives you the option of using the wizard or choosing between an ODBC or OLEDB data source.

2. By default, Standard creation mode is activated. You can use Advanced SQL creation mode to create your query.

**Standard creation mode**

3. In the Select table list, select a table where the search is to be carried out.

4. In the Select result field list, select the field whose value will be transferred into your variable.

5. Click to add a row.

6. Select the field in the external table on which the search will be carried out.

7. Select the current document variable containing the search value.

8. Click the Test button to display the result.

**Advanced SQL creation mode**

3. Click SQL mode.

4. Enter your query in SQL format.

   - or -

   Click SQL Query Builder to access Query Builder. This provides an easy-to-use interface for building SQL database queries. You can create new requests graphically or insert existing requests in your document.

5. Click the Test button to display the result in the Query dialog box.

**Active Query Builder**

Active Query Builder is a visual query builder component that allows you to build complex SQL queries using an intuitive visual query building interface.

To work with Active Query Builder, you must have basic knowledge of SQL concepts. Active Query Builder will help you to write correct SQL code hiding technical details, but you must have an understanding of the SQL principles to achieve desired results.
Getting started

The main Active Query Builder window is divided into the following parts:

- The **Query Building area** is the main area where the visual representation of query will be displayed. This area allows you to define source database objects and derived tables, define links between them and configure properties of tables and links.
- The **Columns pane** is located below the Query Building area. It is used to perform all necessary operations with query output columns and expressions. Here you can define field aliases, sorting and grouping, and define criteria.
- The **Query Tree pane** is located at the right. Here you may browse your query and quickly locate any part of it.
- The page control above the Query Building area will allow you to switch between the main query and sub-queries.
- The small area in the corner of the Query Building area marked by the letter **Q** is the union sub-query handling control. Here you can add new union sub-queries and perform all necessary operations.

![Query Builder Window](image)

Adding an object to the query

To add an object to the query, right-click the Query Building area and select the **Add Object** item in the drop-down menu.
The **Add New Object** window allows you to add multiple objects at once. The objects are grouped according to their type in four tabs: **Tables, Views, Procedures** (or Functions) and **Synonyms**. You may select one or several objects by holding the **CTRL** key and then press the **Add Object** button to add these objects to the query. You may repeat this operation several times. After you finish adding objects, click the **Close** button to hide this window.

To remove an object from the query, select it and press the **DELETE** key or simply click the **Close** button in the object header.

For those servers that have schemas or allow selection of objects from different databases, you can filter objects by database or schema name by selecting the necessary schema or database from the combo box at the top of the window.

Active Query Builder can establish links between tables based on information about foreign keys in the database. This ability is turned on by default. To turn it off, clear the **Create links from foreign keys** checkbox.

**Editing object properties**

You can change the properties of each object added to the query by right-clicking the object and selecting the **Edit** option from the drop-down menu or simply by double-clicking the object header.
The **Datasource Properties** dialog may vary from server to server, but the **Alias** property is the same for all database servers.

**Joining tables**

To create a link between two objects (i.e., join them), you should select the field to link the objects and drag it to the corresponding field of the other object. After drag the field to the other object, a line connecting the linked fields will appear.
The default join type is INNER JOIN. This means that only matching records in both tables will be included in the resulting dataset. To define other types of joins, right-click the link and select the Edit option in the drop-down menu or double-click it to open the Link Properties window. This window allows you to define join type and other link properties.

To remove a link between objects, right-click the link line and select the Remove option in the drop-down menu.

**Sorting output fields**

To sort of output query fields, use the Sort Type and Sort Order columns in the Columns Pane.

The Sort Type column allows you to specify the way the fields will be sorted, either in the Ascending or Descending order.

The Sort Order column allows you to set up the order in which fields will be sorted if more than one field will be used for sorting.

To disable sorting by a field, clear the Sort Type column for the field.

**Defining criteria**

To define criteria for the expression listed in the Columns Pane, use the Criteria column.

In this column, write the criterion omitting the expression itself. For example, to get the following criterion in your query:
WHERE (field >= 10) AND (field <= 20)

you should write

>= 10 AND <= 20

in the Criteria column.

You can specify several criteria for a single expression using the Or columns. These criteria will be concatenated in the query using the OR operator.

Defining a parameterized query

Query Builder lets you create a parameterized query where the value of the parameter is held in a variable.

