1

Report Designer

In this chapter we will describe in detail all the GUI tools contained in ELS-Script® package, along with their respective operations. In particular, we will introduce the user to the *Report Designer* application via a few simple tutorials, in which standard tabular and hierarchical record based, XML based and custom report samples will be constructed in various methods. This quick-start tutorial will be followed by a detailed exposition of all the GUI elements and tools, including the visual report designer together with all related HTML edit operations and tools, also the *Query Builder*, *SQL Editor* and *Data Shape Builder*. To help the user in acquiring a rudimentary and thorough understanding of the ELS-Script® software system, this exposition of features will be supported by illustrative examples and sample code.

In brief, we will discuss the following issues:

- Standard Report Template Wizard tool
- DHTML report design and edit tools
- Data row tabulation, group and summary definition tools
- Data access and retrieval definition tools
- Data field and expression insertion and format definition tools
- SCRIPT edit and manipulation tools
- Query form design and parameter option definition tools
- Report script compilation and diagnostic tools

A Quick Start

To deviate from all these computer manuals and books that were written in more or less non-technical legal languages and ended up in piles of dust never to be looked upon, we chose to write this manual in concise practical technical language. In our method, all concepts will be illustrated by concrete sample code segments or related pictures, so that the user can digest the material in a natural and painless manner. If you are a software developer, who never reads user guides or manuals, and figures the nuts and bolds of an application relying on some mysterious insight or intuition, well guess what? ELS-Script® software is specially designed for you. Nevertheless, we should recommend that you read at least this quick-start section of this chapter and follow chapter 2 and 3 to some extend. For those other developers who like reading very much, well then – "Welcome aboard Enterprise! Ship's library is one floor below the deck. Be careful on the starboard, reading at warp-speed may get you a little dizzy. By the way, I am reading Superstring Theory Vol.1, there are two more volumes, after which I'll read Spaces of Constant Curvature, and then follow up with Kerr Space. I sure want to read Fermat's Last Theorem, but Vol.5 is missing, I am sure Spoke has it. He's kind of frustrated why Vulcans didn't prove it first".

Copyright © 2003-2004 by Epsilon-Logic Systems

Getting back to our original subject, we will start with a brief introduction to ELS-Script®, acquainting the user with the precise essence of the software system itself. What exactly is ELS-Script®?
In a nutshell, ELS-Script® is a well-designed, flexible and powerful software solution that addresses the needs of both web based and traditional window's based reporting. In particular, this package to its bare minimum contains a flexible report engine component, called SCRIPT engine, and a sophisticated *Report Designer* application, which besides powerful report design tools also contains the SCRIPT compiler, so that eventually the user may debug and compile report scripts into binary form.

In this subsection we will cover the following topics:

- Running the *Report Designer* application and creating a new project,
- Creating a new data access connection and defining a query over this connection,
- Creating a query-based report using the *SRT Wizard*,
- Creating a hierarchical records report using the *SRT Wizard*,
- Creating an XML based report using the *SRT Wizard*,
- Creating a custom report script with the help of the *SRT Wizard*,
- Inserting a query into the report script,
- Inserting an *ELS-Row* into the report script,
- Inserting data fields or expressions into the report script,
- Finally, compiling and running the report.

Running the Report Designer

We will start the tutorial by outlining the first few things that the user must do to get started with a report project. Now, when you installed the ELS-Script® software the installer created program shortcuts in the **Start > Programs** Window's menu bar. In this collection of shortcuts, look for **Script++** item, and click on it to activate the *Report Designer* application. The application looks like Microsoft© InterDev 6.0, but other then the appearance it has nothing to do with InterDev. Essentially, we have designed the GUI of the *Report Designer* completely from scratch to more or less look like InterDev (simply because we had to follow Window's standards for designing GUI). Also, do not be alarmed by this analogy, the visual HTML editor of the **Script++** application is much more sophisticated than that of the InterDev's, and to some extend it surpasses Microsoft© FrontPage. In the future versions of ELS-Script®, we have plans to further improve this visual HTML editor user interface to surpass even Macromedia DreamWeaver application in terms of affinity and ease of use.
Getting back to the task of the report making, the first thing that the user must do in the **Script++** application is to create a report project. A report project will simply organize all your report files for your current development project into a single user-defined project folder. For example, you may have a Visual Basic application to handle order entry for your business, and you want to develop reports to be called by this VB application. In this case, you can make a single report project that will group together all these reports. We should point however, that ELS-Script® report engine is not restricted to one single report project or database connection, it can naturally handle multiple projects and even multiple data access connections inside one single report.
To create a report project, in the **Script++** application select **New Project** menu item under the **File** menu. This will prompt the **New Project** dialog shown in Figure 1.1 below.

Figure 1.1. Showing the New Project dialog

Copyright © 2003-2004 by Epsilon-Logic Systems

In this dialog, you must enter a name for your project, for example Proj1, and then optionally you may select a location for storage of the project files. Observe that as you type in the project name, the suggested folder name for your project will have the same name as the project. The default path for the location of this project folder is C:\ELSS\BIN\, which may be changed via the 3-dotted browse button on the right of the textbox. The example in the figure depicts that the user has selected an existing path C:\RepProjects.
After project name and location entry in this dialog, click the **OK** button to create the project and open it in the **Project Explorer** pane. The **Project Explorer** is the top-right pane in the *Report Designer* main window. Observe that this pane currently should show a tree consisting of **Workspace** root node, which contains child nodes **Proj1**, **Database**, **XML** and **Miscellaneous**.

Figure 1.2. Project directory structure

When you create a project, the *Report Designer* will create the project folder as you prescribed, along with the following subfolders: BIN, DATABASE, MISC and REPORTS. Moreover, the REPORTS and BIN folders themselves will contain subfolders Image and Include.
All report script files that you create for this project will reside in the REPORTS subfolder of the current project's folder, with the image files that are used in the reports, contained in the Image subfolder, while other *CSS* or include files in the Include subfolder. All database support files for the current project will reside in the DATABASE subfolder. This includes the data access connections, query statements, as well as SQL script files.
The use of the BIN subfolder will be described later on, when we get to compiling or building the reports into binary forms.

In addition to this directory structure (see Figure 1.2 for details), the new project process creates a project file with RPJ file extension. In particular, note that for the current project, the process created a project file with the filename Proj1.RPJ. We should emphasize that a project file is a simple text file that stores information about the report filenames and other content of the project.

Connecting to the Data Source

The report that we will create in the later subsection of this chapter will be used to display all the records of the Orders table of the Northwind.mdb database, which came with Microsoft© Access application. Therefore, before indulging more into the details of report creation and design, we will first define the data access connection followed by the creation of a fairly simple query object using the built-in query builder of the ELS-Script® software system.

Figure 1.3. Showing the New Database Connection dialog

To define a data access connection, select the **New DB Connection** menu item under the **File** menu. This will prompt the **New Database Connection** dialog shown in Figure 1.3. In this dialog you must define a user-friendly name for the connection in the **Connection Name** textbox, for example enter NWDB, and then select a **Data Access Type**. Depending on the data access type selected, different kind of dialogs will prompt as a result of

Copyright © 2003-2004 by Epsilon-Logic Systems

clicking the **OK** button. For now, make sure that the OLE DB Dynamic Data Access:ADO item is selected in this **Data Access Type** list-box, and then click the **OK** button.

This will prompt the standard Window's **Data Link Properties** dialog as shown in Figure 1.4 below. Since we are going to use an MS-Access database, therefore, from the **Provider** tab of this dialog, select the Microsoft Jet 4.0 OLE DB Provider item from the **OLE DB Provider** list, and then click the **Next** button.
This will simply switch the tab to the **Connection** tab, where you must define the data source location, see Figure 1.5 for more details.

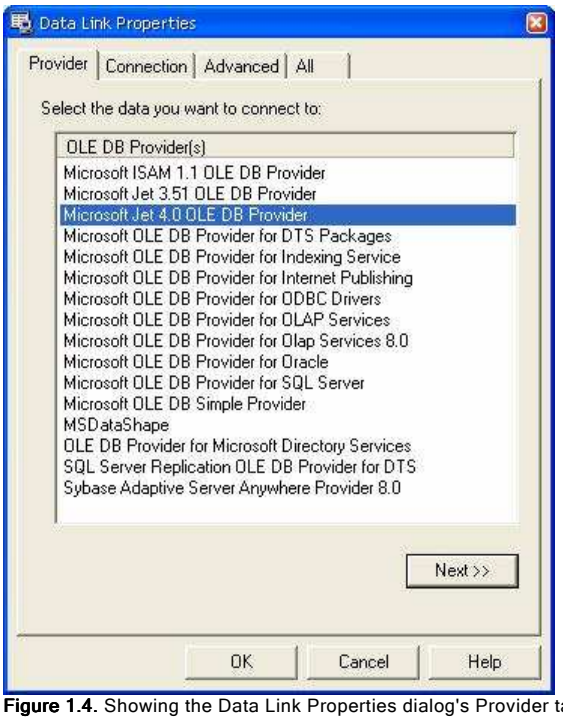


Figure 1.4. Showing the Data Link Properties dialog's Provider tab

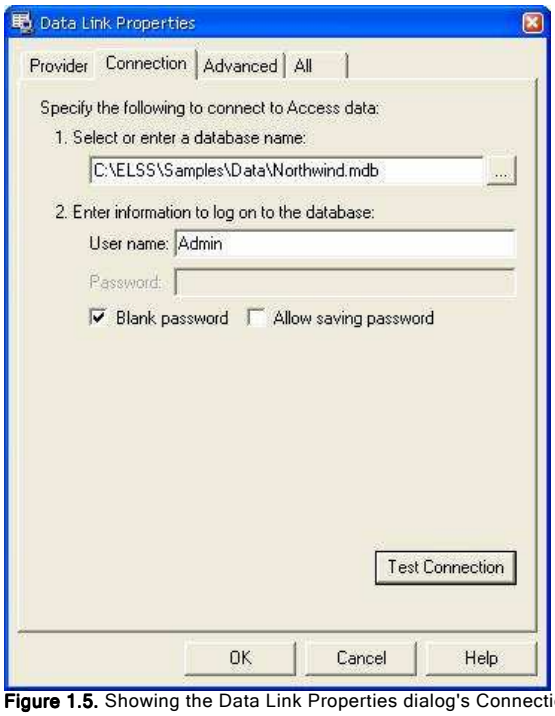


Figure 1.5. Showing the Data Link Properties dialog's Connection tab

In the **Connection** tab, click on the 3-dotted browse button, and select the Northwind.mdb file from the Data subdirectory of the Samples directory of the ELS-Script® software. Setting the rest of the controls as shown in Figure 1.5, click the **OK** button to create the data access connection (optionally, you may test the connection via the **Test Connection** button). This will create a NWDB connection node under the **Database** root-node of the **Project Explorer** pane, with **AUX** subfolder, which itself contains the **Queries** and **Datashapes** child nodes.

To define queries over this data access connection, we must open the connection first. To open a connection node that exists in the **Project Explorer** pane, you must double-click on the connection node.

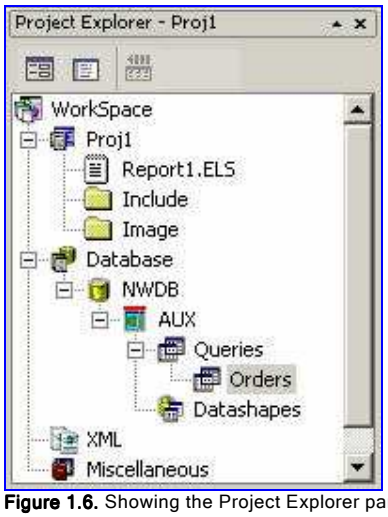


Figure 1.6. Showing the Project Explorer pane

For example, double-clicking on the NWDB node will open the corresponding database structure into the **Data View** pane. Note that the **Data View** is the pane located in the lower-right corner of the *Report Designer* window.
To create a new query, position the mouse on the **Queries** node and click the right mouse-button. This will display the popup menu of the **Project Explorer**. From the popup menu select the **New** menu item. This will create a new node NewQuery under the **Queries** node, and will open this query node as an instance of **Query Builder** window in the *Report Designer* application's main window. If you prefer query nodes to have user-friendly names instead of sequential names, you can rename the query node via the **Rename** command of the popup menu. Alternatively, you may rename a node by a second-click technique similar to Window's Explorer. For example, click the NewQuery node once, and then a second time, to make the label of the node editable, and then enter the name Orders for this query (as shown in Figure 1.6).

Copyright © 2003-2004 by Epsilon-Logic Systems

In the **Data View** pane, expand the tree by clicking on the (+)-sign icon. Note that there are three folder nodes under the **Database**, namely:

- **Tables**, which will display all table objects of the database,
- **Views**, which will display all the view objects of the database,
- **Procedures**, which will display all the stored-procedures of the database.

For example, expanding the **Tables** node, we will see the list of the table names of the Northwind.mdb database, as shown in Figure 1.7 on the right.
Incidentally, the **Data View** pane has features to expose almost everything about the database objects, including the actual structure of tables, views, stored-procedures, as well as retrieval of the respective source information. The availability of such features will depend dynamically on the backend database type used. For example, in the case when the backend database is SQL Server 2000 or IBM DB2, there will be an additional **Functions** folder that will list all the user-defined functions in the database.

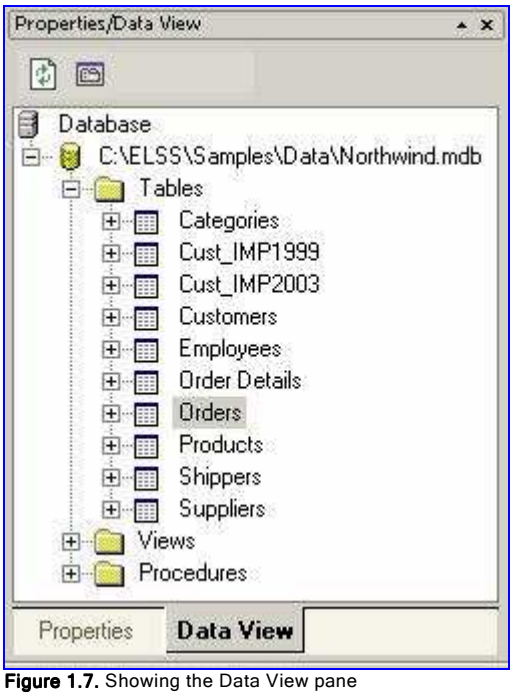


Figure 1.7. Showing the Data View pane

Now, recall that we created a query with an instance of **Query Builder** window open and waiting for us to engage, but instead we digressed into the details of the **Data View** pane. Now then, the story goes like this, the **Query Builder** window comprises of three tightly integrated panes, namely the **Relations**, **Columns** and **SQL** panes. Briefly said, the **Relations** pane handles relations between table or query objects, it also may be used to define the columns of the query. The **Columns** pane handles other parts of the query, including the definition of column list, conditions over fields, sorting and grouping. Finally, the **SQL** pane is an alternative way to directly edit the SQL statement itself in a command oriented fashion.

To proceed with the definition of the Orders query, drag-drop the Orders table from the **Data View** pane into the **Relations** pane, located at the top of the **Query Builder** window. This will draw a diagram of the table in the **Relations** pane, displaying all the fields of the table along with the "(All Columns)" option as items. Now, since we want a data source comprising of all the fields of Orders table, so we need to check the checkbox of the "(All Columns)" item. This will update the **SQL** pane to show the following text:

```
SELECT
        *
FROM
        Orders
```

Moreover, the entire **Query Builder** window will look like Figure 1.8 below.

In the later parts of this chapter, we will present a detailed exposition of the query builder tool that is built in the *Report Designer* application. We will also cover the *SQL Editor* with the corresponding built-in SQL functions for each database type. The *Report Designer* application also contains an integrated **Data Shape Builder** tool, with which the user may create data shape commands and drop them in the report script.
Also, please look at the online query builder help file in the *Report Designer* application. You may view this help file via the **Query Builder Help** menu item under the **Help** menu.

Some of the parts of the query construction may also be achieved via the *Standard Report Template Wizard* tool or *SRT Wizard* in short. To illustrate the power of the *SRT Wizard*, in the next subsection we will utilize the *SRT Wizard* to create a single record query based report using only point and click methods.

Copyright © 2003-2004 by Epsilon-Logic Systems

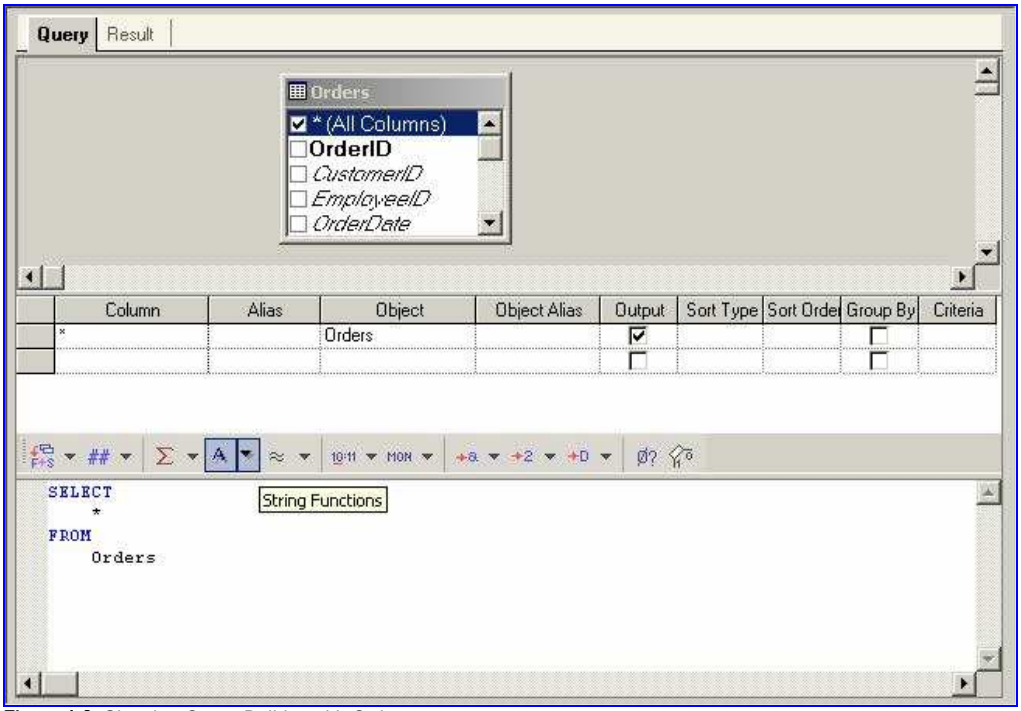
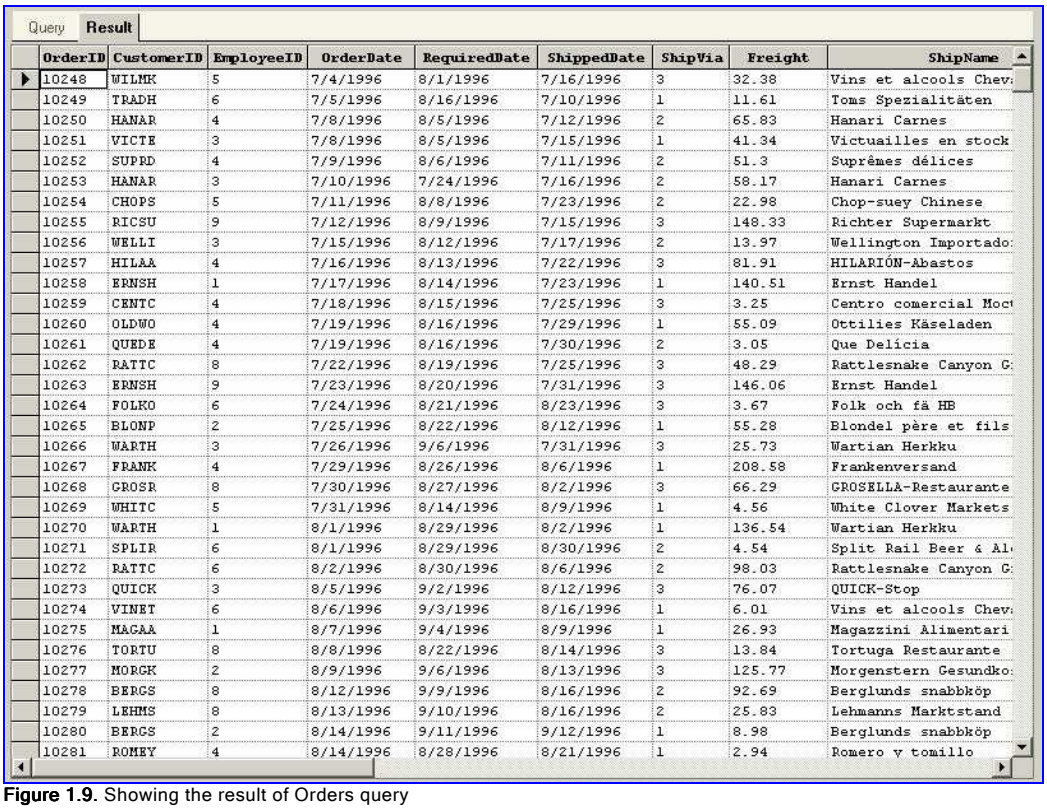


Figure 1.8. Showing Query Builder with Orders query open

For now, the query construction in this special case is complete and you are ready to execute the Orders query for the purpose of testing or debugging. To run the query click on the **Execute Query** green arrow toolbar button of the *Report Designer*'s main toolbar. The query result will be displayed in the **Result** tab of the **Query Builder** window, as shown in Figure 1.9.