**Note:** You must first create a variable before you can define parameters...

1. Drag and drop the table on which your query is to be carried out.
2. Select the field(s) to which the criterion or criteria is/are to apply.
3. In the Criterion column or in the SQL-format edit field, specify the variable to be used as the object of a search criterion.
   Example: To look for the value of the variable Var0:

   - **In SQL:**
     
     ```sql
     SELECT [Table].*
     FROM [Table]
     WHERE [Table].Field = APPLICATION.DOCUMENT.Var0
     ```

   - **Criterion column**
     
     = APPLICATION.DOCUMENT.Var0

4. Click to display the result of your query.

The Query result grid

To access the Query result grid, click the button in the Defining a query dialog box, in the Merge database browser toolbar or via the Data source > Database menu.

This grid allows the result of a query to be displayed and allows you to search for a particular term and all its occurrences and print the corresponding labels.

The Query result grid contains:

- Search functions
**Search** field, which allows you to enter the field you want to search.

Data to search for, which allows you to enter the value to search for.

Search for the value anywhere in the field or at the start of the field.

- **Navigation functions for browsing query result records:**
  - First record
  - Previous record
  - Next record
  - Last record

- **The results grid**
  Displays the search results.

- **Requery**
  Requery the request and update the grid.
Database Manager

Database Structure Window

The Database Structure window is used to manage the structure of the database file. From this window, you can add, modify or delete tables/fields, etc.

Choose a database from the list of connections

1. Click the Database drop-down list.
2. Select the required database.
Choose a table in a database

1. Click the Table drop-down list.
2. Select the required table.

Add a table to the active database

1. Click Add table.
2. Enter the name of the new table.
3. Click OK.

You can also copy the structure of the table from a table that already exists in the selected database. To copy a table:

1. Select the Duplicate with check box.
2. Click on the drop-down list.
3. Select the required data.
4. Click OK.

Delete a table in the active database

1. Click the Table drop-down list.
2. Click on the data required.
3. Click Delete table.
View/hide active table’s data

1. Click **View data**.

Define a key field

1. Select the check box next to the required field.
2. Click **Apply**.

Define a field’s content type

1. Click the required field in the **Type** column.
2. Click the drop-down list button.
3. Select the required data.
4. Click **Apply**.

Define a field’s maximum size

1. Click the required field in the **Length** column.
2. Enter the value required.
3. Click **Apply**.

Allow an empty field

1. Select the **Allow Null** check box for the required field.
2. Click **Apply**.

Edit Database window
The Edit Database window is used to manage the contents of the database file. In this window, you can add, modify, or delete data.

These actions depend on the type of database. Excel file records cannot be modified.

Select records according to their content

Use the content of a field to find a record.

1. Click the table drop-down list button.
2. Click the data required.
3. Click the Data Input field.
4. Enter the value required in the Data Input field.

Select all identical records

At least one record must have been found.

1. Click the drop-down list button.
2. Click the data required.
3. Click the **Data Input** field.
4. Enter the required data in the **Data Input** field.
5. Click the **Select all** button ( ).

**Select an identical record**

At least one record must have been found. There must be several identical records in the **Search** field.

To select a record, use the search tool:

**Create a new record**

1. Click a field in the row marked with an asterisk.
2. Enter the values required in the corresponding fields.
3. Click **Apply**.

**Modify a record**

1. Click the data you want to modify.
2. Enter the required data.
3. Click **Apply**.

**Delete a record**

1. Click the database cursor for the required field.
2. Right-click the database cursor for the required field.
3. Click **Delete Record**.

**Database Query window**
The Database Query window is used to create and apply various filters.

**Add a query**

1. Click **Add query** on the **Fields** tab.
2. Enter a name for the query.
3. Click **OK**.

**Select/deselect one or more fields**

1. Use the buttons to select the required (fields).
2. Click **Query** to refresh the database preview.

**Modify the order of fields selected**
Database Manager

1. Click the required field in the **Ordered fields** window.
2. Click the up or down arrow to reorder the fields.
3. Click Query to refresh the database preview.

**Create a filter using predefined data**

1. Select the **Filters** tab.
2. Click the **Add row** button ( ![Add Row](add_row.png)).
3. In the **Field** field, select in the drop-down list the data required.
4. In the **Operator** field, select the desired operator from the drop-down list.
5. In the **Value** field, enter the required value.
6. Click Query to view the result.

**Apply a logical operator to several filters**

1. Click the **Add row** button ( ![Add Row](add_row.png)).
2. In the **Logical** field, select the desired logic (AND or OR) from the drop-down list.
3. Create a filter (as described above).
4. Click Query to view the result.

**Remove a filter**

**Note:** At least one filter must exist.

1. Click the database cursor for the required field.
2. Click the **Delete row** button ( ![Delete Row](delete_row.png)).