When you are satisfied with the query definition, save it and you are done with the **Query Builder** window, therefore, you may close the Orders query window.



Query | Result

OrderID	CustomerID	EmployeeID	OrderDate	RequiredDate	ShippedDate	ShipVia	Freight	ShipName
10248	WILMK	5	7/4/1996	8/1/1996	7/16/1996	3	32.38	Vins et alcools Chev
10249	TRADH	6	7/5/1996	8/16/1996	7/10/1996	1	11.61	Toms Spezialitäten
10250	HANAR	4	7/8/1996	8/5/1996	7/12/1996	2	65.83	Hanari Carnes
10251	VICTE	3	7/8/1996	8/5/1996	7/15/1996	1	41.34	Victuailles en stock
10252	SUPRD	4	7/9/1996	8/6/1996	7/11/1996	2	51.3	Suprêmes délices
10253	HANAR	3	7/10/1996	7/24/1996	7/16/1996	2	58.17	Hanari Carnes
10254	CHOPS	5	7/11/1996	8/8/1996	7/23/1996	2	22.98	Chop-suey Chinese
10255	RICSU	9	7/12/1996	8/9/1996	7/15/1996	3	148.33	Richter Supermarkt
10256	WELLI	3	7/15/1996	8/12/1996	7/17/1996	2	13.97	Wellington Importado
10257	HILAA	4	7/16/1996	8/13/1996	7/22/1996	3	81.91	HILARIÓN-Abastos
10258	ERNSH	1	7/17/1996	8/14/1996	7/23/1996	1	140.51	Ernst Handel
10259	CENTC	4	7/18/1996	8/15/1996	7/25/1996	3	3.25	Centro comercial Moc
10260	OLDWO	4	7/19/1996	8/16/1996	7/29/1996	1	55.09	Ottilies Käseladen
10261	QUEDE	4	7/19/1996	8/16/1996	7/30/1996	2	3.05	Que Delícia
10262	RATTC	8	7/22/1996	8/19/1996	7/25/1996	3	48.29	Rattlesnake Canyon G
10263	ERNSH	9	7/23/1996	8/20/1996	7/31/1996	3	146.06	Ernst Handel
10264	FOLKO	6	7/24/1996	8/21/1996	8/23/1996	3	3.67	Folk och fä HB
10265	BLONP	2	7/25/1996	8/22/1996	8/12/1996	1	55.28	Blondel père et fils
10266	WARTH	3	7/26/1996	9/6/1996	7/31/1996	3	25.73	Wartian Herkku
10267	FRANK	4	7/29/1996	8/26/1996	8/6/1996	1	208.58	Frankenversand
10268	GROSR	8	7/30/1996	8/27/1996	8/2/1996	3	66.29	GROSELLA-Restaurante
10269	WHITC	5	7/31/1996	8/14/1996	8/9/1996	1	4.56	White Clover Markets
10270	WARTH	1	8/1/1996	8/29/1996	8/2/1996	1	136.54	Wartian Herkku
10271	SPLIR	6	8/1/1996	8/29/1996	8/30/1996	2	4.54	Split Rail Beer & Al
10272	RATTC	6	8/2/1996	8/30/1996	8/6/1996	2	98.03	Rattlesnake Canyon G
10273	QUICK	3	8/5/1996	9/2/1996	8/12/1996	3	76.07	QUICK-Stop
10274	VINET	6	8/6/1996	9/3/1996	8/16/1996	1	6.01	Vins et alcools Chev
10275	MAGAA	1	8/7/1996	9/4/1996	8/9/1996	1	26.93	Magazzini Alimentari
10276	TORTU	8	8/8/1996	8/22/1996	8/14/1996	3	13.84	Tortuga Restaurante
10277	MORGK	2	8/9/1996	9/6/1996	8/13/1996	3	125.77	Morgenstern Gesundko
10278	BERGS	8	8/12/1996	9/9/1996	8/16/1996	2	92.69	Berglunds snabbköp
10279	LEHMS	8	8/13/1996	9/10/1996	8/16/1996	2	25.83	Lehmanns Marktstand
10280	BERGS	2	8/14/1996	9/11/1996	9/12/1996	1	8.98	Berglunds snabbköp
10281	ROMEY	4	8/14/1996	8/28/1996	8/21/1996	1	2.94	Romero y tomillo

Figure 1.9. Showing the result of Orders query

Copyright © 2003-2004 by Epsilon-Logic Systems

Creating a Report via SRT Wizard

We now move on to create the first report script for the current project. To create a new report, select **New Report** menu item from the **File** menu. This will prompt the **New Report** dialog as shown in Figure 1.10 below. Note that by default the **Use Standard Template Wizard** radio-option is selected in the dialog, and a default suggested name for the report is already defined in the **Report Name** textbox.

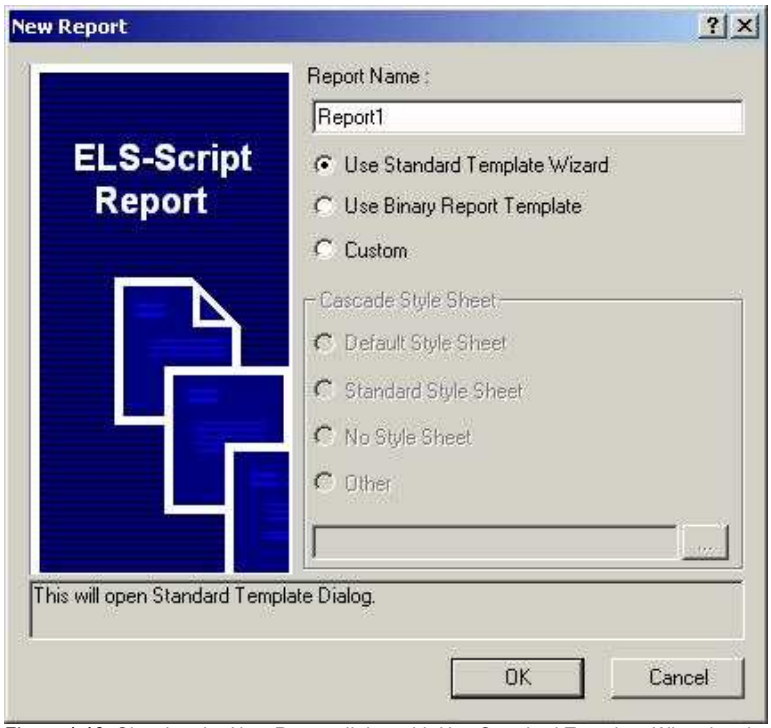


Figure 1.10. Showing the New Report dialog with Use Standard Template Wizard option

In a real situation, you may change the report name to your hearth's desires, but for this tutorial we will keep this sequentially defined name. Now, proceed by clicking the **OK** button. Since we have selected to use the standard report template wizard, the **Select Standard Template** window will be displayed, as shown in Figure 1.11 below.

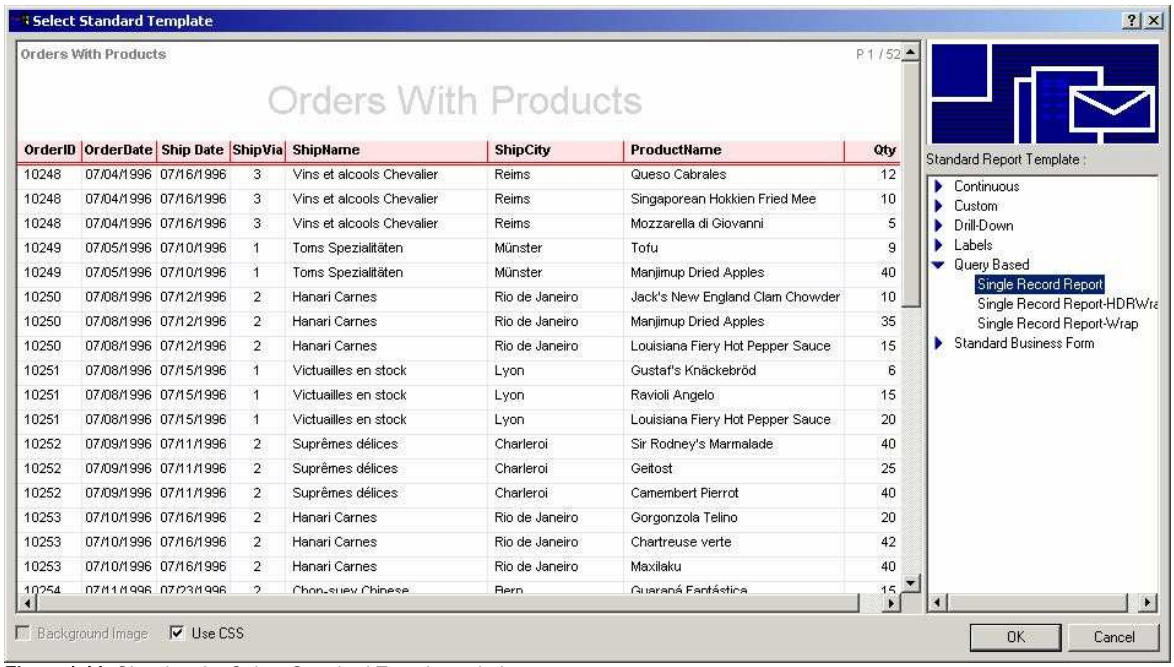


Figure 1.11. Showing the Select Standard Template window

Copyright © 2003-2004 by Epsilon-Logic Systems

Depending on the version of the ELS-Script® software that is installed on your system, you may browse through various standard report templates via this handy **Select Standard Template** window.
From this window select the Single Record Report template under the **Query Based** group, and leaving all other options as they are, click on the **OK** button. This will call the *SRT Wizard* dialog with the Single Record Report template activated, as shown in Figure 1.12 below.

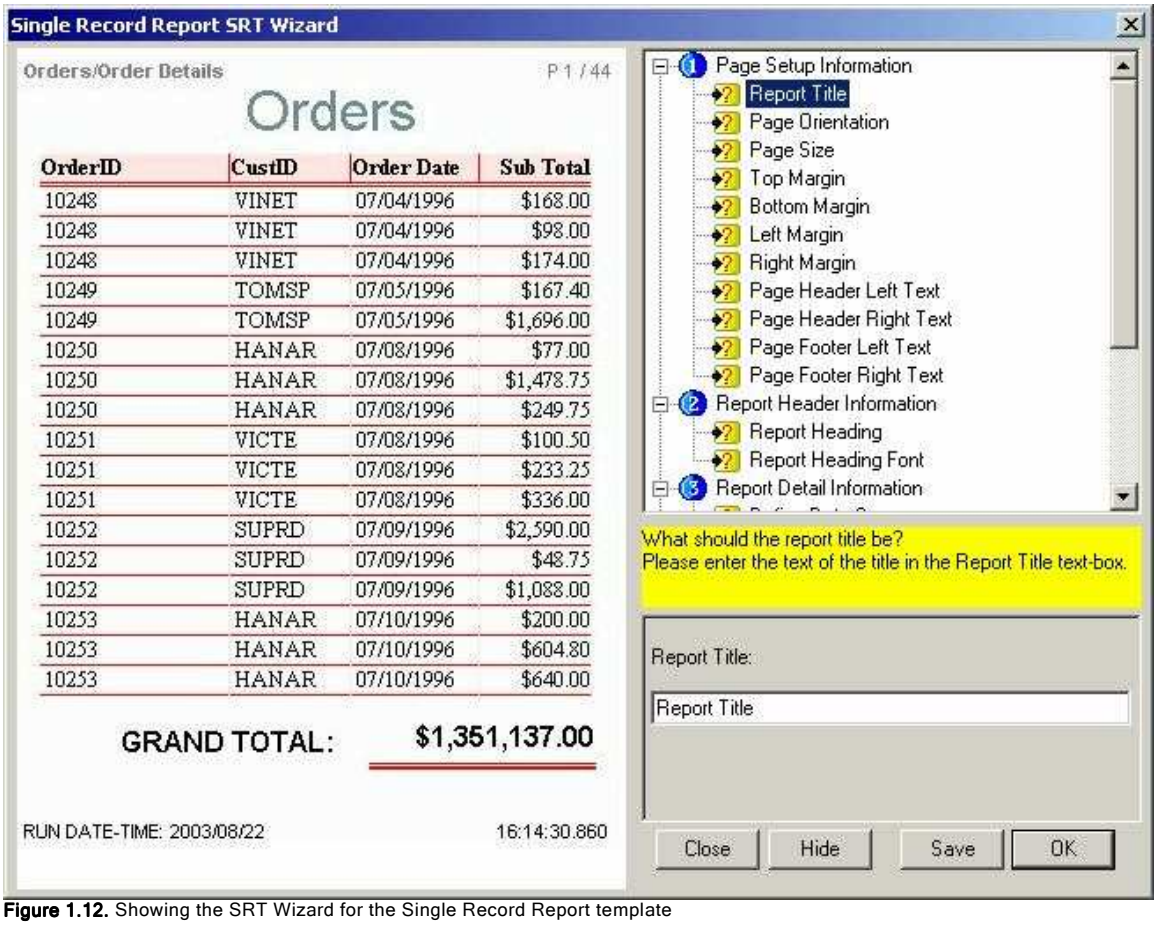


Figure 1.12. Showing the SRT Wizard for the Single Record Report template

The *SRT Wizard* is a smart tool that parses and analyses the selected template's *SRTmL* instructions and presents the intended requests in a linear and simple question/answer dialog. This dialog essentially has a tree-view pane in which the questions are represented by nodes under selective groups. The left side of the window displays the image that relates to the particular selected question. The image view also positions a yellow oval object to highlight the area in the image that relates to the selected question. The yellow rectangular region is where the text of the question is displayed, and below this question text the input controls are located. The user may enter the answers or perform the actions necessary using these input controls. The appearance of these input controls may vary, depending on the particular question that is selected in the tree-view.
Getting back to the creation of our first report via the *SRT Wizard*, note that the questions in the tree-view of the selected template's *SRT Wizard* dialog are organized into the following five groups:

- Page Setup Information
- Report Header Information
- Report Detail Information
- Report Summary/Footer Information
- Horizontal/Vertical Borders

We begin with the questions under the **Page Setup Information** group. In particular, the first question is about the internal report title and is requesting to enter the text for this title. In the input controls under the yellow rectangular section, enter the report title text in the text-box, for example enter the value "`Orders`".
Now, proceed to the next question by selecting the **Page Orientation** question node in the current group. Note that the input controls for this question comprise of a combo-box with default value of Portrait already selected. In this

Copyright © 2003-2004 by Epsilon-Logic Systems

case, please leave this default value and proceed to next question, which is about **Page Size**. The default value of this question is Letter 8 ½ x 11 in, which is what we will use. So, keep the default value as is, and proceed to the next question. Note that the next four questions are about page margins, namely, the **Top Margin**, **Left Margin**, **Right Margin** and the **Bottom Margin**. Again, keep the default values for these four questions and proceed to the next question. The next question is about the page header's left side text. For this question, enter some text that you would like to appear on the top-left side of all the pages of the report output, for example enter the value "`Orders By OrderID`". The next question is about the page header's right side text, which has the default value of "`P &Page / &PageCount`". This is a simple formula that will produce the pagination in the standard *page of page count* format, for example "P 1 / 92". You may keep this default value as is, and proceed to the next question, which is about the page footer's left side text. The default for this is the value "`RUN DATE-TIME: &RunDate(&RunTime)`". We may want to delete the "`(&RunTime)`" portion of this text, so that we can add the run-time at the right side of the page footer. In the **Page Footer Right Text** question, delete the existing default text and click on the **Insert run time** toolbar button (the button with the clock icon) so that the value "`&RunTime`" is inserted in the text-box, as shown in Figure 1.13 below:

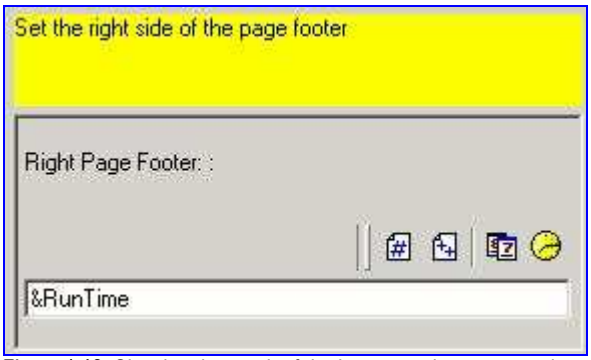


Figure 1.13. Showing the result of the Insert run time command

This last page setup question completes the first group of the questions, so that we can move on to the next group, which happens to be the **Report Header Information** group. The first question of this group is about the report header text that will be displayed at the beginning of the report output. For example, you may enter the following text: "`Orders By OrderID`". The next question is about the font type for this report header text. In this case, the input controls consist of the traditional *brown-fox* story sample text displaying the typeface effect, together with the **Select Font** button. Click this **Select Font** button to call the standard **Font** dialog. In the **Font** dialog, select the font type, size, color and other properties that you would like. For example, set the font size to 26pt with font color light grey.

The next group, **Report Detail Information**, contains the most important part of the wizard questions, namely the definition of the data source, the field columns and the parameter fields for the report. To define the data source, select the **Define Data Source** question node. This will update the input controls to the form displayed in Figure 1.14 below:

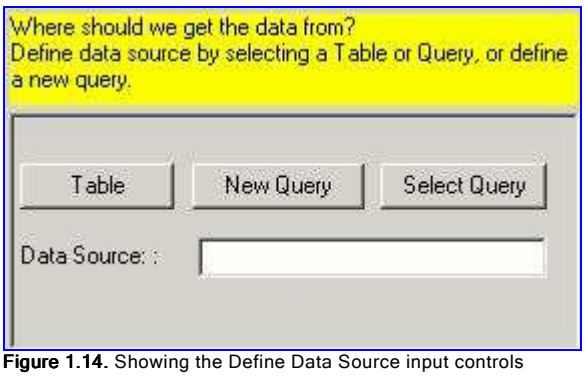


Figure 1.14. Showing the Define Data Source input controls

Observe that there are three ways to define a data source for the report's detail section. The **Table** button may be used to call the **Select Table** dialog, which lists all the table objects that exist in the selected backend database (see Figure 1.15 for more details). The **New Query** button will hide the *SRT Wizard* dialog, and then create and open an instance of the query builder window, so that the user may construct a new query. Finally, the **Select Query** button

Copyright © 2003-2004 by Epsilon-Logic Systems

will call the **Select Query** dialog, which displays a list of the existing queries under the currently open data connection.

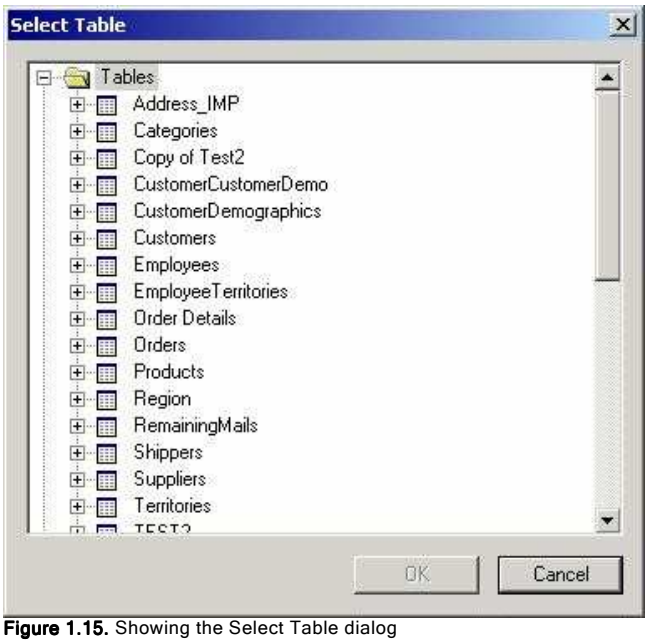


Figure 1.15. Showing the Select Table dialog

For the current report example, we will use the **Select Query** dialog to select the Orders query that we already created in the previous sections. Figure 1.16 shows this **Select Query** dialog with the Orders query selected. Note in particular that the upper pane of the **Select Query** dialog displays the SQL command of the query selected in the **Existing Queries** list-box. Therefore this dialog is a very handy tool to browse through the collection of all the queries that exist under the currently open data connection node.

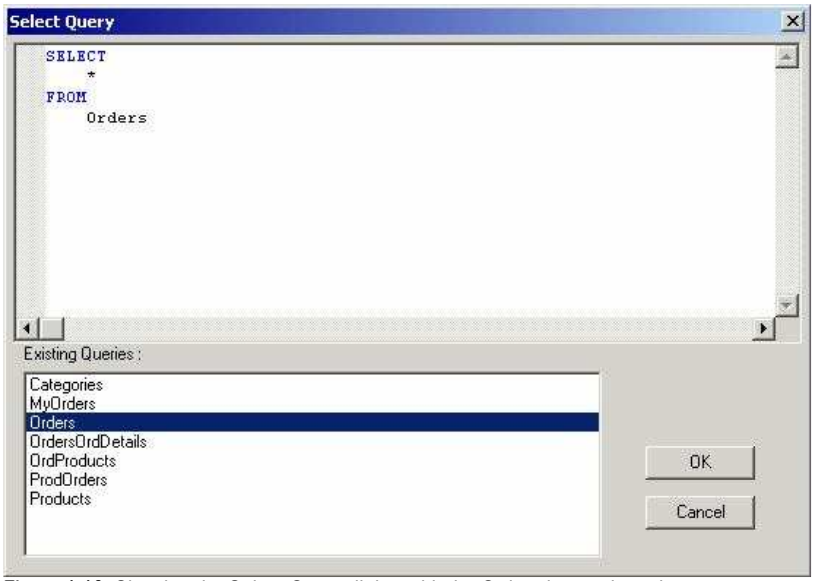


Figure 1.16. Showing the Select Query dialog with the Orders item selected

After selecting the Orders query and submitting the **Select Query** dialog, the **Data Source** text-box in the *SRT Wizard* should read as follows: "`QUERY: Orders`", which means that we have selected the existing Orders query.
The next question is about the parameter fields for the report. These are the variables that will make the report generation very similar to the execution of parameterized query, so that whenever the report is run, a dialog will prompt requesting some values against which the report output will be generated. For our special case, we want to define the *OrderID* field of the Orders data source as parameter field.

Therefore, to proceed, please select the **Define Parameter Fields** question node, which will update the input

Copyright © 2003-2004 by Epsilon-Logic Systems

controls. Click on the **Select Fields** button of these input controls to call the **Select Parameter Fields** dialog (see Figure 1.17 for details). Observe that the **Select Parameter Fields** dialog consists of the following controls:

All Fields	list-box, that contains the fields of the selected data source,
Add / *Remove*	toolbar buttons, to add fields to or remove fields from the **Selected Fields** grid,
Operator	list-box, to select an operator associated with the parameter,
Selected Fields	grid, that contains the fields selected as parameters organized in AND/OR matrix,

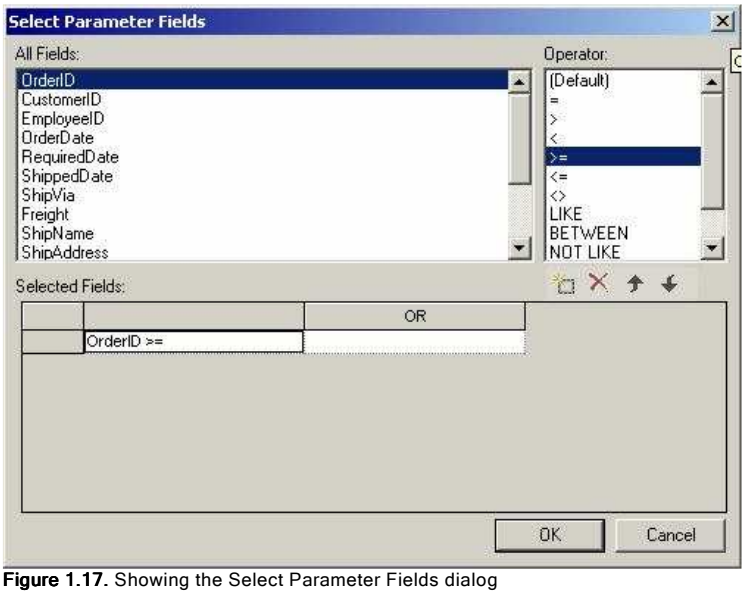


Figure 1.17. Showing the Select Parameter Fields dialog

To add a field to the **Selected Fields** grid, first select a field and an operator and then click on the **Add** toolbar button (i.e. first button from left). For example, select the "`>=`" operator in the **Operator** list-box, select the *OrderID* field in the **All Fields** list-box, and then click the **Add** button. This action should insert the item "`OrderID >=`" into the **Selected Fields** grid's first column.

Next, we submit this **Select Parameter Fields** dialog, and proceed to the next question concerning the column fields of the resulting report, which incidentally happens to be the most important question node of the *SRT Wizard*. The importance is based on several factors including the field selection and organization into columns, resizing the columns and rows to the appropriate page fitting, setting the font type, font size, color and background color of each column field along with the column header, horizontal alignment of the content of each cell, the definition of the column header, as well as a few other factors. All of these tasks may be performed by the functions and edit operations that are readily available in the **Select Fields** window.

To call the **Select Fields** window, select the **Define Column Fields** question node and in the input controls region click on the **Select Fields** button. Before we start using the **Select Fields** window, we take a moment to carefully study the features that are present in this sophisticated window. Observe in particular that this **Select Fields** window is a resizable window, which consists of the following control elements (see Figure 1.18 for more details):

Main Toolbar	this toolbar contains command controls that act upon the field grid,
Field List	this list-view will display all the fields of the data source selected in the prior data source question node,
Show Header	this check-box will include or exclude the column headers of the fields inserted into the field grid,
Field Grid Toolbar	this mini-toolbar contains the necessary commands to add, delete, move left or move right a selected field in the field grid,
Field Grid	this grid will contain all the fields that will be included in the final report script.

The *Main Toolbar* contains commands to set the font type, font size, font style, horizontal text alignment, fore color, back color, and format of selected fields in the *Field Grid*. The *Field List* essentially contains the names of all the fields of the data source defined in previous questions and corresponding to the current field selection question.

Copyright © 2003-2004 by Epsilon-Logic Systems

To add a field into the *Field Grid*, all you have to do is to select the field in the *Field List* and then click on the **Add** command button of the field list mini-toolbar, or alternatively just double-click the field item in the *Field List*. For example, Figure 1.18 depicts the insertion of the fields: *OrderID*, *OrderDate*, *ShippedDate*, *ShipVia*, *ShipName*, *ShipAddress*, *ShipCity* and *ShipPostalCode*.

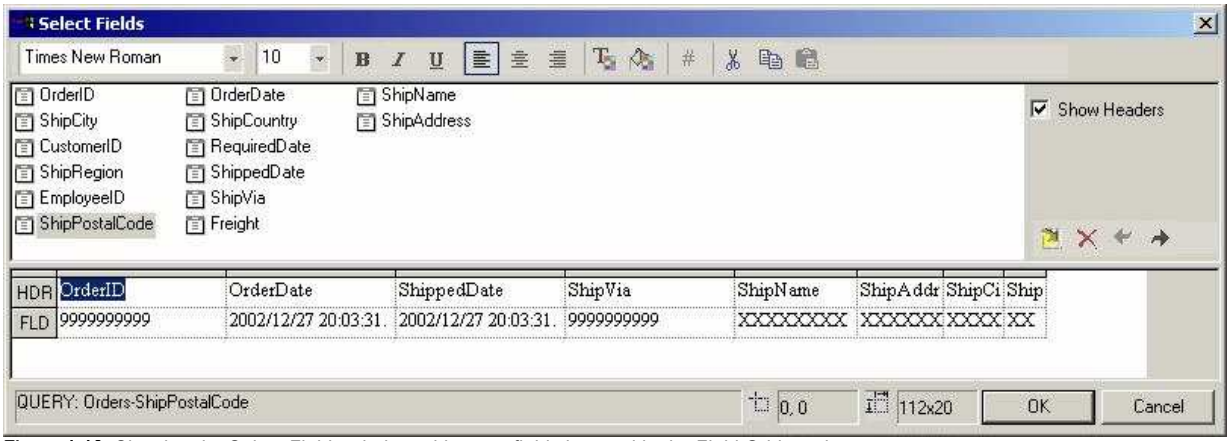


Figure 1.18. Showing the Select Fields window with some fields inserted in the Field Grid section

Note that when inserting a field, by default the field's name is put in the column header row of the grid. After insertion, however, you may select the header cell and enter your desired column header text directly via keyboard. For example, select the OrderDate column header cell and put a space between the "`Order`" and "`Date`" substrings of the "`OrderDate`" string, and similarly, for the other columns. You may apply the font properties on multiple cells at a time, for example, select both rows of the *Field Grid*, and then apply the font commands, making the typeface of type Arial 8pt. Then select the *HDR* row of the grid and apply the **Bold** button command so that the column header text becomes bold. While we are at it, click on the **Back Color** button to pull-down the color menu, and from this menu select the **More** item. This calls the **Color** dialog, as shown in Figure 1.19 below.

Figure 1.19. Showing the Color dialog with MISTYROSE color selected

In this dialog select the MISTYROSE color from the array of the additional colors, and then click the **OK** button. This will make all the selected *HDR* row cells to have the MISTYROSE background color.

We can also set the text alignment of multiple columns or cells at once. For example, select the columns Order Date, Shipped Date and Ship Via, and then click on the **Center** alignment toggle button in the main toolbar. This will center the text alignment in the selected cells.

We apply next the format command to format the date-time fields Order Date and Shipped Date. First click on the Order Date field's *FLD* cell, and then click on the **Format/Conversion** command button (i.e. the button with the #-icon). This action will call the **Format / Conversion Wizard** dialog, as shown in Figure 1.20. In this dialog, change the **Format** combo-box from General Date to Medium Date and then submit the dialog. As a result of this, the date format of the Order Date cell will be updated to a medium date format (e.g. "27-Dec-02").
In a similar manner, repeat this format modification process for the Shipped Date cell as well.

Next we set the format of the Ship Via cell. Therefore, select the *FLD* cell of this column, and click the format command button in the main toolbar. In the **Format / Conversion Wizard** dialog, the **Expression Type** combo-box will have Number type in this case. Proceed to the **Format** combo-box and enter "`##`" directly via keyboard,

Copyright © 2003-2004 by Epsilon-Logic Systems

and then hit the *Enter*-key, and finally submit the dialog by clicking the **OK** button. The result of the format operation will be shown in the *Field Grid* with the text "99" in the corresponding cell.

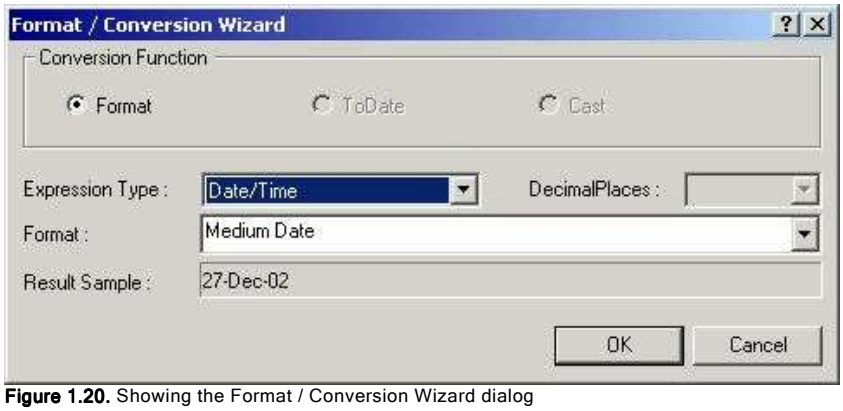


Figure 1.20. Showing the Format / Conversion Wizard dialog

Now, we are ready to resize the cells of the *Field Grid* to fit the page margins. Resizing in the *SRT Wizard* is an extremely efficient and rewarding task. All you need to do is to grab an edge of the column or row by pressing the left button of the mouse and holding on the button, resize the column or row. Note that the white rectangular region of the *Field Grid* represents the boundaries of the selected page margins, so that, when resizing you must take caution not to exceed the page width. For example, after resizing and fitting we obtain the result presented in Figure 1.21 below.

Figure 1.21. Showing the result of resize operations

At this point, we have completed the operations in the **Select Fields** window, therefore click on the **OK** button to submit the dialog. The result will be shown in the *SRT Wizard* by populating the **Selected Fields** list-box with the field names of the fields that we have selected in the **Select Fields** window.

The next group of questions is the **Report Summary/Footer Information** group. In this special case, given the nature of the data source that we have selected, we do not want any summaries or totals at the end of the report output, and therefore, we will skip this group and proceed to the next group.

The last group in the current *SRT Wizard* is the **Horizontal/Vertical Borders**, which requests whether or not you would like borders, and if so, you must specify the color of the borders. So select the **Column Header Borders** question node, this action will update the input controls to a pair of check-boxes with corresponding 3-dotted buttons to define the border color. For example, check both the **Vertical Borders** and the **Horizontal Borders** check-boxes, and then click on the 3-dotted button corresponding to the **Vertical Borders** check-box. This will call the **Color** dialog (see Figure 1.19 for details). In this dialog select the color RED and submit the dialog. Repeat this same procedure for the **Horizontal Borders**, so that the input controls are updated to look like Figure 1.22 below:

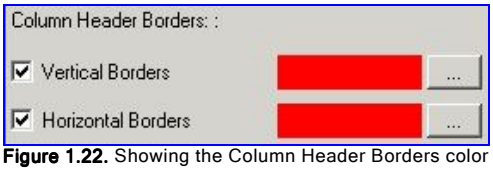


Figure 1.22. Showing the Column Header Borders color

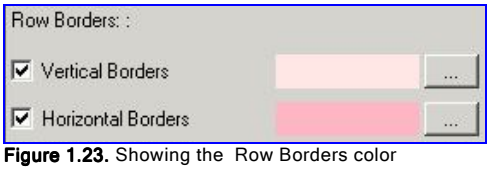


Figure 1.23. Showing the Row Borders color

Similarly, for the next question about the **Row Borders**, but this time select MISTYROSE for the **Vertical Borders** color and LIGHTPINK for the **Horizontal Borders** color.
The questions about the borders essentially complete the set of requests that are available in the current *SRT Wizard*. Therefore, submit the wizard window by clicking the **OK** button at the bottom of the *SRT Wizard* window. The wizard may prompt with the following message:

Copyright © 2003-2004 by Epsilon-Logic Systems

Some of the values necessary to generate a complete report script are still undefined.
Are you sure to proceed anyway?

In this case, this message prompt occurred because we skipped the **Report Summary/Footer Information** group of the wizard. Therefore, be confident and click on the **Yes** button of this message dialog to proceed with the report script generation process. The report script Report1.ELS will be generated and displayed in either the **Design** view or the **Source** view of the *Report Designer* main window.

To run the report, make sure that the **Source** view tab is active and then click on the **Run** button of the main toolbar (see Figure 1.24 for more details about this toolbar button).

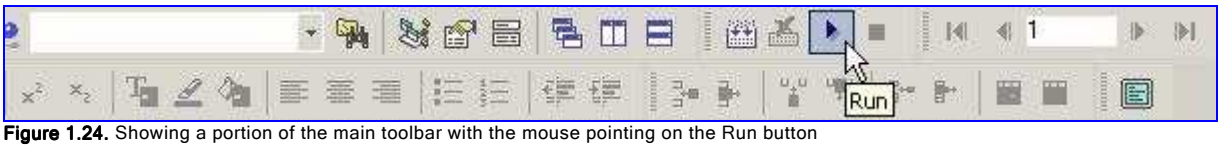


Figure 1.24. Showing a portion of the main toolbar with the mouse pointing on the Run button

The **Run** process will first perform a JIT (*Just-In-Time*) compilation of the report script into a binary form, then, it will prompt the **Evaluate Parameters** dialog, so that you may enter some value for *RWOrderID* parameter, and then immediately, it will generate the report output pages. The result will be displayed in the report viewer, as shown in Figure 1.25 below:

OrderID	Order Date	Shipped Date	Ship Via	Ship Name	Ship Address	Ship City	Zip Code
10248	04-Jul-96	16-Jul-96	3	Vins et alcools Chev	59 rue de l'Abbaye	Reims	51100
10249	05-Jul-96	10-Jul-96	1	Toms Spezialitäten	Luisenstr. 48	Münster	44087
10250	08-Jul-96	12-Jul-96	2	Hanari Carnes	Rua do Paço, 67	Rio de Janeiro	05454-876
10251	08-Jul-96	15-Jul-96	1	Victuailles en stock	2, rue du Commerce	Lyon	69004
10252	09-Jul-96	11-Jul-96	2	Suprêmes délices	Boulevard Tirou, 255	Charleroi	B-6000
10253	10-Jul-96	16-Jul-96	2	Hanari Carnes	Rua do Paço, 67	Rio de Janeiro	05454-876
10254	11-Jul-96	23-Jul-96	2	Chop-suey Chinese	Hauptstr. 31	Bern	3012
10255	12-Jul-96	15-Jul-96	3	Richter Supermarkt	Starenweg 5	Genève	1204
10256	15-Jul-96	17-Jul-96	2	Wellington Importad	Rua do Mercado, 12	Resende	08737-363
10257	16-Jul-96	22-Jul-96	3	HILARIÓN-Abastos	Carrera 22 con Ave. Carl	San Cristóbal	5022
10258	17-Jul-96	23-Jul-96	1	Ernst Handel	Kirchgasse 6	Graz	8010
10259	18-Jul-96	25-Jul-96	3	Centro comercial Mo	Sierras de Granada 9993	México D.F.	05022
10260	19-Jul-96	29-Jul-96	1	Ottilies Käseladen	Mehrheimerstr. 369	Köln	50739
10261	19-Jul-96	30-Jul-96	2	Que Delícia	Rua da Panificadora, 12	Rio de Janeiro	02389-673
10262	22-Jul-96	25-Jul-96	3	Rattlesnake Canyon (	2817 Milton Dr.	Albuquerque	87110
10263	23-Jul-96	31-Jul-96	3	Ernst Handel	Kirchgasse 6	Graz	8010
10264	24-Jul-96	23-Aug-96	3	Folk och fä HB	Åkergatan 24	Bräcke	S-844 67

Figure 1.25. Showing the report output's first page in the report viewer with the result pane visible

Looking at the report output in the report viewer, the first thing one may want to do is to hide the result pane at the bottom of the *Report Designer*'s main window, so that the report viewer is maximized to show more of the report output page. This may be done by clicking the **Show Results Pane** toggle button, as illustrated in Figure 1.26.
The next issue, that obviously may concern you at this moment, is – how do I navigate through the pages of the report output in the report viewer?

Copyright © 2003-2004 by Epsilon-Logic Systems

Figure 1.26. Showing the Show Results Pane together with the page navigation buttons

The answer to this question is – via the **First**, **Previous**, **Next** and **Last** page toolbar buttons. These buttons are the maroon colored left and right arrow buttons on the top-right corner of Figure 1.26.

Creating Hierarchical Records Report via SRT Wizard

In this section, we will use the *SRT Wizard* again, but this time we will follow another report template, which will create a *hierarchical records report* with a parent and child row structure.

To proceed, in the *Report Designer* application select the **New Report** menu item from the **File** menu. This will call the **New Report** dialog, prompting for a report name. Enter a report name in the **Report Name** textbox, or you may leave the sequentially defined default name, for example Report2.ELS. Also, make sure that the **Use Standard Template Wizard** radio-option is checked, and then click the **OK** button to submit the dialog. This will proceed with the **Select Standard Template** window. In this later window, select the **Hierarchical Records Report** standard report template, which you will find located under the **Drill-Down** group, and then click on the **OK** button. As a result of this action the *SRT Wizard* dialog will be displayed with the *Hierarchical Records Report* template activated.

Now, observe that in this standard template wizard, the collection of questions is organized into the following groups:

- Page Setup Information
- Report Header Information
- Report Detail Information
- Report Summary/Footer Information
- Field Alignment Information
- Horizontal/Vertical Borders

Most of the questions in these groups are very similar to those of the *Single Record Report* standard report template, which we already have covered in the previous subsection of the current chapter. Therefore, we will only elaborate on the questions which are different in this *Hierarchical Records Report* template case, while presenting all the other questions by simply listing the answers in a tabular format.

We begin by listing the values or answers that need to be entered for the questions in the **Page Setup Information** group. The following table represents this listing with suggested values:

Question Node	Suggested or Default Value
Report Title	`Orders With Products`
Page Orientation	`Portrait`
Page Size	`Letter 8 ½ x 11 in`
Top Margin	`0.75`
Left Margin	`0.50`
Right Margin	`0.50`
Bottom Margin	`0.50`
Page Header Left Text	`Orders / Products`
Page Header Right Text	`P &Page / &PageCount`
Page Footer Left Text	`RUN DATE-TIME: &RunDate (&RunTime)`
Page Footer Right Text	` `

The next group is the **Report Header Information**, for which you must evaluate the **Report Heading** and the **Report Heading Font** questions. For the **Report Heading** question, enter the text "`Orders With Products`", and for the **Report Heading Font** question, maintain the default values.

Copyright © 2003-2004 by Epsilon-Logic Systems

The questions in the **Report Detail Information** need more elaborate exposition, and therefore we will describe them one by one in more details.

The first question requests to define the data source for the main parent row in the resulting report. To proceed, select the **Define Parent Data Source** question node, and note that the input controls update to the three buttons **Table**, **New Query** and **Select Query**. Since the report that we want to create is based on the *Orders*, *Order Details* and *Products* tables of the Northwind database, and the parent row will be populated directly from the *Orders* table, we want to use the **Table** button. Clicking this later button will call the **Select Table** dialog displaying all the tables that exist in the Northwind database. From these tables select the *Orders* table and submit the **Select Table** dialog.

The next question is about the query parameters of the parent data source that we just defined. Selecting this **Define Parent Query Parameter Fields** question node will update the input controls to a list-box and a **Select Fields** button. Clicking this later button will display the **Select Parameter Fields** dialog, as shown in Figure 1.17 in the previous subsection. In this dialog, select the ">=" operator and the *OrderID* field to insert the "`OrderID >=`" item in the parameter fields list, and then submit the dialog.