**Modify a filter in SQL**

**Note:** At least one filter must exist.

1. Select the **SQL Query** tab.
2. Check **Modify the query in SQL language** to activate the SQL Query and make manual changes.
3. Click Query to view the result.

**The Print window**
The Print window is used to select files for printing, assign printers, and define various parameters before printing is launched.

Select a document to be printed

1. Select a document from a file

   - or -

2. Select the **File** check box in the **Label** name group.

   - or -

1. Click the **Label Creation Wizard** button ( 🍃 ).
2. Follow the on screen instructions.

**Note:** Creating a label in relation to the database allows you to define exactly which elements are required to position each database field.

Select an existing label template

1. Click the **Open an existing document** button ( 🍃 ).
2. Select a .lab .lbl file.
3. Click **OK**.

**Note:** The Field radio buttons in the Label name and Printer name groups of options allow you to choose the required label or printer, when these options are defined in one of the fields of the active database.
Select a document from a field

If your database contains the name of the label to be used for the printing job in one of the fields, you can define this field as the location where Database Manager is going to pick up the .lab .lbl file. For example, consider the following database:

<table>
<thead>
<tr>
<th>Ref</th>
<th>Designation</th>
<th>Qt</th>
<th>Code</th>
<th>Labname</th>
</tr>
</thead>
<tbody>
<tr>
<td>6574</td>
<td>Ref1</td>
<td>1</td>
<td>9876546321</td>
<td>Label1.lab</td>
</tr>
<tr>
<td>6354</td>
<td>Ref2</td>
<td>2</td>
<td>1236478855</td>
<td>Label2.lab</td>
</tr>
<tr>
<td>6987</td>
<td>Ref3</td>
<td>3</td>
<td>6987456321</td>
<td>Label1.lab</td>
</tr>
<tr>
<td>3684</td>
<td>Ref4</td>
<td>4</td>
<td>3698745632</td>
<td>Label3.lab</td>
</tr>
</tbody>
</table>

1. Check **Field** box in the **Label name** group.
2. Select the required field.

Select a printer

1. Click on the **Add** or **Remove a printer** button.
2. Select the required printer.
Formulas

The Formula data source

The Formula data source contains a list of data sources.

These data sources are populated by combinations of operators, constants, data sources, control variables, formulas, and functions. Data can be numeric or alphanumeric.

In order to carry out a calculation within a document, you must first create a Formula data source.

This data source has a specific dialog box allowing you to define the required function(s) for a given formula.

About functions

Functions are predefined formulas that carry out calculations by using values called arguments, placed in a certain order, called syntax.

Functions are used to return a numeric, character string, or logical value, which is the result of a calculation or an operation.

There are six groups of functions in the formula definition:

Operators

Operators are mathematical symbols indicating an operation to be carried out. There are different types of operators: arithmetic, comparative, concatenation, and logical.

Arithmetic operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>Multiply two numbers.</td>
</tr>
<tr>
<td>+</td>
<td>Add two numbers together.</td>
</tr>
<tr>
<td>-</td>
<td>Subtract one number from another, or assign a negative value to an operand.</td>
</tr>
<tr>
<td>/</td>
<td>Divide one number by another.</td>
</tr>
<tr>
<td>^</td>
<td>Raise a number to the power of the exponent.</td>
</tr>
<tr>
<td>%</td>
<td>Modulo.</td>
</tr>
</tbody>
</table>

Comparative operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;</td>
<td>Less than.</td>
</tr>
<tr>
<td>&lt;=</td>
<td>Less than or equal to.</td>
</tr>
<tr>
<td>&gt;</td>
<td>Greater than.</td>
</tr>
<tr>
<td>&gt;=</td>
<td>Greater than or equal to.</td>
</tr>
</tbody>
</table>

Formulas

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
<td>Equal to.</td>
</tr>
<tr>
<td>&lt;&gt;</td>
<td>Different from.</td>
</tr>
</tbody>
</table>

**Concatenation operator**

Used to combine two strings.

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;</td>
<td>Concatenation of two strings.</td>
</tr>
</tbody>
</table>

**Logical operator**

(See also Logical Functions)

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>!</td>
<td>Not logical.</td>
</tr>
</tbody>
</table>

**Mathematical functions**

**Abs(data):** Returns the absolute (positive) value of data.

Examples:

- \( \text{Abs}(-5) = 5 \)
- \( \text{Abs}(5) = 5 \)

**base10tobaseX(«string_1»,«string_2»):** Converts \( \text{string}_2 \) from base 10 to base \( \text{string}_1 \).

Examples

- If the field named Base 16 contains the string "0123456789ABCDEF"
  - \( \text{BASE10TOBASEX(Base16, 12)} \) produces C
  - \( \text{BASE10TOBASEX(Base16,10)} \) produces A
  - \( \text{BASE10TOBASEX("012345","9")} \) produces 13

Note: This formula cannot accept negative decimal numbers for \( \text{string}_2 \) parameter.

**baseXtobase10(«string_1»,«string_2»):** Converts \( \text{string}_2 \) from base \( \text{string}_1 \) to base 10.

Examples

- If the field named Base 16 contains the string "0123456789ABCDEF"
  - \( \text{BASEXTOBASE10(Base16, "E")} \) produces 14
  - \( \text{BASEXTOBASE10(Base16,10)} \) produces A
  - \( \text{BASEXTOBASE10("012345","9")} \) produces 13

**Ceil(data):** Rounds data up to the next whole number.
### Formulas

**Example:**

Ceil(3.234) = 4  
Ceil(7.328) = 8

**Decimals(data1, data2):** Uses data2 decimal places in data1.

**Example:**

Decimals(4, 2) = 4.00  
Decimals(3.524, 1) = 3.5

**eval_add(«string»,«string»):** Returns the sum of parameters.

**Example**

eval_add(5, 5) = 10

**eval_div(«string»,«string»):** Returns the division of parameters.

**Example**

eval_div(20, 2) = 10

**eval_mult(«string»,«string»):** Returns the multiplication of parameters.

**Example**

eval_mult(5, 2) = 10

**eval_sub(«string»,«string»):** Returns the substraction of parameters.

**Example**

eval_sub(20, 10) = 10

**Floor(data):** This function rounds data down to the next whole number.

**Example:**

Floor(3.234) = 3  
Floor(7.328) = 7

**hex(«val_1»,«val_2»):** Converts the val_1 decimal number to hexadecimal format with a total value of val_2.

**Note:** This formula cannot accept negative decimal numbers for val_1 parameter.

**Example:**

hex(2, 8) = 00000002

**int («value»):** Returns the largest integer less than or equal to the value argument.
Examples:
int (-5.863) = -6
int (5.863) = 5

max(data1, data2, ...) Displays the largest value in the data series.

Example:
Max(5, 12.3) = 12.3

min(data1, data2, ...): Displays the lowest value in the data series.

Example:
Min(5, 12.3) = 5

mod («val_1», «val_2»): Returns the remainder of the division of the val_1 argument by the val_2 argument. The result has the same sign as the divisor.

Examples:
mod (7,2) = 1
mod (-7,2) = -1
mod (7,-2) = 1
mod (-7,-2) = -1

quotient («val_1», «val_2»): Returns the integer result of the division of the val_1 argument by the val_2 argument.

Example:
quotient(10, 2) = 5

round («val_1», «val_2»): Returns the argument val_1 rounded to the number of figures indicated by val_2.

- If val_2 is greater than 0, val_1 is rounded to the number of decimals indicated.
- If val_2 is equal to 0, val_1 is rounded to the closest integer.
- If val_2 is less than 0, val_1 is rounded off to the left of the decimal point.

Examples:
round (4.25,1) = 4.3
round (1.449, 1) = 1.4
round (42.6,-1) = 40

trunc («value»): Returns the integer part of the value argument.

Example:
trunc(10.0001) = 10

Logical functions
Logical functions allow you to check whether one or more conditions have been fulfilled.

**Note**: TRUE equals 1 and FALSE equals 0.

**and («expr_1», «expr_2»)**: Returns TRUE if both arguments are true, FALSE if at least one is false. The arguments must be calculated from logical values.

**Example**:
- `and(exact("string","string"),exact("string","string")) = 0`
- `and(exact("string","string"),exact("string","string")) = 1`

**exact («string_1», «string_2»)**: Returns TRUE if the two strings are identical, FALSE if not. This function is case-sensitive.

**Example**:
- `exact("software","software") = 1`
- `exact("software","software") = 0`

**if («expr», «Val_if_true», «Val_if_false»)**: Returns the `Val_if_true` value if `Expr` is true and the `Val_if_false` argument if `Expr` is false.

**Example**:
- `if(exact("string", "string"), "true", "false") = false`
- `if(exact("string", "string"), "true", "false") = true`

**not («logical»)**: Gives the opposite of the `logical` argument.

**Example**:
- `not(exact("string", "string")) = 1`
- `not(exact("string", "string")) = 0`
- `not(False) = 1 or not(0) = 1`
- `not(True) = 0 or not(1) = 0`
- `not(1+1=2) = 0`

**or («expr_1», «expr_2»)**: Returns TRUE if one of the two arguments is true and FALSE if both arguments are false. The arguments must be calculated from logical values.