The next question node is **Define Parent Column Fields**, which is similar to the previous tutorial example. Therefore, click on the **Select Fields** button to display the **Select Fields** window. In this later window make the selections and settings depicted in Figure 1.27 below:

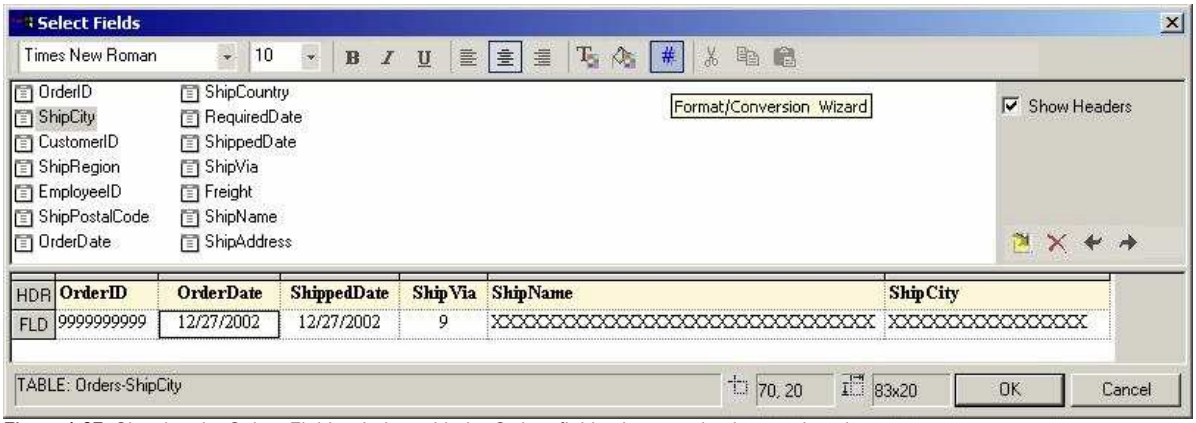


Figure 1.27. Showing the Select Fields window with the Orders field columns selections and settings

Observe in particular that we have selected the fields *OrderID*, *OrderDate*, *ShippedDate*, *ShipVia*, *ShipName* and *ShipCity*, we have set the *HDR* row text to bold and with CORNSILK yellow background color, and finally, the columns OrderDate, ShippedDate and ShipVia are centered and properly formatted.

The next few questions are about the child row data source, parameter fields and field columns. We will begin with the **Define Child Row Offset** question node. In hierarchical records presentations, sometimes leaving an offset at the beginning of child rows will make the report more appealing and much easier to read. This is the main reason behind the **Define Child Row Offset** question. Therefore, for this question you may maintain the default value of 100 pixels, and proceed to the next question about the child record data source definition.

Now, select the **Define Child Data Source** question node, and in the input controls click on the **New Query** button to define a new query comprising of a join of the Order Details and Products tables of the Northwind database. Note that the *SRT Wizard* creates an empty query node under the current connections **AUX\Queries** subfolder and displays it as an instance of the query builder window. Also, observe that this action will hide the *SRT Wizard* window without closing the current wizard session. So that, once the query is completed and tested, the user may get back to the *SRT Wizard* and continue the entry of the remaining questions.

In the query builder window, the first thing you must do is to rename the newly created empty query node, for example, name it OrdDetialProducts. This may be performed by pointing and clicking the mouse device on the new query node in the **AUX\Queries** subfolder under the current connection node in the **Project Explorer** pane, and clicking a second time on the node so that the node label becomes editable. After the query window title bar displays the new name of the query, in the **Data View** pane, expand the **Tables** folder to display all the tables of the Northwind database. From these tables drag and drop into the **Relations** pane, first the Order Details and then

Copyright © 2003-2004 by Epsilon-Logic Systems

the Products table. These actions will result to drawing the Order Details and Products diagrammatic objects in the **Relations** pane, with the related fields of tables automatically joined by a chord, as shown in Figure 1.28 below.

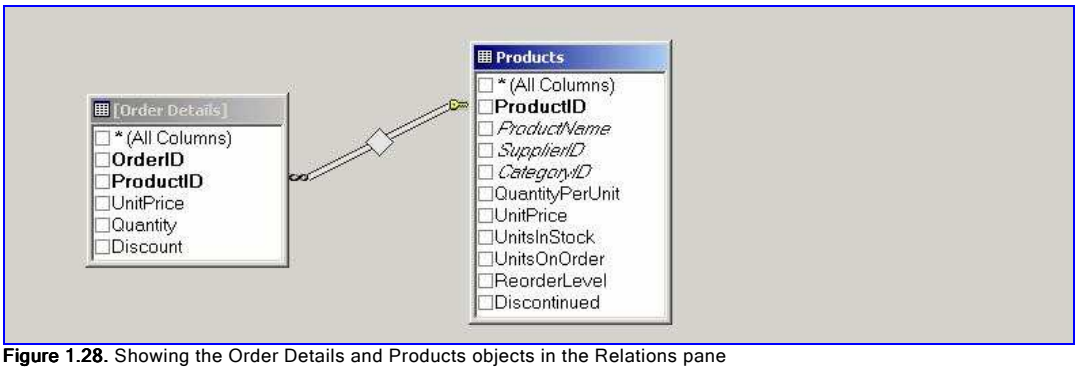


Figure 1.28. Showing the Order Details and Products objects in the Relations pane

The *Query Builder* module of the *Report Designer* is discussed in glorious details in a special query builder help document. You may look at this online help via the **Query Builder Help** menu command under the **Help** menu. For now, we will cover only the features that we will need in our task of creating the hierarchical records report via *SRT Wizard*. Therefore, the next step in defining the query is to select the field columns. This may be achieved by simply clicking on the desired field check-boxes of the diagrammatic objects in the **Relations** pane representing the tables. For example, click on the check-boxes of the fields *OrderID*, *ProductID*, *Quantity*, *UnitPrice* and *Discount* of the Order Details table object, and then click on the check-box of the *ProductName* field of the Products table object. After these selection actions, the result should look like Figure 1.29 below.

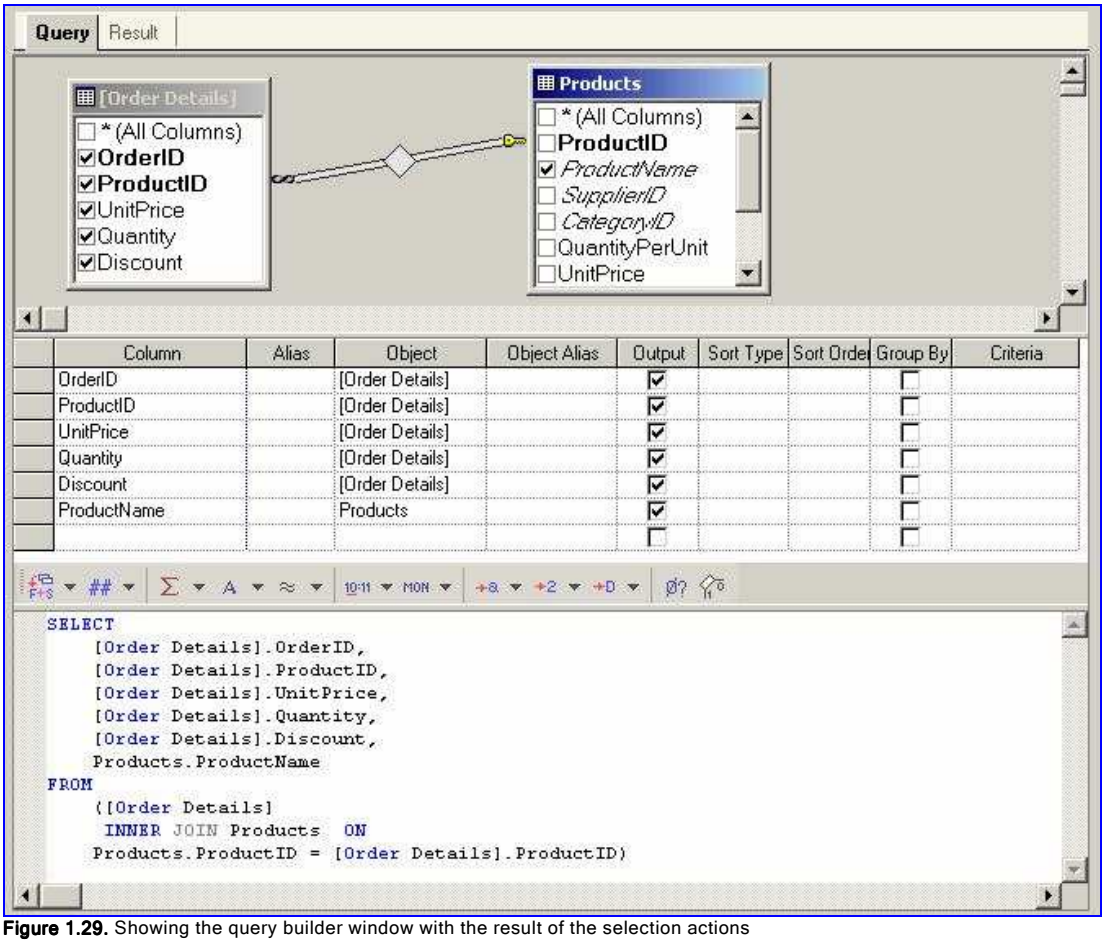


Figure 1.29. Showing the query builder window with the result of the selection actions

As a final step in completing the query, use copy/paste methods or direct typing to add the additional *SubTotal* field column, as shown by the yellow highlighted line in the SQL command below:

Copyright © 2003-2004 by Epsilon-Logic Systems

```
SELECT
     [Order Details].OrderID,
     [Order Details].ProductID,
     [Order Details].UnitPrice,
     [Order Details].Quantity,
     [Order Details].Discount,
     [Order Details].Quantity*([Order Details].UnitPrice - [Order Details].Discount) As SubTotal,
     Products.ProductName
FROM
     [Order Details]
      INNER JOIN Products  ON
     Products.ProductID = [Order Details].ProductID
```

Essentially, this completes the query definition, but before closing the query builder window and getting back to the *SRT Wizard*, it would be a good idea to test this new query command. To test this query, click on the **Execute Query** toolbar button, or simply click the **F5**-key on the keyboard. This will run the query displaying the result in the **Result** tab of the query builder window, as illustrated in Figure 1.30 below.

Query | **Result**

OrderID	ProductID	UnitPrice	Quantity	Discount	ProductName
11077	3	10	4	0	Aniseed Syrup
10309	4	17.6	20	0	Chef Anton's Cajun Seasoning
10326	4	17.6	24	0	Chef Anton's Cajun Seasoning
10336	4	17.6	18	0.1	Chef Anton's Cajun Seasoning
10339	4	17.6	10	0	Chef Anton's Cajun Seasoning
10344	4	17.6	35	0	Chef Anton's Cajun Seasoning
10464	4	17.6	16	0.2	Chef Anton's Cajun Seasoning
10511	4	22	50	0.15	Chef Anton's Cajun Seasoning
10527	4	22	50	0.1	Chef Anton's Cajun Seasoning
10533	4	22	50	0.05	Chef Anton's Cajun Seasoning
10606	4	22	20	0.2	Chef Anton's Cajun Seasoning
10635	4	22	10	0.1	Chef Anton's Cajun Seasoning
10636	4	22	25	0	Chef Anton's Cajun Seasoning
10654	4	22	12	0.1	Chef Anton's Cajun Seasoning
10704	4	22	6	0	Chef Anton's Cajun Seasoning
10726	4	22	25	0	Chef Anton's Cajun Seasoning
10846	4	22	21	0	Chef Anton's Cajun Seasoning
10913	4	22	30	0.25	Chef Anton's Cajun Seasoning
10950	4	22	5	0	Chef Anton's Cajun Seasoning
11000	4	22	25	0.25	Chef Anton's Cajun Seasoning
11077	4	22	1	0	Chef Anton's Cajun Seasoning
10258	5	17	65	0.2	Chef Anton's Gumbo Mix
10262	5	17	12	0.2	Chef Anton's Gumbo Mix
10290	5	17	20	0	Chef Anton's Gumbo Mix
10382	5	17	32	0	Chef Anton's Gumbo Mix
10635	5	21.35	15	0.1	Chef Anton's Gumbo Mix
10708	5	21.35	4	0	Chef Anton's Gumbo Mix
10848	5	21.35	30	0	Chef Anton's Gumbo Mix
10958	5	21.35	20	0	Chef Anton's Gumbo Mix
11030	5	21.35	70	0	Chef Anton's Gumbo Mix
11047	5	21.35	30	0.25	Chef Anton's Gumbo Mix

Figure 1.30. Showing the result of the query in the Result tab view of the query builder

Now, we can save the query and close the query builder window instance. To get back to the *SRT Wizard* window, from the **Tools** menu select the **Show Report Wizard** menu item. Observe, that in the *SRT Wizard*, the **Define Child Data Source** question is selected and the **Data Source** textbox in the input controls now has the value "`QUERY: OrdDetailsProducts`".

The next question is about the parameter fields for the child query. In this particular example of report, we do not need parameters for the child query, and therefore we may skip this step.
Now, select the **Define Child Column Fields** question node, and in the input controls region click on the **Select Fields** button. This will display the **Select Fields** dialog for the child data source, in which the user needs to select and configure the data fields. Observe, that in this later window, the width of the area that contains these column fields is reduced from the actual page size by the amount set in the previously defined row offset question node (which in this case is 100 pixels). In the **Select Fields** dialog select the desired fields, and set the text alignment, font, color, background color and the format of the data in each cell, so that for example, the dialog may look like

Copyright © 2003-2004 by Epsilon-Logic Systems

Figure 1.31 below.

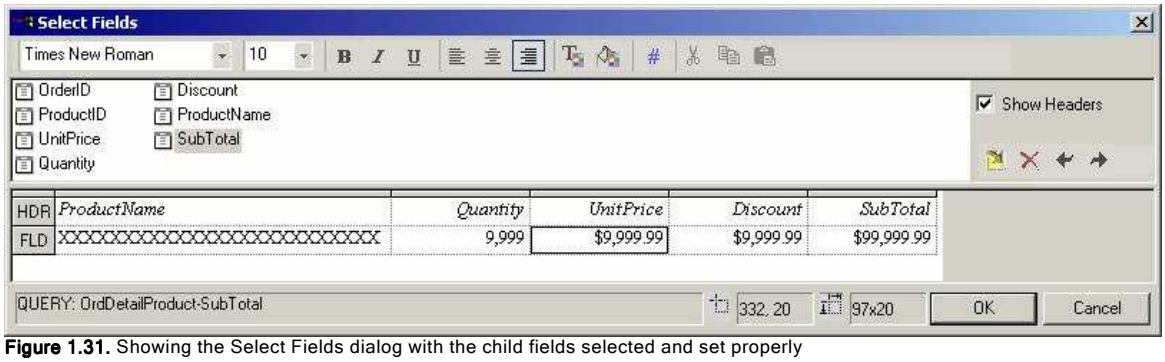


Figure 1.31. Showing the Select Fields dialog with the child fields selected and set properly

Note that in particular we have selected the fields *ProductName*, *Quantity*, *UnitPrice*, *Discount* and *SubTotal*., and have set the *HDR* row to italic typeface, and finally the last four columns are right aligned and the corresponding data cells properly formatted.

The next question is about the relationship of the parent and child data sources. In particular, select the **Define Parent/Child Relationship** question node, and in the input controls region click on the **Relate Fields** button. This will display the **Define Relation** dialog, in which you may relate the *OrderID* fields between the Orders table and the OrdDetailsProducts query. This may be performed by selecting OrderID item in both the **Parent Fields** and the **Child Fields/Parameters** combo-boxes, and then clicking on the **Add** button, to insert a relation item in the list-box, as depicted in Figure 1.32 below.

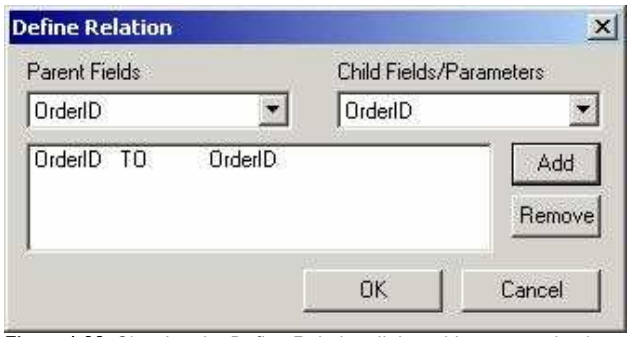


Figure 1.32. Showing the Define Relation dialog with proper selection

Submitting the **Define Relation** dialog will update the **Relationship** textbox of the input controls in the *SRT Wizard* window, by simply showing the selected "`OrderID TO OrderID`" relationship.

We continue to answer the questions of the next group, which is the **Report Summary/Footer Information**. For the **Define Child Group Summary Fields** question node, click on the **Select Fields** button to call the **Select Fields** window. In this later window, add a label cell by clicking the **Add Cell** command in the *Field List* mini-toolbar, and then add the *SubTotal* field from the *Field List* by double-clicking the field in the *Field List* list-box. Then make these cells to have bold typeface, right align the text, and apply the format on the *SubTotal* column, as shown in Figure 1.33 below.

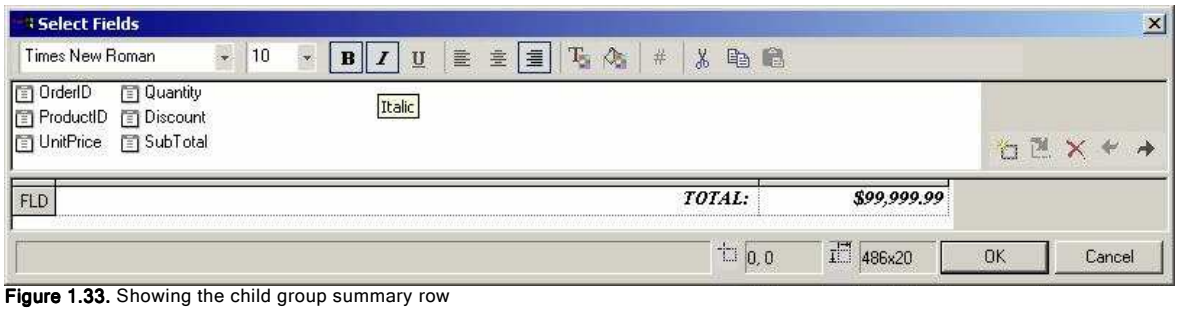


Figure 1.33. Showing the child group summary row

Copyright © 2003-2004 by Epsilon-Logic Systems

Note that in similarity with the child row, the width of the region that contains the fields in the field grid section of the window is reduced by an amount defined by the row offset. Perform similar operations for the **Grand Total Summary Fields** question node, with the label text having the text "`GRAND TOTAL:`".

The next question group, namely **Field Alignment Information**, may be used to adjust the widths of the fields relative to the other field-row groups, so that the overall alignments of the field columns become more proper, in this way, improving the appearance of the report output. To apply field alignment, from the input control of the question click on the **Align Fields** button. This will display the **Align Fields** dialog as shown in Figure 1.34 below:

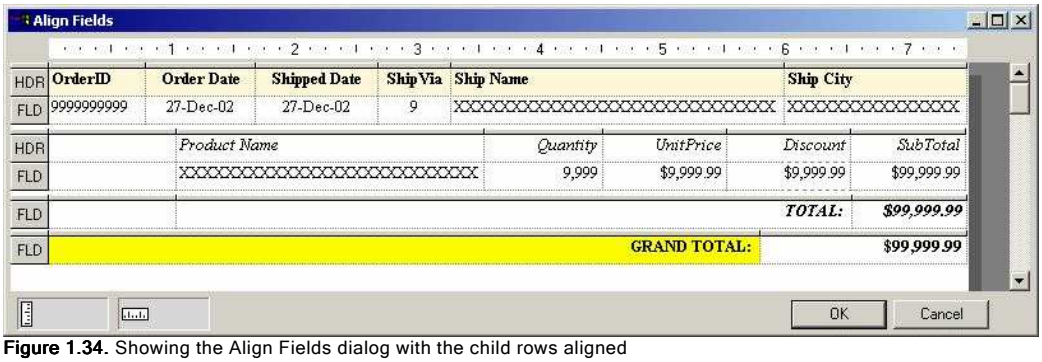


Figure 1.34. Showing the Align Fields dialog with the child rows aligned

The next question group is about the vertical and horizontal borders. For this group, the following table outlines some suggested values.