**Example**:
- `or(exact("string", "string"),exact("string", "string")) = 0`
- `or(exact("string", "string"),exact("string", "string")) = 1`
- `or(true,true) = 1 or or(1,1) = 1`
- `or(true,false) = 1 or or(1,0) = 1`
- `or(false,false) = 0 or or(0,0) = 0`

**Text functions**
A character string can be assimilated into a table if each field contains a character. It is defined by its length (total number of characters in the string, including spaces). The position of a character in the string corresponds to its place in the table. For example, the first character is at position one.

**Example:** Position 3 corresponds to the third character in the string.

**AI253:** Specific function for preparing string for application identifier 253.

**AI8003:** Specific function for preparing string for application identifier 8003.

cyclebasex (): Allows counting to take place in any kind of database counting system. The numbering system must be defined within the linked expression. The start value, the value of each increment and the number of copies must also be specified for each number. All these values can be linked to other fields in the label, but the field names must not be enclosed in quotation marks.

**Example:**
If a field named Base 16 contains the character string 0123456789ABCDEF, then:
cyclebasex(base16, "8", 1,1) = 8,9,A,B,C
cyclebasex(base16, "F", -1,1) = F,E,D,C,B,A 9,8,7
cyclebasex(base16, "B0 ", 1,1) = B0, B1, B2
cyclebasex("012345", "4",1,2) = 4,4,5,5,10,10,11,11

cyclechar (): Creates a user-defined set of characters for a complete cycle.

**Examples:**
cyclechar("A", "C") = A B C A B C A B C
cyclechar("A", "C", 1,2) = A A B B C C A A B B

cyclenumber (): Allows you to set your own sequence of numbers, instead of using the normal sequence of numbers or letters (0,1,2; or A,B,C).

**Examples:**
cyclenumber(1,3) will produce labels in the following sequence: 1 2 3 1 2 3 1 2 3...
cyclenumber(1,3,1,2) will produce labels in the following sequence: 1,1,2,2,3,3,1,2,3,3,1,1...

cyclestring (): Allows you to create a group of words or characters using a complete cycle as an increment field. The whole string must be enclosed in quotation marks (" ") and each word or group of characters must be separated from the others by a semicolon (;).

**Example:**
cyclestring("Mon ; Tue ; Wed ; Thu ; Fri ; Sat ; Sun") = Mon Tue Wed Thu Fri Sat Sun

The following example is for labels that use all letters of the alphabet except for O and I.
cyclestring("A;B;C;D;E;F;G;H;J;K;L;M;N;P;Q;R;ST;U;V;W;X;Y;Z")

**exact («string_1», «string_2»):** Returns TRUE if the two strings are identical, FALSE if not.
Examples:

`exact("software","software") = 1`
`exact("software","software") = 0`

**extract («string», «sep», «pos»):** Returns the substring from the character string «string» at the specified position «pos» that contains data separated by string «sep».

**Example:**

`Extract("1;2;3;4", ";", 3) = 3`

**find («string», «key», «start»):** Returns the position of the first occurrence of the «key» argument in the «string» argument. The search in the «string» argument starts from the position returned by the «start» argument («start» >= 1). The function resets to zero if no occurrence of the «key» argument is found. The function distinguishes between upper and lower case letters.

**Example:**

`find("Peter McPeepert","P",1) = 1`
`find("Peter McPeepert","p",1) = 12`

**FormatNumber(number):** Allows you to format a numeric field where a pound sign (#) means only display if a value is there and zero (0) means always display.

**Examples:**

`FormatNumber(123.45, "US$ #,###,###.00") = US$ 123.45`
`FormatNumber(123.45, "US$ 0,000,000.00") = US$ 0,000,123.45`
`FormatNumber(.45, ",##0.00") = 0.45`
`FormatNumber(.45, ",###.00") = 45`
`FormatNumber(7188302335, "(###) ###-####") = (718) 830-2335`
`FormatNumber(123.45, "00.00") = 23.45`
`FormatNumber(123.567, ",###,##0.00") = 123.57`

**left («string», «num_char»):** Returns the character string extracted from the «string» argument. This string starts at position one of the «string» argument and has a length equal to the «num_char» argument.