Question Node	Suggested or Default Value
Parent Column Header Borders	`Both vertical and Horizontal, color GRAY`
Parent Row Borders	`Both vertical and Horizontal, color GRAY`
Parent Summary Borders	`Only Horizontal, color GRAY`
Child Column Header Borders	`Both vertical and Horizontal, color SILVER`
Child Row Borders	`Both vertical and Horizontal, color SILVER`
Child Summary Borders	`Only Horizontal, color GRAY`

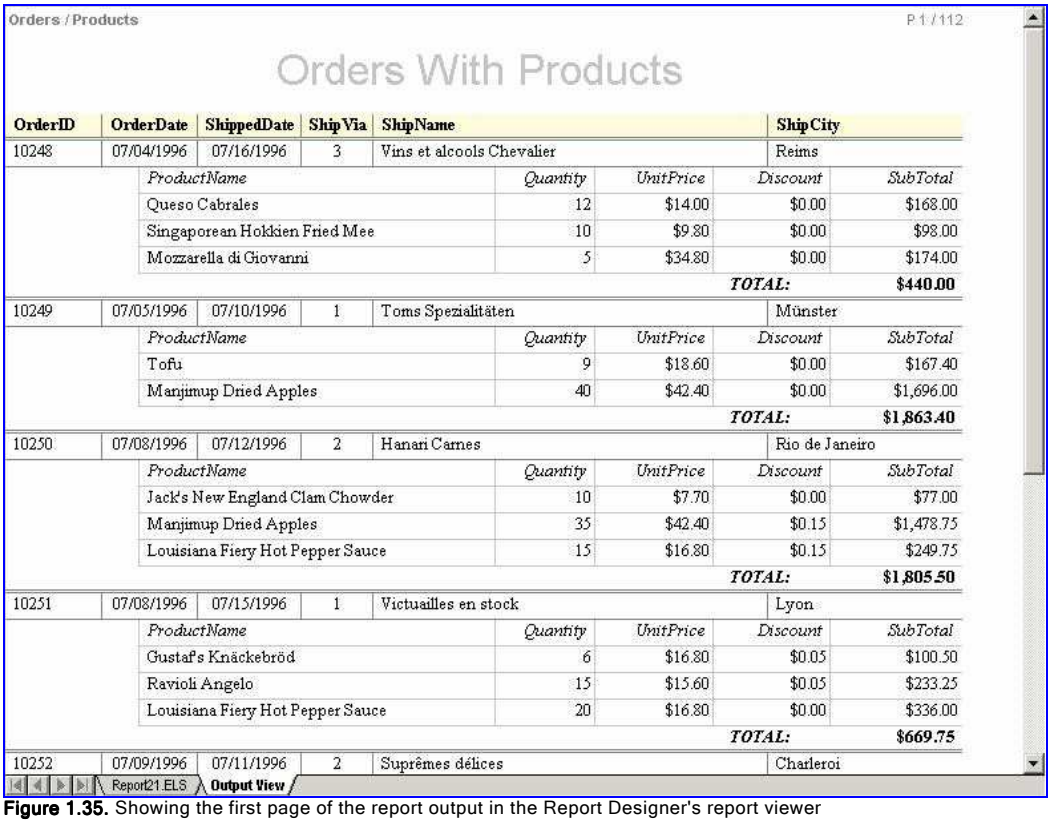


Figure 1.35. Showing the first page of the report output in the Report Designer's report viewer

Copyright © 2003-2004 by Epsilon-Logic Systems

At this point, we have completed all the requested questions of this wizard, and we are ready to generate the report script. Therefore, click on the **OK** button to generate the report script. Again, it may prompt a message that some questions are not complete, to which you should click on the **Yes** button to continue anyways.

After generating the report script, make sure that the **Source** tab is active and then click on the **Run** toolbar button in the main toolbar of the *Report Designer* application (see Figure 1.24 and 1.26 for more details) to run the report output generation process. The report output will be generated and the first page of the output will be displayed in the report viewer, as shown in Figure 1.35.

Creating Custom Report

In this section, we will create yet another report script. But this time, the *SRT Wizard* will be used to jumpstart the report script, creating the page setup information of the report settings, while the rest of the report will be left empty. The user then may develop the report based on the custom requirements of the particular business object.

Therefore, start with a new report by selecting the **New Report** menu item from the **File** menu. And then, in the **New Report** dialog enter a name for the report, and leaving all other options the defaults, submit the dialog. In the **Select Standard Template** window select the **Basic01** template under the **Custom** group node. Clicking the **OK** button will display the *SRT Wizard* for the **Basic01** template.

The first thing that you must observe is that this *SRT Wizard* will only help you on the **Page Setup Information** and the **Report Header Information**, leaving all the remaining report tasks at the mercy of your desires and imagination. We will see shortly, what exactly the term: "at the mercy of your desires and imagination" could imply, but for now, examine the following table of suggested entry values for both, page setup and report header groups:

Question Node	Suggested or Default Value
Report Title	`Invoices`
Page Orientation	`Portrait`
Page Size	`Letter 8 ½ x 11 in`
Top Margin	`0.75`
Left Margin	`0.50`
Right Margin	`0.50`
Bottom Margin	`0.50`
Page Header Left Text	`Orders`
Page Header Right Text	`P &Page / &PageCount`
Page Footer Left Text	`RUN DATE-TIME: &RunDate (&RunTime)`
Page Footer Right Text	` `
Report Heading	`Invoices`
Report Heading Font	`Arial 26pt SILVER`

After entering all values for respective questions, click on the **OK** button of the *SRT Wizard* window, to generate the empty report script. At this point the report should look like Figure 1.36, when the **Design** view tab is activated.

Observe that the wizard has already defined the page header and footer text, the report heading title and the page setup information. You may get some idea about the page setup when looking at the horizontal ruler on the top of this **Design** view, but for a more precise investigation, click on the **Source** view tab to see the actual report script. In particular, the page setup information is contained between the `<ELS_RSETTINGS>` and `</ELS_RSETTINGS>` tags. Observe also that the report detail section, which is section **D** in the **Design** view, is completely empty.

In the following paragraphs we will describe, how to add data to your custom report script, and then present this data in a tabular format via *ELS-Row* element. This tutorial will be about sales invoices based on the Northwind database that comes with the MS-Access application. The creation of the current report script will be a little more involved than the previous two tutorials, and therefore, in our presentation we will inevitably reveal some advanced features of the *Report Designer*, as well as, some aspects of the SCRIPT language itself.

Copyright © 2003-2004 by Epsilon-Logic Systems

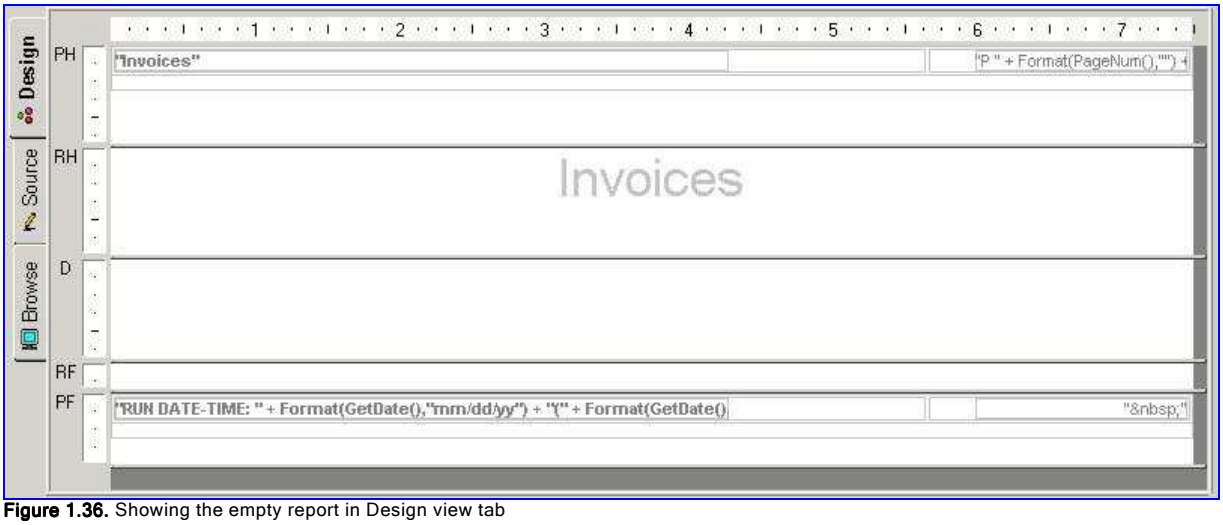


Figure 1.36. Showing the empty report in Design view tab

We continue the report design process by creating a new query via the *Query Builder* tool of the *Report Designer*. Use the right-mouse-button click method on the **Queries** subfolder of the **NWDB** connection node to display the popup menu. Then select the **New** menu command to create and open an instance of the query builder window in the *Report Designer* application's main window.
Now, expand the **Views** folder in the **Data View** pane, and from the view items drag-drop the *Invoices* view object into the **Relations** pane of the query builder window. Next, rename the query as InviocesQry, and check that the resulting SQL command in the **SQL** pane of the query builder window is as follows:

```
SELECT
    *
FROM
    Invoices
```

Recall that, in previous tutorials when defining the data source, the user only specifies the query, while the *SRT Wizard* will automatically construct and insert the query command into the report script. In contrast, when developing a custom report, we need to manually insert the query command at a proper location in the report script.

In general, to include a query command in a report script, all one has to do is to drag-drop the query node from **Project Explorer** into the report script. In our current case, we want to insert the InvoicesQry query into the Report3.ELS report script. To do so, make sure that the report is open in the *Report Designer* application, and then switch to the **Source** view to see the source of the report script. This is necessary, because insertion of query definition must be carefully put in proper location in the script, and therefore cannot be arbitrary.

In the current report case, we will insert InvoicesQry query inside the `<ELS_RSETTINGS>` report section. Therefore, in the **Source** view find the end-tag `</ELS_RSETTINGS>`, and position the cursor in front of this end-tag. Then press the *Enter* key a couple of times to make two empty lines before this end-tag. Then, using the mouse, drag-drop the InvoicesQry query node from the **Project Explorer** into the report script at the empty line preceding the `</ELS_RSETTINGS>` end-tag. This will insert a `DATASOURCE` variable declaration followed by a `SET` statement defining the query, so that the code in between the `<ELS_RSETTINGS>` and `</ELS_RSETTINGS>` tags will now look like the following:

```
    <ELS_RSETTINGS>
            SET REPORT_TITLE      = "Invoices";
            SET PAGE_ORIENTATION  = ELS_PORTRAIT;
            SET PAGE_SIZE         = ELS_LETTER;
            SET PAGE_SOURCE       = ELS_UPPER;
            SET PAGE_MARGINS.LEFT = 0.50;
            SET PAGE_MARGINS.RIGHT = 0.50;
            SET PAGE_MARGINS.TOP  = 0.75;
            SET PAGE_MARGINS.BOTTOM = 0.50;
            SET DEFAULTMODE       = ELS_FAST;
```


Copyright © 2003-2004 by Epsilon-Logic Systems

```
        DECLARE @InvoicesQry DATASOURCE;
        SET @InvoicesQry = "SELECT " +
                                " * " +
                        "FROM " +
                                "Invoices"
</ELS_RSETTINGS>
```

At this point, you have completed the inclusion of the InvoicesQry query in the report script, you may save the report to update your work. This is how you include data source inside a report. We will see later on in Chapter 2, how to adjust the data source to have the flexibility of passing a parameter from the host application.

Inserting ELS-Rows

Now assuming that the Report3.ELS is still open in the *Report Designer*, switch to the **Design** view. In this view position the cursor in the detail section of the report, this is the pane with a **D** in front of it. Then select the **ELS Row** menu item from the **Insert** menu, which will call the **Insert ELS-Row** window. In this **Insert ELS-Row** window, from the **Datasource** combo-box select the @InvoicesQry item (which, incidentally, is the only data source defined in the current report), and then click on the **Add Line** toolbar button at the top-right corner of the window (note that this button is the second button from the left of the toolbar). This **Add Line** command will prompt the **Insert ELS-Line** dialog. In this dialog, set the line attribute **Width** to 100%, the **Columns** to 7, and the **Border size** to 0, as shown in Figure 1.37 below.

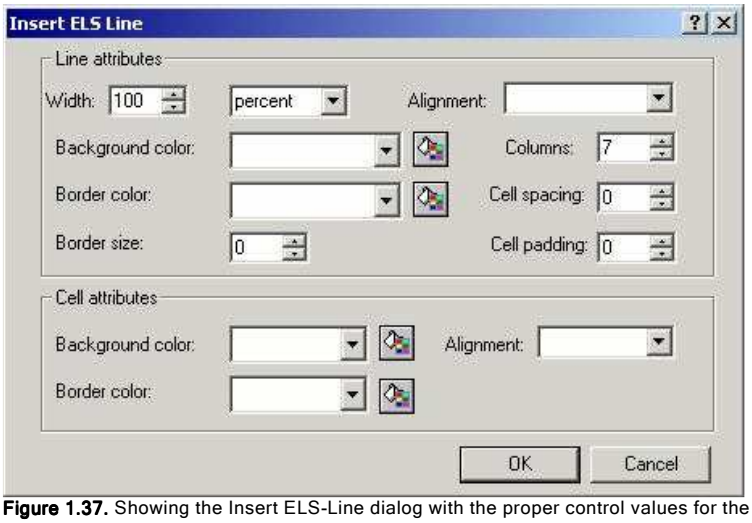


Figure 1.37. Showing the Insert ELS-Line dialog with the proper control values for the example

Clicking the **OK** button of this dialog will insert an *ELS-Line* element in the **Insert ELS-Row** window, as shown in Figure 1.38 below. In similar fashion, add another *ELS-Line*, with **Columns** = 1 and having 100% width and border size 0.

We should caution the user that since for all the added *ELS-Lines* the border sizes were set to 0 and the cell background was by default transparent, consequently these *ELS-Line* elements, although inserted, seem to be invisible. In fact, clicking in the top of the client area section of the **Insert ELS-Row** window will make the corresponding cell appear with outlined dotted guidelines, as shown in Figure 1.38.

Note that in the **Insert ELS-Row** window, you can resize inserted elements, but this is optional, since you can always resize *ELS-Lines* after inserting the *ELS-Row* into the **Design** view. Therefore, we are ready to click the **OK** button of the **Insert ELS-Row** window to insert the *ELS-Row* into the report script. After doing so, we observe that since the inserted *ELS-Row* has transparent cell background and no borders, consequently it may be the case that it is invisible in the **Design** view. In such cases, to make *ELS-Row* visible, click on the **Visible Borders** toggle button of the *Report Designer*'s main toolbar.

Following the outlined method of *ELS-Row* insertion, insert another *ELS-Row* that contains only one *ELS-Line* with 7 columns, 100% width and no borders.

Copyright © 2003-2004 by Epsilon-Logic Systems

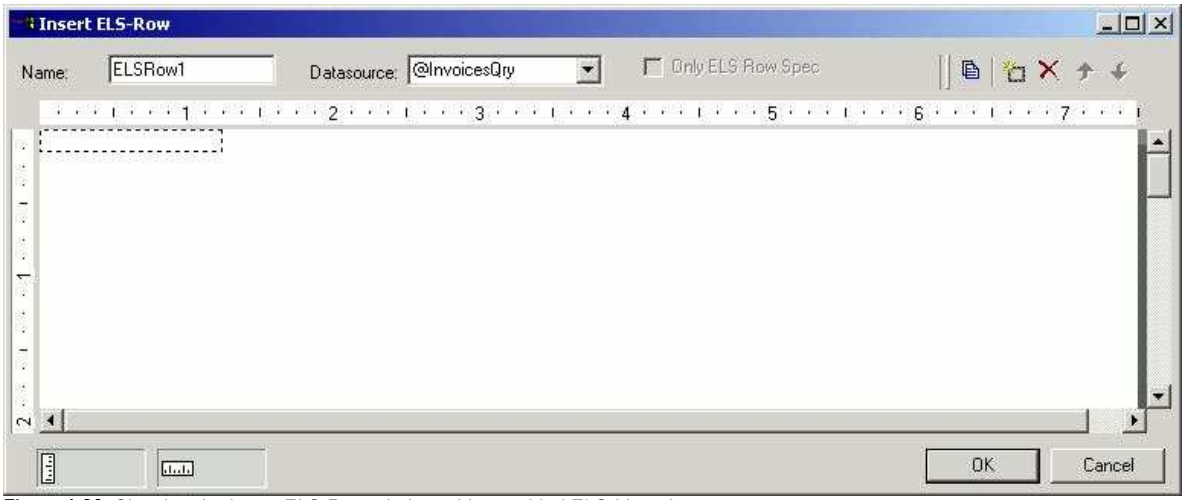


Figure 1.38. Showing the Insert ELS-Row window with an added ELS-Line element

Inserting Data Fields and Column Headers

Although we could insert fields into the report script without the use of *ELS-Row*, but there are a few advantages when using *ELS-Row* elements:

- First, the *ELS-Row* elements serve as guidelines to the inserted field elements, for example borders, background color, font type, as well as other properties may be supplied via the *ELS-Row* elements.
- Using *ELS-Row* elements will produce much better report output, in terms of tabulation and alignment. For example, the output produced via *ELS-Row* when copy-pasted into Microsoft© Excel will preserve its tabular structure.
- Finally, it is much easier to create reports with running-column-headers when using *ELS-Row* elements.

Therefore, now that we have already inserted an *ELS-Row* into the report's detail section, we will proceed next to define column headers and insert selected fields from the @InvoicesQry data source. But first, we resize and arrange the cells of the *ELS-Line* elements to suit the needs of the current report. In particular, the first *ELS-Line* must be used to display multi-lined *Ship-To* and *Bill-To* addresses. The second *ELS-Line* will be used simply to separate the address part of the invoice from the parent row, while the *ELS-Line* of the second *ELS-Row* will represent the parent row of each invoice. Of course, we will need yet another *ELS-Row* element to handle the child rows of each invoice.

In the **Design** view, perform the following steps:

- Resize the height of the first *ELS-Line* to make space for address lines.
- In the second column enter the text "`Ship To:`", and in the fifth column enter the text "`Bill To:`".
- Select the text in these columns separately, and using the **Font**, **Font Size**, **Bold** and **Fore Color** controls of the main toolbar, change the typeface to Arial 10pt Bold, with STEELBLUE fore-color.
- Finally, resize the columns of this *ELS-Line* to look like Figure 1.39 below.

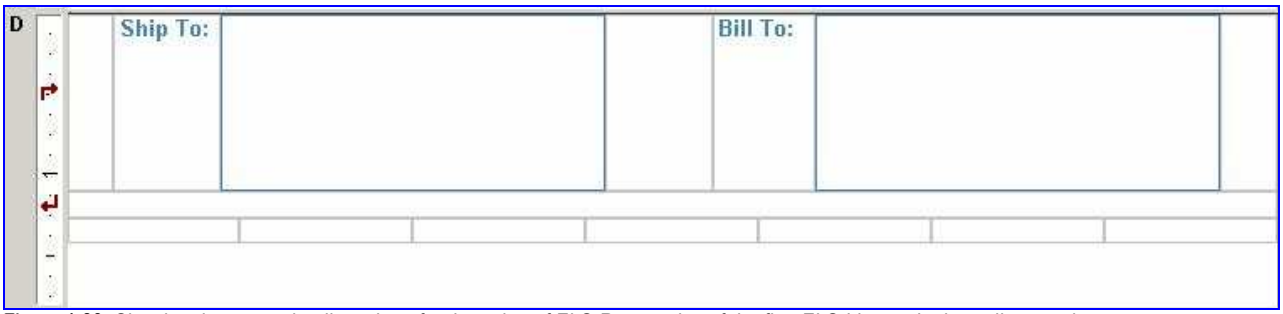


Figure 1.39. Showing the report detail section after insertion of ELS-Row, resize of the first ELS-Line and other edit operations

Now, we will apply the drop-down toolbar buttons of the **Borders** toolbar, shown in Figure 1.40, to define borders

Copyright © 2003-2004 by Epsilon-Logic Systems

for the cells next to "`Ship To:`" and "`Bill To:`" cells.

Figure 1.40. Showing the Borders toolbar

So put the cursor in one of these cells, and apply the border controls in the following sequence:

- First, drop-down the **Border** button, and select *all borders* item, note that this item is the leftmost button in the bottom row of the drop-down array.
- This being selected, drop-down the **Border Type** button and select the first item after the **No Borders** item in the drop-down list.
- Then apply the **Border Weight** button, selecting ½ pt weight for the borders.
- Finally, apply the **Border Color** button, selecting STEELBLUE color via the **More** item of the drop-down list and the **Color** dialog.

We are now ready to insert the data fields related to the *Ship-To* and *Bill-To* addresses of the invoices. So position the cursor in the *Ship-To* column (incidentally, this is the third column from the left), and select the **Data Fields** menu item from the **Insert** menu. This will display the **Data Fields** floating window showing all the data sources that are currently defined inside the report script. In this case, we only defined the InvoicesQry data source, so we will see only the @InvoicesQry node. Now expand this node by clicking on the (+)-sign icon, as a result, all the fields of the data source will be displayed in the **Data Fields** window, as depicted in Figure 1.41 below.