**Example:**

`left("Peter McPeepert",1) = P`
`left("Peter McPeepert ",5) = Peter`

**LTrim(«string»):** Will automatically trim off any trailing spaces or leading spaces to the «left» data.

**Example:**

`LTrim(" No."); No`

**len («string»):** Gives the length of the «string» argument. Spaces are counted as characters.

**Example:**

`len("Paris, New York") = 15`
len("") = 0
len(" ") = 1

**lower («string»):** Converts all upper case letters in a text string into lower case letters.

**Example:**
lower("Paris, New York") = paris, new york

**mid («string», «start», «num_char»):** Returns the character string extracted from the string argument. This string starts at the position corresponding to the value of the start argument (start >= 1) and has a length equal to the num_char argument.

**Example:**

**pad («string», «length», «char»):** Adds characters to the left of the field to assign a predefined length to the whole input. Any character can be selected as a padding character.

**Example:**
If a field named GREETING displays a value HELLO, then:
pad(GREETING,8,0) = 000HELLO
pad(5,3,0) = 005
pad("Nine",6,"a") = aaNine

**replace («string», «start», «num_char», «new_string»):** Returns the converted string argument. A number (equal to the num_char argument) of characters from the position defined in the start argument has been replaced by the new_string argument.

**Example:**

**replaceString («string», «old_string», «new_string»):** Replaces all occurrences of a specified old_string in character string string with another specified new_string.

**Example:**
ReplaceString( "abc12def12", "12", "") = abcdef

**rept («string», «num_char»):** Returns the character string where the string argument is repeated the number of times in the num_char argument.

**Example:**
rept("Ah Paris! ",2) = Ah Paris! Ah Paris!

**right («string», «num_char»):** Gives the character string composed of the last characters of the string and has a length equal to the num_char argument.

**Example:**
right("Purchase order",5) = order
**RTrim («string»):** Will automatically trim off any trailing spaces or leading spaces to the right data.

**Example:**

RTrim("Part ") : Part

**search («string», «key, start»):** Gives the position of the first occurrence of the key argument in the string argument. The search starts from the position defined by the start argument (start >= 1). The function resets to zero if no occurrence of the key argument is found.

**Examples:**

search("Purchase order","order,1) = 10
search("Purchase order","c",1) = 4

**StrAfter(«data»,«start after», «length»):** This function results in a string that is exactly length characters long after a specified start after character.

**Examples:**

StrAfter("1234-5678", '-', 3)= Takes the 3 characters after the dash (567)
StrAfter("1234-5678", '-')= Takes all characters after the dash (5678)

**StrBefore(«data»,«start before», «length»):** This function results in a string that is exactly length characters long before a specified start before character.

**Examples:**

StrBefore("1234-5678", '-', 2)= Takes the 2 characters immediately before the dash (34)
StrBefore("1234-5678", '-')= Takes all characters before the dash (1234)

**trim («string»):** Returns the converted string argument. All spaces encountered at the beginning and end of the string are deleted. The number of spaces included between two words is reduced to one.

**Example:**

trim(" Purchase order") = Purchase order

**trimall («string»):** Returns the converted string argument. All spaces encountered are deleted.

**Example:**

trimall("Paris / New York / Rome") = Paris/NewYork/Rome

**upper («string»):** Gives the string converted into upper case.

**Example:**

upper("Purchase order") = PURCHASE ORDER

**ztrim («value»):** Deletes the zeros from the left side of the numerical value. Fields are entirely numerical.

**Example:**

If a field named WEIGHT displays a value of 000200, then:

ztrim(weight) = 200
Defining the properties of a Formula data source

Command: Data sources > Formula > formulaname > Properties

1. Enter the formula directly in the Edit field.
   - or -
   Select the elements, and click Insert.
2. Click Test to verify that the syntax is correct. If an error occurs, follow the instructions displayed on the screen and make any necessary changes.
3. Click OK.

Hint: You can insert an element by double-clicking on it.

Note: If a variable used in the formula has a name containing one of the following characters &+-*/<>^%!,"", it must be enclosed in brackets {}.

Note: You can check your formula by clicking Test. If the message displays the formula value, it means that your formula is correct. If the value is not correct, follow the instructions displayed on the screen to make the necessary modifications. If the value obtained is truncated, you must modify the maximum length specified in the Output tab.

Exercise: Creating a simple formula

Displaying the price of a product

In this example, the production label must show the price of the product as a function of its weight and the price per kilogram.

1. Open a label. Two variables have to be created: WEIGHT and PRICEPERKG.
2. For the WEIGHT variable, enter 788 (the weight of the product is 788 g) as the Local value, enter "Please enter the weight in g" in the Prefix field, then click OK.
3. For the PRICEPERKG variable, enter 15.70 (the price/kg is FF15.70) as the Local value of the variable
4. Enter "Please enter the price per kg" in the Prefix field, then click OK.
5. Add a formula and name it Price.
6. Enter the formula WEIGHT*PRICEPERKG/1000, then click OK.
7. Save your label.

Exercise: Adding the "Warning" formula variable to display a warning message

In the following exercise, we will create a formula to display a warning message telling the user that the value of the Total_Weight shared variable exceeds 1,000 kg.
If the weight value exceeds 1,000 kg, the message "Attention! Error! Total Weight exceeds maximum!" will appear.

1. Open the label.
2. Create a formula and name it "Warning".
3. In the Formula dialog box, enter the following expression: \( \text{if}(\text{Total\_Weight}>1000, \text{"Attention: Total Weight exceeds maximum!"}, \text{""}) \)
4. On the Output tab, enter 50 in the Maximum length field and click OK.
5. Position the variable as text within the label.
6. In the Text dialog box, select Scalable as the font and set its Height to 12.70 mm.
7. On the Paragraph tab, select the Wordwrap option, then select Centered in the Alignment field.

**Information on the IF function**

The IF function returns one value if the condition you specify is TRUE, and another value if it is FALSE.

Use the IF function to carry out conditional tests on values and formulas.

**Syntax**

\( \text{if}(\text{"expr"},\text{"val\_if\_true"},\text{"val\_if\_false"}) \)

- \( \text{expr} \) represents any value or expression, the result of which can be TRUE or FALSE.
- \( \text{val\_if\_true} \) is the value returned if \( \text{expr} \) is TRUE. The \( \text{val\_if\_true} \) argument can be another formula.
- \( \text{val\_if\_false} \) is the value returned if \( \text{expr} \) is FALSE. The \( \text{val\_if\_false} \) argument can be another formula.

**Exercise: Creating a specific modulo**

In this exercise, we will convert the "Customer\_Code" EAN8 barcode into a 2/5 Interleaved barcode using the "Formula\_4\_NewCustCode" Formula data source.

The barcode must have the following properties:

- Symbology: printer
- Height: 4 mm
- Narrow bar width: 1 mm
- Ratio: 2
- Human readable: below and centered
- Distance from the bars: 0 mm
- Characters font: printer font.

Open the ORDER\_WS2 label.

**To calculate the weight**
Create the Formula_1_Weighted formula, keeping in mind that the first character of Customer_Code is multiplied by 1, the second character by 2, the third character by 1, the fourth character by 2, and so on.

The maximum output length of the variable is 6.

Formula_1_Weighted:
mid(Customer_Code,1,1)*1&mid(Customer_Code,2,1)*2&mid(Customer_Code,3,1)
1&mid(Customer_Code,4,1)*2

To add up the result of the weight calculation:

The next step involves adding together the figures resulting from the previous formula, bearing in mind that the maximum permitted length of this character string is 2.

Create a second formula and name it Formula_2_Sum.

To calculate the check digit:

Using the previous result, we will now calculate the value of the check digit.

Create a third formula and name it Formula_3_CheckDigit.

The expression is as follows:
if ((Formula_2_Sum % 10) > 0,10 - Formula_2_Sum % 10,0)

To calculate the data to be encoded:

When creating the barcode, you must include the data to be encoded; for example, the value of the Customer_Code variable concatenated with the value of the check digit (Formula_3_CheckDigit).

Create a fourth formula and name it Formula_4_NewCustCode. This formula is the result of concatenating Customer_Code and Formula_3_CheckDigit.

To create the barcode:

1. Select the Formula_4_NewCustCode formula then drag and drop it into the label over the Customer_Code barcode.
2. Define the properties of the barcode.
Installing the network version

Description

To use the network/multi-user version of CODESOFTLABELVIEW, you must first install Network and Users Utilities, either on the server or on a workstation that will act as the server, then install your labeling software on each workstation.

Installing the dongle

The dongle must be installed on the computer (server or workstation) on which the license service is installed. A single dongle, pre-programmed with the number of licenses purchased, is included with the network version of your software.

The dongle must be installed before the license service can be started.

Note: When you start the program, if the dongle does not match the product version, a dialog box with the necessary instructions is displayed.

If you need to use a printer on the parallel port, connect it to the dongle. In this case, you might have to turn the printer on in order for the dongle to be recognized.

Network Installation procedure
Network configuration

Before you install the software, the network administrator must first define the structure of the network for the group of users:

- Define the license server on which the Network and Users Utilities and dongle will be installed.
- Define the workstations, or the client workstations that will use the labeling software.

Description of Network Manager

The Network and Users Utilities lets you use the network configuration of your labeling software. The Network Manager includes:

- The Network License Manager (License Service).
- Network Settings Wizard: The Network Settings Wizard helps you define the network configuration.
- User Manager: The User Manager is installed with the Network and Users Utilities so you can define access rights to the labeling software in a network setting.