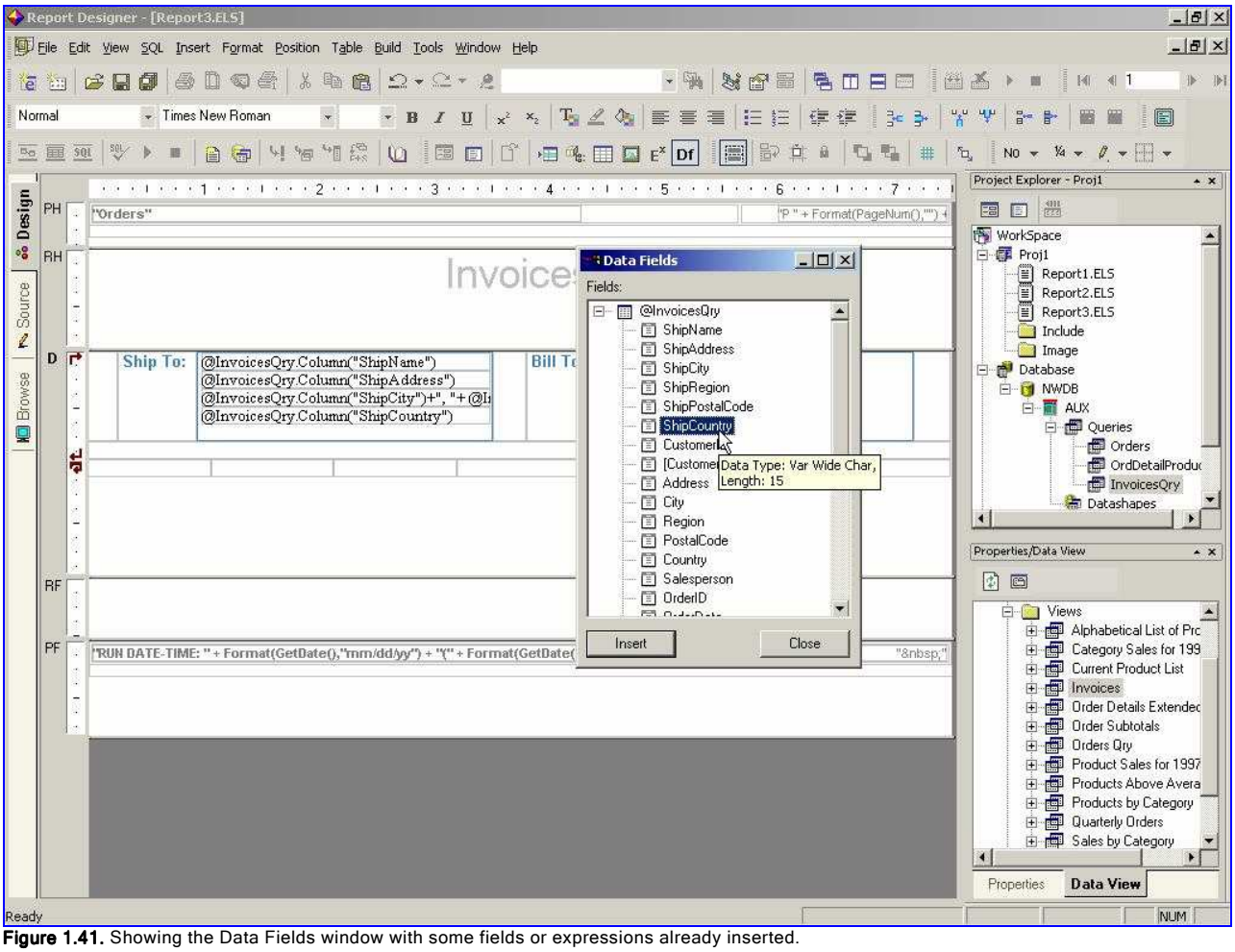


Figure 1.41. Showing the Data Fields window with some fields or expressions already inserted.

Next, we will utilize the **Data Fields** and **Expression Builder** windows to insert the *Ship-To* and *Bill-To* address information. In particular, for *Ship-To* address information, we will insert the *ShipName*, *ShipAddress*, *ShipCity*, *ShipRegion*, *ShipPostalCode* and *ShipCountry* fields via the following steps:

Copyright © 2003-2004 by Epsilon-Logic Systems

- Position the cursor in the empty cell next to the "`Ship To:`" label cell.
- Then drag-drop the *ShipName* field from the **Data Fields** window into this cell.
- Insert a line-break via the **Break** menu item under the **Insert** menu, or equivalently make sure that the cursor is in the cell and click the *Enter*-key.
- Repeat this scenario to insert the *ShipAddress* field in this cell (below the *ShipName* line).
- Close the **Data Fields** window, and call the **Expression Builder** window via the **Expression Builder** menu item under the **Insert** menu (see Figure 1.41 for more details).
- In the **Fields/Variables** pane of the **Expression Builder**, expand the @InvoicesQry node to see all the fields of the *Invoices* view object.
- Then insert the *ShipCity* field into the editor pane by double-clicking this field in the **Fields/Variables** pane.
- Put the cursor in the editor pane of the **Expression Builder**, and manually enter the text: `+ ", " +`
- Then, in this same way insert the *ShipRegion* field followed with the following text: `+ " " +`
- Finally, insert the *ShipPostalCode* field to complete the desired expression.
- Click the **Insert** button of the **Expression Builder** window to insert this expression into the *Ship-To* address information cell under the *ShipAddress* field's line.
- Now, close the **Expression Builder** window and call the **Data Fields** window again, to insert the *ShipCountry* field. Make sure you insert a line-break before inserting this last field in this *Ship-To* cell.

Incidentally, the **Expression Builder** window is a very handy tool to build SCRIPT expressions. We will outline more details about this tool in the later sections of this chapter.

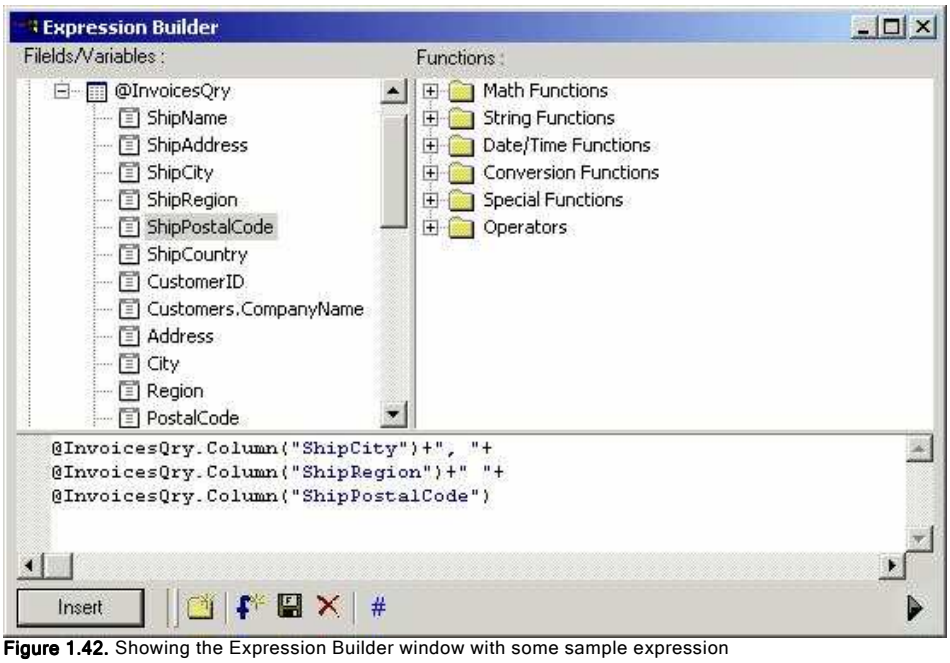


Figure 1.42. Showing the Expression Builder window with some sample expression

Now, perform these same steps for the *Bill-To* address information cell, inserting the fields *CompanyName*, *Address*, *City*, *Region*, *PostalCode* and *Country*. This completes the address part of the invoice. We will continue next, with the insertion of fields for the parent row of the invoice, as well as, setting the back color, typeface, alignment and the formatting of the parent row columns.

To proceed with the preparation of the invoice parent row, first call the **Data Fields** window, and using methods similar to that of the previous paragraphs, insert the following fields into the second *ELS-Row*: *OrderID*, *CustomerID*, *Salesperson*, *OrderDate*, *RequiredDate*, *ShippedDate* and *Shippers.CompanyName*. After insertion you may want to resize these field cells, so that some columns are not clipped in the report output. You may resize cells and fields directly in the **Design** view, or you may want to use the **Precision Resize** dialog, to resize objects with much better precision. For example, make sure the cursor is in a cell of the second *ELS-Row*, and from the **Tools** menu select the **Precision Resizer** menu item. In the **Precision Resize** dialog select a **Zoom** of 400% to enlarge the *ELS-Row* and field objects four times the original size, as shown in Figure 1.44.

The magnification of the *ELS-Row* should direct our attention to the observation that an *ELS-Row* consists of a

Copyright © 2003-2004 by Epsilon-Logic Systems

tabular object comprising of several cells, very similar to HTML tables. And that these cells may contain field objects, which themselves have borders, as depicted by the diagram in Figure 1.43 below.

Figure 1.43. Showing the boundaries of fields contained in ELS-Row cells

The cells of the tabulation object are intended to be used as borders and serve as guidelines for alignment of the contents, both horizontally and vertically.

When resizing elements in an *ELS-Row*, the user must consider resize operations of both, the cell boundaries, as well as the field objects contained inside these cells. In the current sample report, we want to resize the first three fields, to make room for the *Salesperson* field's cell. To resize the field object in the first cell, in the **Precision Resize** dialog click on the field object to activate the resize-guide controls. Then using the mouse, resize the field object. Then use the mouse again to resize or adjust the cell width. Perform similar methods to resize the other fields and cells of the *ELS-Row* element. We should emphasize that when using the **Precision Resize** dialog, you may use both the magnified view of the selected *ELS-Row* and the original size in the **Design** view. Moreover, all changes in the **Precision Resize** dialog may be updated in the **Design** view by simply clicking the **Apply** button of the **Precision Resize** dialog.

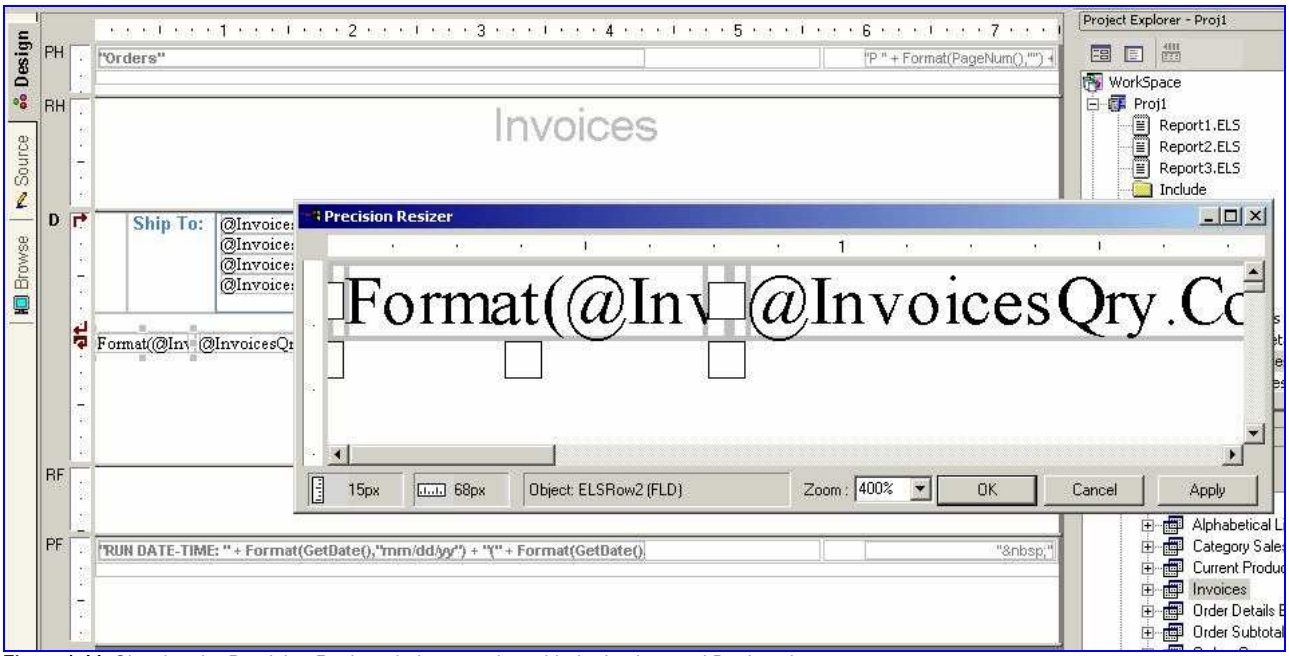


Figure 1.44. Showing the Precision Resize window together with the background Design view

Next, we will insert a column header row for the invoice parent row. This may be performed via the following steps:

- Position the cursor in any cell of the *ELS-Row* corresponding to the invoice parent row.
- Then from the **Table** menu select the **Insert Column Headers** menu item.
- This will insert a special empty row above the selected *ELS-Row*, with the same cell structure as the *ELS-Row*.
- In this column headers row you must include all the header labels associated with the selected *ELS-Row*. For example, for the column header cells, from left to right, enter the following text: "`Order ID`", "`Customer ID`", "`Sales Person`", "`Order Date`", "`Required Date`", "`Shipped Date`" and "`Ship Via`".
- In parallel to this entry, for each cell apply the **Back Color** toolbar button. This will drop-down a menu of colors and the **More** menu item. Select this later item to call the **Color** dialog, and in this dialog select the STEELBLUE color.
- Then, select the text of all these seven cells and apply the **Font**, **Bold** and **Fore Color** commands, setting the typeface of these cells to Arial 10 bold white.

Copyright © 2003-2004 by Epsilon-Logic Systems

The result of these operations is shown in Figure 1.45 below.

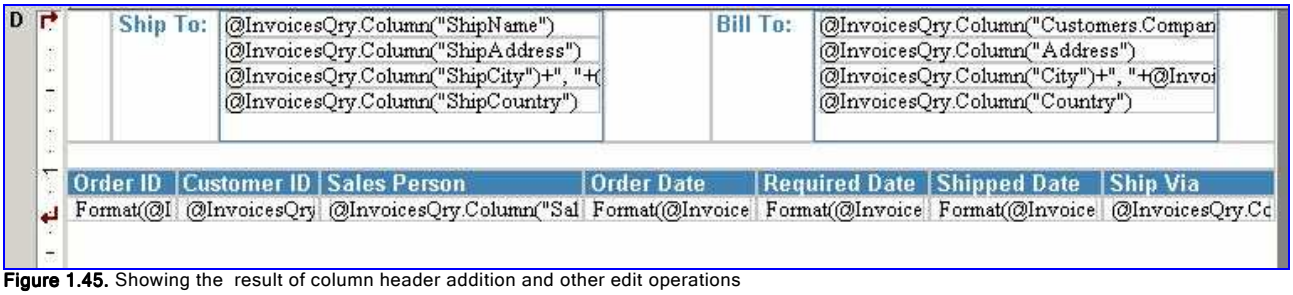


Figure 1.45. Showing the result of column header addition and other edit operations

Finally, we will apply data formatting on the *OrderDate*, *RequiredDate* and *ShippedDate* fields, to format them as medium dates. To proceed, click on the *OrderDate* field object to select it, and then in the **Properties/Data View** pane click on the **Properties** tab. In this **Properties** view, click on the **Categorized** toolbar button, to display the properties of the selected field object in categorized mode. Observe that the <FLD>-tag element is selected in the top combo-box control, indicating that the selected object in the **Design** view is a field object.

In the **Properties** pane (see Figure 1.46 for details), find the **Format** group, and click on the edit-control's button located at right-side edge of the Specification property cell. This will call the **Format/Conversion Wizard** dialog.

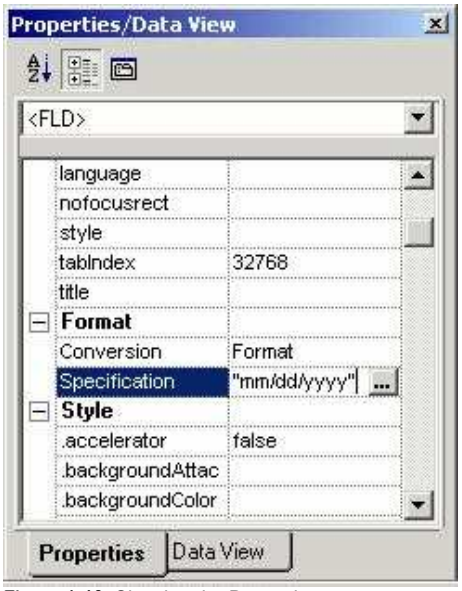


Figure 1.46. Showing the Properties pane

In this later dialog, make sure that the **Format** radio-option is selected in the **Conversion Function** frame, and the **Expression Type** is set to Date/Time. Then select Medium Date item from the **Format** combo-box and submit the dialog to update the format of the selected field.
Repeat this sequence of operations for the *RequiredDate* and *ShippedDate* fields.

We will describe more of the features and details of the **Properties** pane in later paragraphs of this chapter, but for now, it is worth to note that almost anything in the **Design** view may be edited via the **Properties** pane. Moreover, these edit operations are much more precise than their respective visual counterparts in the **Design** view. Therefore, together with the visual *HTML* and *ELS-Row* edit operations, editing via the **Properties** pane supplements the capabilities and features of the *Report Designer* application.

This completes the parent row preparation of the invoices. We will insert next, yet another *ELS-Row* element, which will handle the details row of the invoices.

Details Row and Loop Construct

Continuing the development of the Invoices report script, we will insert another *ELS-Row* element in the report script, essentially to handle the invoice details row. Before we do this, we would like to inform the reader that the insertion of *ELS-Row*, *ELS-Line* and field elements may be performed in **Source** view as well as the **Design** view. In fact, sometimes it is much easier to properly insert objects in the **Source** view than in the **Design** view.

Therefore, to switch to the proper location in the **Source** view, double-click on the red loop arrows located inside the vertical ruler of the **Design** view (see Figure 1.45 for more details). This action will switch the view to **Source** view, and scroll to the location where the *WHILE*-loop is defined. Now put the cursor between the `</ELS_ROW>` end-tag and the `<ELS>` begin-tag, and call the **Insert ELS-Row** window via the **ELS Row** menu item of the **Insert** menu. In this window click on the **Add** toolbar button (this is the second toolbar button from the left at the top-right corner of the **Insert ELS-Row** window) to call the **Insert ELS-Line** dialog. In the **Insert ELS-Line** dialog, set the width to 100%, the columns to 6, and border size to 0, then click the **OK** button. This will insert an *ELS-Line* in the **Insert ELS-Row** client-area. Now submit the **Insert ELS-Row** window by clicking the **OK** button.

Copyright © 2003-2004 by Epsilon-Logic Systems

This will insert the following code segment in the current cursor location of the **Source** view:

```
<ELS_ROW NAME="ELSRow3">
    <L border="0" CellSpacing="0" CellPadding="0" WIDTH="100%" HEIGHT="15">
            <C WIDTH="16.67%" HEIGHT="15">
                     
            </C>
            <C WIDTH="16.67%" HEIGHT="15">
                     
            </C>
            <C WIDTH="16.67%" HEIGHT="15">
                     
            </C>
            <C WIDTH="16.67%" HEIGHT="15">
                     
            </C>
            <C WIDTH="16.67%" HEIGHT="15">
                     
            </C>
            <C WIDTH="16.65%" HEIGHT="15">
                     
            </C>
    </L>
</ELS_ROW>
```

At this point, since we are in the **Source** view, it is a good idea to examine how the *Report Designer* generated the underlying SCRIPT code of the report. In particular, check the three *ELS-Row* elements to see how the SCRIPT language identifies such elements. For example, the `NAME` attribute has the values "`ELSRow1`", "`ELSRow2`" and "`ELSRow3`" respectively for the address information, invoice parent row and invoice details row elements.
The next observation that you should make is the code for the loop construct, namely the following code segment:

```
<ELS>
    ResultRow("ELSRow1");
    BeginHeader("ELSRow2");
    WHILE NOT @InvoicesQry.Eof()
            ResultRow("ELSRow2");
            @InvoicesQry.Next();
    END LOOP
    EndHeader("ELSRow2");
</ELS>
```

We need to modify this code to fit the parent-child row structure in the current Invoices report. In fact, for every new *OrderID* we want to display a set of *Ship-To* and *Bill-To* addresses together with the invoice parent row. Moreover, for each *OrderID* we need to display all products under the corresponding order invoice.
The following code segment illustrates with comments all the necessary changes, making the loop construct more suitable to the current report example:

```
<ELS>
// declare an integer variable to store the old OrderID value
DECLARE @OrdID int;
// initialize this variable
SET @OrdID = -1;

// loop over the invoices data source
WHILE NOT @InvoicesQry.Eof()
    // check if the OrderID is distinct from the previous value
    IF @InvoicesQry.Column("OrderID") <> @OrdID THEN
            // end the column header scopes of ELSRow3 and ELSRow2
            EndHeader("ELSRow3");
            EndHeader("ELSRow2");
            ResultRow("ELSRow1");
            // begin the column header scope for the parent row
            BeginHeader("ELSRow2");
            // write the invoice parent row
            ResultRow("ELSRow2");
            // begin the column header scope for the details row
```

Copyright © 2003-2004 by Epsilon-Logic Systems

```
            BeginHeader("ELSRow3");
            // store the current OrderID value for later comparison
            SET @OrdID = @InvoicesQry.Column("OrderID");
    END IF
    // write the invoice details row
    ResultRow("ELSRow3");
    // fetch the next record of the invoices data source
    @InvoicesQry.Next();
END LOOP
// end the column header scopes of ELSRow3 and ELSRow2 for the last record
EndHeader("ELSRow3");
EndHeader("ELSRow2");
</ELS>
```

Now, switch the view to **Design** view and insert the following fields via the **Data Fields** window: *ProductID*, *ProductName*, *Quantity*, *UnitPrice*, *Discount* and *ExtendedPrice*. After these field insertions, resize the fields and the columns of the details row *ELS-Row*, to make enough space for the *ProductName* field. Also, you may want to increase the height of the *ELS-Row* itself. Recall that this resize operation may be performed either in the **Design** view directly, or via the **Precision Resize** window.

Once the resize operations for the invoice details row is finalized, we are ready to insert a column header row above this details row. Put the cursor anywhere inside the invoice details row *ELS-Row* element, and from the **Table** menu select the **Insert Column Headers** menu item. This will insert a new *ELS-Line* specialized for column headers. Observe that the column widths have the same structure as the original field row columns from which this column headers row was made.

Now enter the following text "`ProductID`", "`Product Name`", "`Quantity`", "`Unit Price`", "`Discount`" and "`Ext-Price`" respectively for the headers of the fields *ProductID*, *ProductName*, *Quantity*, *UnitPrice*, *Discount* and *ExtendedPrice*.