Installing the Network and User Utilities on the server

Before installing the labeling software on all the workstations that will use it, you must install the Network and User Utilities on the server to configure the network.

1. Insert the DVD for the installation in the appropriate drive.
   The Installation window is displayed.
   
   If the DVD for the installation does not run automatically:
   Go to Windows explorer and expand the letter of the DVD drive. Double click on index.hta.
2. Select Network and Users Utilities, which includes License Service, and User Manager. Then click the Install button.
3. Follow the on-screen instructions..
4. Share the [TKDONGLE] folder with full control, using TKDONGLE as the share name.

   The default access path for this folder is:
   For XP: C:\Documents and Settings\All Users\ApplicationData\TKI\LicenseManager\TKDongle
   For Vista, 7, 8, 8.1, Server 2008 and Server 2012: C:\ProgramData\TKI\LicenseManager\TKDongle

   > RIGHT CLICK > Properties > Sharing tab and Permissions button.
5. If you want to define settings for your network configuration, start the Network Settings Wizard on the server. By default, if you do not modify the configuration, each workstation will have its own settings.

Note: For administrators: You can grant Write access to the network license by completing the following two steps:

1. Sharing the TKDongle folder and authorizing the user:
   For XP: C:\Documents and Settings\All Users\ApplicationData\TKI\LicenseManager\TKDongle
Installing the network version

For Vista, 7, 8, 8.1, Server 2008 and Server 2012: \C:\ProgramData\TKI\LicenseManager\TKDongle\.

> RIGHT CLICK > Properties > Sharing tab and Permissions button.

2. Give Write Access to the user in the Security tab of the TKDongle properties.

Configuration

All the necessary tools to configure the network version are available from the Network tool bar, which can be accessed from:

- For Windows versions older than Windows 8: Start > Programs > Teklynx > Network and Users Utilities > Network.
- For Windows 8 and 8.1 users: Access the search tool. To access and use the search tool, open the Charms menu by pressing Win [Windows key] + C on your keyboard or by swiping in the top-right or bottom-right corner. Once the Charms menu pops open, click/tap the Search button. Type Network and Users Utilities.

The Network Settings Wizard helps you define the settings for your network version.

1. To start the Network Settings Wizard, click on the icon.
2. In step 1 of the wizard, select a settings mode: Generic, By user or By station.
   - **Generic**: All users will use the same settings on all workstations. (user.ini).
   - **By user**: Each user can access his or her own settings on any workstation. (user name.ini).
   - **By station**: Each workstation has its own settings (station.ini).
3. In step 2, specify the location in which you want to store these settings. If you want to share these settings between various workstations, specify a network path that is accessible to all workstations.
4. In step 3, specify the location in which you want to store the shared data (variables, lists, printing logfile, etc.).

To configure the User Manager

If you want to define network access rights for all users of the labeling software, you must do so during installation (consult the User Manager help system). Click the User Manager icon available on the Network toolbar.

Starting the License Service

Before installing the labeling software on all workstations, you must be sure the License Service is started.

The License Service was installed as service called SLicense. You do not need to start it. In fact, SLicense starts when the workstation is turned on and runs as a background task as long as the workstation is on.
To start the License Service Controller

- Click the icon available on the Network toolbar.

- or

- Double-click the SLICENSECTRL.EXE file in the [DONGLE] folder in C:\[InstallDir]\COMMON\TOOLS\.

Note: For Windows versions older than Windows 8, to start License Manager automatically when a Windows session is started, copy the shortcut for the program to the Start > Programs > Startup menu in Windows.

Installing the software on the workstations

The labeling software must be installed on all the workstations on which it will be used.

To install the software on a workstation

1. Insert the DVD in the appropriate drive.
   The Installation window is displayed.

   If the DVD does not run automatically:
   Go to Windows explorer and expand the letter of the DVD drive. Double click on index.hta.

2. Select the product to be installed, click the Install button and follow the instructions on the screen.

3. Start the labeling software. A message is displayed to inform you that no dongle has been found. Click Yes to start the software.

4. From the Tools menu, choose Network Administration.

5. Enable Use Network License.

6. Click Modify to select the server on which the license manager and dongle are installed.
   - or -
   Click Browse to automatically search for the server on which the license manager is installed.

   If the network has already been configured, a message asking if you want to use the current network configuration is displayed.

7. If you want to modify or configure the network settings, click the Network Settings Wizard button.

8. Click OK.

9. Restart CODESOFT.