For each of these header cells we will set the back color to LIGHTSTEELBLUE by performing the following. First put the cursor inside the cell, then using the **Back Color** drop-down toolbar button, select the **More** item to call the **Color** dialog. In the **Color** dialog select the LIGHTSTEELBLUE color and submit the dialog. Repeat these steps for the other column header cells as well.
Also, select the text of the entire column headers row, by pushing down the left mouse button and moving the mouse across the end of the column headers row. Then using the **Fore Color** drop-down toolbar button, select the **More** item to call the **Color** dialog. In the **Color** dialog select the DARKBLUE color and submit the dialog. This will set the color of the column headers text. Moreover, keeping this entire row text selected, click on the **Italic** toggle button of the main HTML formatting toolbar. This will make the selected text italic.

Next, we want to right-align the columns containing the *Quantity*, *UnitPrice*, *Discount* and *ExtendedPrice* fields. To proceed, for example, put the cursor in the column that contains the *Quantity* field, and then click on the **Align Right** toggle button of the main HTML format toolbar. Similarly, put the cursor in the column header of the *Quantity* field and click on the **Align Right** toggle button. You may repeat this same procedure to make all the other fields right-aligned.

Next, we will apply data formatting on the numeric fields *Quantity*, *UnitPrice*, *Discount* and *ExtendedPrice*. For example, click on the *Quantity* field object to select it (you should see resize control pointers around the field object). Then in the **Properties/Data View** pane make sure that the **Properties** view is active. Also, you may click on the **Categorize** toggle button in the toolbar of the view (see Figure 1.46 for more details). In the **Properties** pane (see Figure 1.46), find the **Format** group, and click on the edit-control's button located at right-side edge of the Specification property cell. This will call the **Format/Conversion Wizard** dialog. In this dialog, select the text of the **Format** comb-box and enter the following format specification "`#,##0`", and then click the *Enter*-key to update the selection. Then click the **OK** button to submit the dialog. Repeat these steps for the *UnitPrice*, *Discount* and *ExtendedPrice* fields using format specification "`$#,##0.00`". Note that for numeric fields, the format specification character "`#`" stands for a number digit or empty when the value is null, whereas the character "`0`" stands for a number digit or 0 when the value is null.

The next task is to make some space before the *Ship-To* and *Bill-To* address section, so switch the view of the *Report Designer* window to **Source** view, and scroll to the location where the ELSRow1 *ELS-Row* element begins (i.e. look for the `<ELS_ROW NAME="ELSRow1">` tag). Now, put the cursor after the `<ELS_ROW NAME="ELSRow1">` tag

Copyright © 2003-2004 by Epsilon-Logic Systems

and click the *Enter*-key to make some space. Then select the **Insert ELS Line** menu item from the **Table** menu to call the **Insert ELS-Line** dialog. In this dialog, set the width to 100%, columns to 1 and border size to 0, and then click the **OK** button to insert a blank *ELS-Line* element at the current cursor location.

Since we are in the **Source** view mode, check the code for the loop construct again to see if it has not changed. In particular, note that the column headers row insertion has automatically added redundant *BeginHeader* and *EndHeader* function calls for the ELSRow3 *ELS-Row* element. Namely, the following line:

```
BeginHeader("ELSRow3");
```

before the *WHILE*-loop, and the corresponding end function

```
EndHeader("ELSRow3");
```

after the *END LOOP* line. Remove these extra lines so that no header duplication occurs, and the loop construct code becomes exactly as it was before the invoice details row's column headers row insertion.

We will now define some borders for the new blank row, as well as the details row. First, switch back to the **Design** view, and put the cursor inside the new *ELS-Line* element at the top of the **D** section. Then from the **Border** drop-down button select the Top edge item (incidentally, this is the second item in the bottom row). From the **Border Type** drop-down button select the Double item, from the **Border Weight** button select 3 pts, and finally from the **Border Color** drop-down button select the STEELBLUE color.
Similarly, for each cell of the invoice details row *ELS-Row* element, put the cursor inside the cell and apply the **Borders** toolbar function, selecting the **Border** to be Bottom, **Border Type** Solid, the **Border Weight** ½ pt, and the **Border Color** LIGHTSTEELBLUE.

This completes the report's **D** section with the exception of the summary totals. For report summaries, we will add summary totals of the *ExtendedPrice* field in two levels: per invoice total for each distinct *OrderID*, and a grand total at the end of the report output.

First, switch to the **Source** view, and scroll to the location after the `</ELS>` end-tag of the loop construct code segment. And put the cursor between this end-tag and the `</ELS_RDETAIL>` end-tag (click the *Enter*-key couple of times to make some space). Now, using the **Insert ELS Row** toolbar button call the **Insert ELS-Row** window one more time. In this later window call the **Insert ELS-Line** dialog to insert an *ELS-Line* with width 100%, columns 2 and border size 0. Then submit the **Insert ELS-Row** window by clicking the **OK** button. This will insert the following code segment at the current cursor location:

```
<ELS_ROW NAME="ELSRow4">
    <L border="0" CellSpacing="0" CellPadding="0" WIDTH="100%" HEIGHT="15">
            <C WIDTH="50.00%" HEIGHT="15">
                     
            </C>
            <C WIDTH="50.00%" HEIGHT="15">
                     
            </C>
    </L>
</ELS_ROW>
```

Now, we will modify the loop construct code segment to handle the iteration of the summary totals. In particular, the new code should be as follows:

```
<ELS>
// declare an integer variable to store the old OrderID value
DECLARE @OrdID int;
// declare summary variables
DECLARE @InvTotal float;
DECLARE @GrandTotal float;

// initialize these variable
SET @OrdID = -1;
SET @InvTotal = 0;
SET @GrandTotal = 0;
```

Copyright © 2003-2004 by Epsilon-Logic Systems

```
// loop over the invoices data source
WHILE NOT @InvoicesQry.Eof()
    // check if the OrderID is distinct from the previous value
    IF @InvoicesQry.Column("OrderID") <> @OrdID THEN
            // end the column header scopes of ELSRow3 and ELSRow2
            EndHeader("ELSRow3");
            // if it's not the first record, write the invoice totals
            IF @OrdID <> -1 THEN
                    ResultRow("ELSRow4");
                    SET @InvTotal = 0;
            END IF
            EndHeader("ELSRow2");
            ResultRow("ELSRow1");
            // begin the column header scope for the parent row
            BeginHeader("ELSRow2");
            // write the invoice parent row
            ResultRow("ELSRow2");
            // begin the column header scope for the details row
            BeginHeader("ELSRow3");
            // store the current OrderID value for later comparison
            SET @OrdID = @InvoicesQry.Column("OrderID");
    END IF
    // write the invoice details row
    ResultRow("ELSRow3");
    // increment summary totals
    SET @InvTotal = @InvTotal + @InvoicesQry.Column("ExtendedPrice");
    SET @GrandTotal = @GrandTotal + @InvoicesQry.Column("ExtendedPrice");
    // fetch the next record of the invoices data source
    @InvoicesQry.Next();
END LOOP
// end the column header scopes of ELSRow3 and ELSRow2 for the last record
EndHeader("ELSRow3");
EndHeader("ELSRow2");
</ELS>
```

At this point, we may switch back to the **Design** view to edit this new *ELS-Row*. In **Design** view, to make this new row visible, apply the right-mouse button on the blue arrow in the horizontal ruler. This will display a popup window showing a tree-view of conditional statements (see Figure 1.47 for more details):

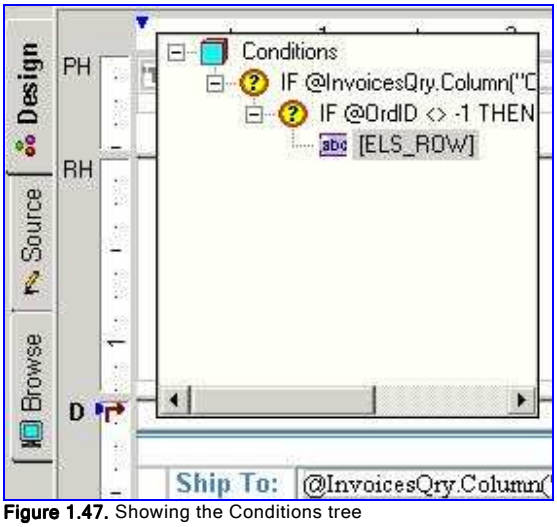


Figure 1.47. Showing the Conditions tree

Double-click on the [ELS_ROW] node to display the new *ELS-Row* defined under conditionals. The row will be displayed in the **Design** view positioned at the top of the **D** section.

In particular, this *ELS-Row* has only two cells, one to show the label text and the other for the invoice total.

Resize the first column to make it almost 75% of the width of the *ELS-Row*. Then put the cursor in the first cell and enter the text "`INVOICE TOTAL:`". Select this text and apply **Bold** toggle button, then **Align Right**, and then set the color to STEELBLUE.

For the second cell, put the cursor inside the cell and call the **Expression Builder** window. In this later window, expand the **Other Variables** folder in the **Fields/Variables** pane. And double-click on the @InvTotal variable to insert it in the editor pane.

Then in the editor pane select this variable's name and click on the **Format/Conversion Wizard** toolbar button at the bottom of the **Expression Builder** window (i.e. the button with the #-icon). In the **Format/Conversion Wizard** dialog, set the **Format** combo-box value to "`$#,##0.00`" by direct keyboard entry, and submit the dialog. This will insert the following text in the **Expression Builder** window's editor pane:

```
Format(@InvTotal, "$#,##0.00")
```

Click on the **Insert** button to insert this expression in the second cell of the summary row.

Copyright © 2003-2004 by Epsilon-Logic Systems

Now, for grand total summary, switch back to **Source** view and scroll to the position where the invoice summary *ELS-Row* is positioned. Copy the code lines from the `<ELS_ROW NAME="ELSRow4">` to the corresponding `</ELS_ROW>` end-tag, and then paste it below these lines and above the `</ELS_RDETAIL>` end-tag. Then change the `NAME` value to `"ELSRow5"`, and the "`INVOICE TOTAL:`" to "`GRAND TOTAL:`", and replace the *@InvTotal* variable by the *@GrandTotal* variable. So that the ELSRow5 *ELS-Row* element will have the following resulting code segment:

```
<ELS_ROW NAME="ELSRow5">
   <L border="0" CellSpacing="0" CellPadding="0" WIDTH="100%" HEIGHT="15"
                                                STYLE="HEIGHT: 16px" VALIGN="top">
      <C WIDTH="81.11%">
            <P align=right>
            <FONT style="COLOR: #4682b4; FONT-FAMILY: Arial">
            <B>GRAND TOTAL:</B></FONT></P>
      </C>
      <C WIDTH="18.89%"><P align=right >
            <B>   
            <SPAN class=Field style="FONT-WEIGHT: bold; OVERFLOW: hidden; WIDTH: 118px;
                  WHITE-SPACE: nowrap; HEIGHT: 15px" >
            <FLD>Format(@GrandTotal, "$#,##0.00")</FLD>
            </SPAN></P></B>
      </C>
   </L>
</ELS_ROW>
```

Finally, to display the grand total at the end of the report we must include a *ResultRow* function call in the report footer section of the report. The following code segment will essentially do just that:

```
<ELS_RFOOTER FONT-FAMILY="Times New Roman" FONT-SIZE="10pt">
<BR>
<ELS>
// write the final grand total row
ResultRow("ELSRow5");
</ELS>

</ELS_RFOOTER>
```

Finally, we will edit the report header section of the report, and along the way we will learn how to insert an image into the report script. Now, in the **RH** section of the report in the **Design** view, instead of the text "`Invoices`", enter the following text "`I N V O I C E S  R E P O R T`", include the space characters to expand the density of the text. Then select this text and modify the font size to 18 points, with font style bold, and font color to LIGHTSLATEGRAY. Then for each word, select the first character of the word and set the font size to 24 points.

Next, put the cursor on the next line, and using the **Picture** menu item of the **Insert** menu, call the **Insert Image** dialog as shown in Figure 1.48 below.

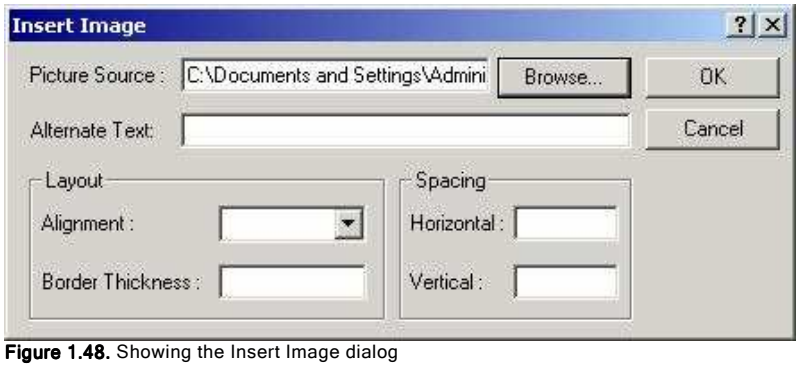


Figure 1.48. Showing the Insert Image dialog

Now select an image via the **Browse** button and submit the dialog. This will insert the image in the report header

Copyright © 2003-2004 by Epsilon-Logic Systems

section at the current cursor location. Click on this inserted image to select it, and then click on the **Absolute Position** toggle button in the main toolbar. This will make the image object to behave in absolute positioning mode, so that you may freely move the image to any absolute coordinates.

Compiling and Running the Report

Before compiling and running the report, you may want to take a look at the result in the **Browse** view. Note that the fields in the **Design** view appear as some sort of formulae instead of traditional dummy values. Consequently, if data formatting is used, then it is hard to follow the pattern of the format. For such purposes among other reasons, we have the **Browse** view. Click on the vertical **Browse** tab to see the dummy values of the fields in the **Browse** view (see Figure 1.49 for more details).

Figure 1.49. Showing the Browse view with dummy values in the fields

Now you are ready to compile and run the report, therefore, click on the **Source** view to make the **Compile** command enabled. From the **Build** menu select the **Compile** menu item, which will start the compilation of the report script into binary form. If there are syntax errors in your report script the compiler will find and report them in the **Results** pane below the **Output View** pane. To get to the line in the report script where the error occurred, you may double-click on the error line in the **Results** pane.

After successful compilation of the report, you are ready to run it by selecting the **Run** menu item under the **Build** menu. This will generate the report output and display the first page in the **Output View** pane. To hide the **Results** pane, you may click on the **Show Results Pane** toggle toolbar button in the main toolbar of the *Report Designer*.

To navigate through the report output pages, you may use the **First Page**, **Previous Page**, **Next Page** and **Last Page** menu items of the **Build** menu, or the equivalent toolbar buttons. The result of the report output for the current example report is shown in Figure 1.50.

To compile or run the report again, you must switch from the **Output View** tab to the **Report1.ELS** tab at the bottom of the main window. Also, note that when you click the **Run** button (or corresponding menu command) the first time after any update to the report script will automatically re-compile the report before generating the report output.

In conclusion to the current section, note that we have outlined the usage of almost all the tools that exist in the *Report Designer* application. The next few sections will cover some issues related to the parameterization of reports, auto-alignment of ELS-Rows, as well as the creation of XML based reports.

Copyright © 2003-2004 by Epsilon-Logic Systems

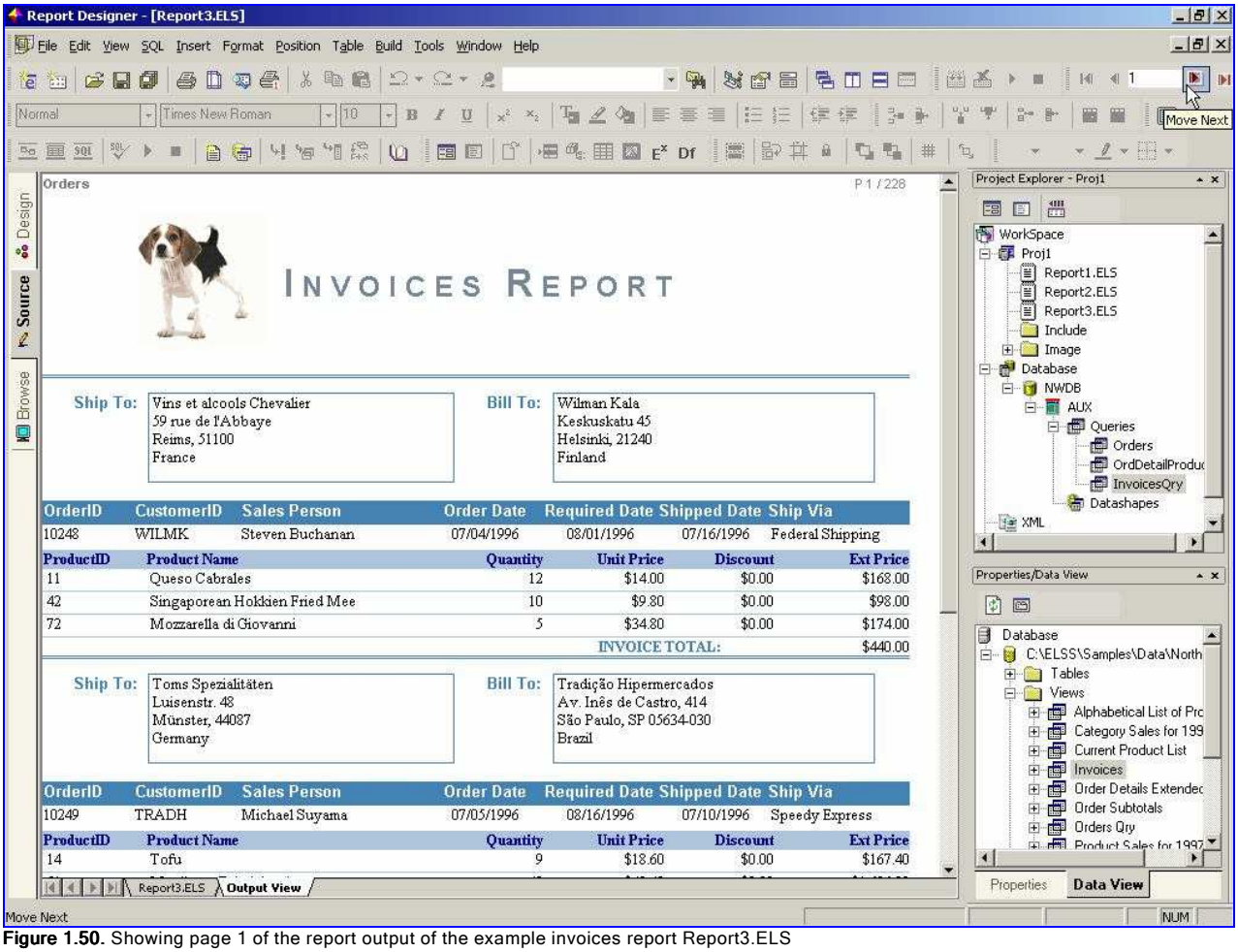


Figure 1.50. Showing page 1 of the report output of the example invoices report Report3.ELS

Creating XML-based Reports via SRT Wizards

In this section we will utilize some *SRT Wizard* templates to generate report scripts for reports that are based on XML data. In particular, we will assume that we have two separate XML documents, namely *Orders.XML* and *OrderDetails.XML*. The first XML document contains all parent records of orders, while the second XML document contains all the child records of the orders. Moreover, we also assume that the tag-element *OrderID* is common between these two XML documents and therefore may provide a means to link these two otherwise separate XML document files.

For the first XML report sample we will use the *Orders.XML* data file. To proceed we will assume that the user has the *Report Designer* application running and the project Proj1 is open in the **Project Explorer** pane. Call the **New Report** dialog from the **New Report** menu command of the **File** menu, and using the suggested Report4 name in the **Report Name** control and the **Use Standard Template Wizard** option, submit this **New Report** dialog. This will call the **Select Standard Template** window, in which you must look for the **XML Based** group and select the Simple XML Report option. After this window is submitted, the *SRT Wizard* window will be displayed with the Simple XML Report template activated, as depicted by the title bar: **Simple XML Report SRT Wizard**.

This *SRT Wizard* template consists of the following standard question groups:

- Page Setup Information
- Report Header Information
- Report Detail Information
- Report Summary/Footer Information
- Horizontal/Vertical Borders
- Alternating Rows

Copyright © 2003-2004 by Epsilon-Logic Systems

Suggested answers for questions in **Page Setup Information** and **Report Header Information** groups are listed in the table below:

Question Node	Suggested or Default Value
Report Title	`Orders`
Page Orientation	`Portrait`
Page Size	`Letter 8 ½ x 11 in`
Top Margin	`0.75`
Left Margin	`0.50`
Right Margin	`0.50`
Bottom Margin	`0.50`
Page Header Left Text	`Orders`
Page Header Right Text	`P &Page / &PageCount`
Page Footer Left Text	`RUN DATE-TIME: &RunDate (&RunTime)`
Page Footer Right Text	` `
Report Heading	`Simple XML Report: Orders`
Report Heading Font	`Arial 26pt SILVER`

In the **Report Detail Information** question group, select the first question, namely **Define XML Source**, and in the input controls section click on the **Browse** button. This will prompt the **Select File** windows dialog, in which you need to select an XML data file. Find and select the *Orders.XML* file from the \ELSS\Samples\Data subdirectory.

Next, switch to **Define XML Dataset** question and click on the **Select** button of this question. This will prompt the **XML Dataset Selector** window, in which you must select a tag to represent the dataset pattern. In particular, for the current case, the **XML Dataset Selector** window shows a tree of nodes with the root node being "`dataroot`", which has a child node with the label "`Orders`". In turn the child node itself has children that resemble the field names of the orders dataset. Therefore, in this case it is quite obvious that the dataset pattern is defined by the "`Orders`" tag. To select this "`Orders`" tag for dataset, the user must double-click the node. This will select the node and close the **XML Dataset Selector** window, automatically entering the "`dataroot/Orders`" pattern in the **XML Dataset** control of the input controls for the question.

The next question is the **Define XML Parameter Fields**, which is very similar to the corresponding RDBMS parameter field definition. Therefore, select this question and click on the **Select Fields** button in the input controls section of this question. This will prompt the **Select XML Parameter Fields** dialog. In this dialog, the user must select a parameter field from the **All Fields** together with a corresponding operator in the **Operator** list-box, and then click the **Add** toolbar button. In this way adding several parameters in the AND/OR grid. For example, select the "`OrderID`" field in the **All Fields** and "`>=`" operator in the **Operator** list-box, and then click the **Add** button to put the string "`OrderID >=`" in the first column of the AND/OR grid. Submitting the **Select XML Parameter Fields** dialog will insert the selected parameter and operator combination string in the **Parameter Fields** list-box of the input controls of the current question.

The **Define XML Column Fields** question is also very similar to the corresponding question in the case of the RDBMS data source. In particular, this question will define the actual columns of the data presentation in the detail section of the report. To proceed, click on the **Select Fields** button in the input controls section of this question. This will display the **Select XML Fields** window, in which the user may define the columns of the row, together with text format, color, background color, alignment, as well as the actual widths of each column. For example, select the following fields by double-clicking the field name in the fields list: *OrderID*, *CurtomerID*, *OrderDate*, *ShipName* and *ShipCity*. Also, you may set a desired background color for the *HDR*-row in the HDR/FLD grid of the window, and resize each column to an appropriate width. Submitting this window will insert the selected fields' names into the **Selected Fields** list-box in the input controls section of the question.

Finally, the **Define XML Summary Fields** question does not apply in the current example and therefore may be skipped. The other remaining questions are very similar to the corresponding questions in the previous samples, and therefore will be left to the user to decide. For the **Alternating Rows** question, you must select the **Alternate Rows** check-box if you want alternating rows in the report. Then you must select a background color for the alternating row via the **Alternate Row Color** question.

Now you are ready to generate the report script. To proceed, click on the **OK** button of the *SRT Wizard* window to generate the script. To run the report, make sure the **Source** view is active and then click on the **Run** toolbar

Copyright © 2003-2004 by Epsilon-Logic Systems

button.

In the rest of this section we will briefly outline yet another *SRT Wizard* template that utilizes two separate XML data files to form a single report obtained by relating a common tag between the two XML structures. These XML data files are *Orders.XML* and *OrderDetails.XML*, located in the \ELSS\Samples\Data subdirectory. To view the content of these XML data files directly inside the *Report Designer* application, from the **Project Explorer** pane make sure the **XML** container node is active (i.e. selected), then use the right-mouse-button method to call the popup menu. From the popup menu select the **New** command. This will prompt the **New XML Document** dialog, in which you must enter a user-friendly name in the **Name** control and then browse and select the actual XML file via the 3-dotted button next to the **Path** control. Submitting this dialog will insert a child node under the **XML** container node with a label name defined by the **Name** control. To open the XML document, all you will need to do is to double-click on the XML document node. This will display the content of the XML document file in an IE browser that is internal to the *Report Designer* application. As an example, it is advisable to create the "Orders" and "Order-Details" XML nodes, respectively pointing to the *Orders.XML* and *OrderDetails.XML* files.

Let us now proceed and call upon the **New Report** dialog, and entering or using the sequential name Report5 submit the dialog. In the **Select Standard Template** look under the **XML Based** group and select the **Hierarchical XML Report** template. Submitting the **Select Standard Template** window will display the **Hierachical XML Report SRT Wizard** window, which has the following standard question groups:

- Page Setup Information
- Report Header Information
- Report Detail Information
- Report Summary/Footer Information
- Field Alignment Information
- Horizontal/Vertical Borders

Most of the questions included under these groups are standard and very similar to the corresponding questions in the previous tutorials. Therefore, we leave all but the **Report Detail Information** questions to the user. To outline the **Report Detail Information** questions, we start with the selection of **Define Parent XML Source** question. In the input controls of the question, click on the **Browse** button to call the **Select File** dialog, and then select the *Orders.XML* file from the \ELSS\Samples\Data subdirectory. This will define the XML data source for the parent row of the report.

The next question, namely **Define Parent XML Dataset**, will define the dataset pattern for the parent row. For this question click the **Select** button, and in the **XML Dataset Selector** dialog, double-click on the "`Orders`" node to select it into the **XML Dataset** control of the question.
For the **Define Parent XML Parameter Fields** question, click the **Select Fields** button and in the **Select XML Parameter Fields** dialog, select the "`OrderID`" field item in the **All Fields** list-box, then selecting the ">=" operator of the **Operator** list-box, click on the **Add** toolbar button. This will select the string "`OrderID >=`" in the AND/OR grid.
Next, select the **Define Parent XML Column Fields** question, and in the input controls section click on the **Select Fields** button to call the **Select XML Fields** window. As usual, in the **Select XML Fields** window select the *OrderID*, *CustomerID*, *OrderDate*, *ShipVia*, *ShipName* and *ShipCity* fields. Then apply the appropriate typeface style, alignment, color and background color, as well as resize the columns to proper widths.

The next question is about the row offset of the child row in the report. For this question retain the default value of 100 pixels for the offset. For the **Define Child XML Source** question click the **Browse** button and from the **Select Files** dialog select the *OrderDetails.XML* file from the \ELSS\Samples\Data subdirectory. Then, for question **Define Child XML Dataset**, click the **Select** button to display the **XML Dataset Selector** dialog, and in this dialog double-click on the "`OrderDetails`" node to select it as a dataset pattern for the child row in the report. Since we do not want any parameters for the child row, we may skip the **Define Child XML Parameter Fields** question. For the next question, namely **Define Child XML Column Fields**, click on the **Select Fields** button to display the **Select XML Fields** window with the selected child XML dataset's field names shown in the fields list. From this list, select the following fields: *ProductName*, *Quantity*, *UnitPrice* and *SubTotal*, and then apply typeface style, color, and background color, as well as resize the columns to fit the page size.
The final question in the **Report Detail Information** group is the **Parent/Child Relationship**, which will relate the contents of the two separate XML data files, namely the *Orders.XML* and the *OrderDetails.XML*. In this question, click the **Relate Fields** button to call the **Define Relation** dialog. In this dialog, select the *OrderID* field item in

Copyright © 2003-2004 by Epsilon-Logic Systems

both of the combo-box controls **Parent Fields** and **Child Fields/Parameters**, and then click the **Add** button to add this relation into the list below. Submitting this dialog will insert the string "`OrderID TO OrderID`" into the **Relationship** control of the question. This finalizes the questions of the **Report Detail Information** group. We leave the rest of the *SRT Wizard* setup to the user.

Saving SRT Wizard's Setup for Reuse

So far we have been using *Standard Report Templates* via the *SRT Wizard* without indulging ourselves into much detail about how *SRT Wizard* templates actually work. In this section however, we will take one step further and present a short introduction on the *Standard Report Template meta-Language*, or *SRTmL* in short. We will also introduce a binary serialization of the setup information entered by the end-user when using the *SRT Wizard*.

The *SRT Wizard* is based on mechanisms that parse and render *SRTmL* script into the GUI of the *SRT Wizard*'s window. For example, the *SRT Wizards* that we have been using for the previous tutorials are based on *SRTmL* templates that are located in the \ELSS\Bin\TEMPLATES subdirectory. For example, the *Single Record Report.SRT* template is essentially a text file, which contains *SRTmL* instructions intertwined with SCRIPT code, as well as HTML and other script code. A *SRTmL* script consists of three parts identified with the following tags:

- `<SRT_VIEW> </SRT_VIEW>`, which defines the boundaries for an HTML page that resembles a sample page of the report output derived from the template. This page will be displayed in the **Select Standard Template** window when the current template is selected on the right side tree-view pane.
- `<SRT_WIZ> </SRT_WIZ>`, which defines the boundaries for all the question groups and respective questions for the template wizard's GUI. The instructions of this section will render the groups and questions of the *SRT Wizard* window, as well as the images that are displayed in the left-side pane.
- `<SRT_SCRIPT> </SRT_SCRIPT>`, which defines the boundaries for the SCRIPT code that will generate the report script based on the answers of the questions in the *SRT_WIZ* section.

For example, a group in *SRT_WIZ* section is defined by the following syntax:

```
#GROUP "name_of_group"
[
       question_1;
       question_2;
       ...
       question_N;
]
```

A question is defined by the following question function:

```
#Q( sNodeName, sQuestion, sLabel, InVariable, InVarType,
              sDefault, [ LinkVars ], sImgPath, x, y );
```

where `InVarType` has the following predefined input control types:

```
ERW_TEXT,
ERW_NUM,
ERW_BOOL,
ERW_PAGESOURCE,
ERW_PAGESIZE,
ERW_PAGEORIENTATION,
ERW_PAGEHEADERFOOTER,
ERW_DATASOURCE,
ERW_RELATEFIELDS,
ERW_SELECTFIELDS,
ERW_SUMMARYFIELDS,
ERW_PARAMETERIZE,
ERW_BORDERS,
ERW_COLOR,
ERW_FONT,
ERW_FILEPATH,
ERW_ALIGNFIELDS,
```


Copyright © 2003-2004 by Epsilon-Logic Systems

```
ERW_XMLDATASET,
ERW_XMLSELECTFIELDS,
ERW_XMLSUMMARYFIELDS,
ERW_XMLPARAMETERIZE,
ERW_XMLRELATEFIELDS.
```

Along with syntax for presentation of questions, *SRTmL* has a full set of keywords for conditional, iteration, as well as other special functions, making the language extremely powerful and flexible to be able to construct or customize wizard templates tailored to generate any type of standard or canned reports.

In addition to customizing or even creating standard report templates via *SRTmL*, the end-user may save the setup during the *SRT Wizard* session. For example, to save your answer entries for the questions in the *SRT Wizard*, you may utilize the **Save** button at the bottom of the *SRT Wizard* window. Clicking the **Save** button will call the **Save Report Template** dialog, which will save your setup along with the selected template as a *Binary Report Template* file with the file extension ".BRT".

Once a *SRT* setup is saved into *BRT* binary file, it may then be reused in the *SRT Wizard* window to create other report scripts just like the *SRTmL* based templates. For example, to reuse a *BRT* file, call the **New Report** dialog, and in this dialog enter the **Report Name** control, and make sure that the **Use Binary Report Template** radio option is selected, and then submit the dialog. This will prompt the **Open Binary Report Template File** dialog, in which you must find and select the desired *BRT* file and open it into the *SRT Wizard* window.

Essential Report Design GUI

The architecture of the *Report Designer* software, as an application for designing report scripts via visual methods, presumes the distinction between a *supplementary interface* and an *application wizard*. This distinction is important especially in the current state of the art of report making. Some report tools on the market have very well designed *application wizards*, but are lacking crucial *supplementary interfaces* to perform even very rudimentary functions such as, altering data source parameters after the report creation, or aliasing some fields defined in an existing report. Although future versions of the SCRIPT++ *Report Designer* will have an abundance of report wizards, nevertheless it is worth mentioning that ELS-Script software system, as a report tool, does not need to address the *application wizard* aspect. This is because report script files, as report templates, essentially have *Open Document Structure* (ODS). Therefore, most of the tasks of the development of custom or specialized report wizards are left to the ELS-Script programmer.

Consequently, the SCRIPT++ *Report Designer* may be viewed as a sophisticated SCRIPT editor with tightly integrated *supplementary interfaces*. As we will see in Chapter 2, that some aspects of report making are better performed via scripting methods, while some other features such as, setting colors or font style, resizing elements and defining borders, can be better controlled via visual methods.

In this subsection we will cover the following topics:

- **Project Explorer** and **Data View** GUI and associated commands
- Details about the **Design**, **Source** and **Browse** views
- Basic HTML edit operations
- Editing via the **Properties** pane
- HTML table and *ELS-Row* operations
- Using the **Precision Resize** tool
- Inserting data fields and expressions
- Formatting and conversion
- Details about the **Parameter List** and **Query Form** designer

Project Explorer

Essentially, the mainframe of the *Report Designer* application consists of the **Project Explorer**, the **Properties** and **Data View** panes, several multi-tabbed MDI child windows, of which the main windows are the report designer's **Design**, **Source** and **Browse** views, the **Query Builder**, the **Data Shape Builder**, the **SQL Editor**, together with several satellite windows and dialogs.

Copyright © 2003-2004 by Epsilon-Logic Systems

The **Project Explorer** is a handy tool to manage various report script files, data access connections, queries, data shapes and SQL scripts, organizing under a single project folder. In this way, the whole project is controlled from the **Project Explorer** pane, which is the docking window located at the top-left corner of the *Report Designer* main window. This window consists of a toolbar and a tree-view control displaying the content of the currently open project. In particular, the toolbar contains the following buttons:

Design View	used to switch or open the selected report in **Design** view,
Source View	used to switch or open the selected report in **Source** view,
Parameter List	used to slide-on the **Parameter List** window.

The tree-view control will in general display a project tree consisting of a root labeled **Workspace**, which contains four child nodes, namely a child node with project's name label, the **Database**, **XML** and **Miscellaneous** child nodes. In its turn, the project node, itself, will contain all the report script files that you create for the current project, as child nodes. It will also contain the **Include** and **Image** subfolder nodes. The **Database** node on the other hand, will contain all the data access connection nodes created in this project. Each connection node, in turn, will contain all the SQL script files created for the connection. Moreover, it will have an **AUX** folder node, which will contain the **Queries** and **Datashapes** child nodes. All queries created via the **Query Builder** will be put under the **Queries** node, and the data shapes created via the **Data Shape Builder** will be put under the **Datashapes** node.

The menu commands associated with the **Project Explorer** are categorized under general project management commands. Most of these commands are grouped under the **File** menu, while the rest are grouped under the **Edit** and **View** menus. The menu commands under the **File** menu are as follows:

New Project	**Ctrl+N**	to create a new report project,
Open Project	**Ctrl+O**	to open an existing report project,
Close Project		to close an open project,

New Report		to create a new report script,
Open Report		to open an existing report script external to the current project,
---		
New SQL Script		to create a new SQL script file,
Open SQL Script		to open an existing SQL script external to the current project,
New DB Connection		to create a new data access connection,

Add To Project		to add an external file to the project at the selected container node,
Remove From Project		to remove a selected object from the current project,
---		
SQL Dictionary	**F12**	to display the **SQL Dictionary** floating window,
---		
Save	**Ctrl+S**	to save all changes incurred upon the currently active document,
Save All		to save all changes incurred upon all open documents in the project,
Save As		to save the current report script or SQL script with another filename,
Save HTML		to save the whole report output as an HTML file,

Print	**Ctrl+P**	to call the standard windows print dialog to print the selected document,
Print Preview		to print preview the document or current page,
Page Setup		to call the page setup dialog to be applied over the selected view,
Print Current Page		to directly print the current page,
Report Settings		to call the **Report Settings** dialog,

Recent Files		to popup a child menu of recently opened report script files,
Recent Projects		to popup a child menu of recently opened report projects,
---		
Exit		to exit the application.

The only important menu command under the **Edit** menu that is associated with the project management is the **Find in Files** menu command, which calls the **Find In Files** dialog, to enter a regular expression search text applied

Copyright © 2003-2004 by Epsilon-Logic Systems

over multiple files.

The menu commands under the **View** menu are as follows:

Toolbars		to call the **Customize** dialog to customize the toolbars,
Project Explorer	**Ctrl+E**	to toggle the visibility of the **Project Explorer** pane,
Properties/Data View	**Ctrl+D**	to toggle the visibility of the **Properties/Data View** pane,
Status Bar		to toggle the status bar,
Full Screen		to toggle the display to full screen or windowed view,
---		
Options		to call the **Options** dialog to set report or project options.

In addition to these menu commands, the **Project Explorer** pane has a popup menu with the following commands:

Add Item	to add items into the current container node,
New	to create a new object of the current container node type,
---	
Open	to open the selected node or object,
Properties	to display the properties of the current node,
Rename	to make the current node's label editable and ready for renaming,
---	
Remove	to remove the selected node or object from the report project,
Copy	to copy the current node or object into the clipboard,
Paste	to paste the copied node from the clipboard into the project,
---	
Allow Docking	to allow window docking,
Hide	to hide the pane.

To display the popup menu, the user must position the mouse on an applicable node in the **Project Explorer** pane and using the right-mouse-button click on the node. Depending on the node, this popup menu will have different appearance. For example, on connection objects it will have an additional **Insert** menu command, and when the connection is open, the **Open** command is replaced by **Close** menu command. Also, depending on the node, the menu commands will have different interpretations. For example, the **New** menu command is interpreted as new data access connection when the right-mouse-button click is applied over the **Database** node, it will mean new query when applied over the **Queries** container node, new data shape when applied over **Datashapes** node, new SQL script when applied over the connection node, new XML document reference when applied on the **XML** node, and finally a new report script when applied over the project node.

The toolbar buttons associated with the **Project Explorer** are grouped under the **Project Explorer** toolbar, as shown in Figure 1.51 below.

Figure 1.51. Showing the project toolbar

In particular, from left to right, they are: **New Project**, **Add Item**, **Open**, **Save**, **Save All**, **Print**, **Page Setup**, **Preview**, **Print Current page**, **Cut**, **Copy**, **Paste**, **Undo**, **Redo**, **Toggle Bookmark**, **Find In Files**, **Project Explorer**, **Properties**, **Show Result Pane**, **Cascade Window**, **Tile Vertically**, **Tile Horizontally** and **Split**. Note that some of these buttons are disabled when no MDI child window is open.

We have already seen in the quick start section of this chapter, some project management operations that were performed via the menu commands described so far. In particular, we know by now how to create a project, and once a project is open, how to create a new report script via standard report templates. We have also touched upon the creation of data access connection and definition of queries via the **Query Builder**. More detailed exposition of such database tools will be given in the later sections of this chapter.

Continuing the description of the operations resulting from the remaining project management menu commands, the **Add To Project** and **Remove From Project** menu commands, alternatively **Add Item** and **Remove** popup menu commands, are used to control the content of a report project file. For example **Add Item** command will call the **Add To Project** window's common file dialog, as shown in Figure 1.52 below.

Copyright © 2003-2004 by Epsilon-Logic Systems

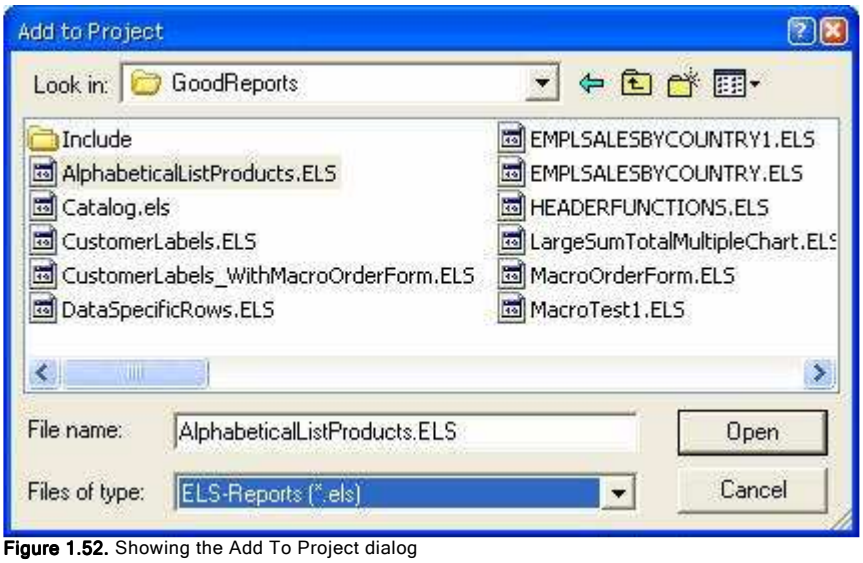


Figure 1.52. Showing the Add To Project dialog

In this dialog, the user may select files of the following types: *ELS-Reports* (*.els), *SQL-Queries* (*.sql), *Text-Files* (*.txt), *Internet Files* (*.htm, *.html, *.asp, *.xml) and *Cascading Style Sheets* (*.css). In addition, if **Add Item** command is applied over an **Image** folder, then you may also include image files of various formats including GIF, JPG, BMP, WMF and XBM. Now, any file that is added to the project from directory location external to the **REPORTS** folder will first make a copy of the file in the **REPORTS** folder, and then the filename will be added to the project file. Similar behavior holds for the other folders or subfolders, in particular, all external image files added to the project will be copied into the **Image** folder, so that the project file will have a reference to the local copy of the file.

In contrast, when removing files from the current project via the **Remove From Project** or **Remove** command, the file is first moved from the respective folder into a backup folder, and then the file reference is deleted from the project file. In this way, for fail-safe practical reasons, all removed files are put in a backup folder instead of physically deleting them.

We have already mentioned that a project file is a flat text file. In fact, the file structure of project files very much resembles classical *INI*-files, as is readily seen in the following NWindProj.RPJ sample project:

```
[PROJECT = NWindProj]
FILE = ProductsByCategory.ELS
FILE = CustomerLabelsWithMacroOrderForm.ELS
FILE = OrdersByProducts.ELS
FILE = CustomerLabels.ELS
FILE = SalesByCategory.ELS
FILE = OrdersByProductsWithPieChart.ELS
FILE = LargeSumTotalMultipleChart.ELS
FILE = Catalog.els
FILE = AlphabeticalListProducts.ELS
FILE = SumTotalMultipleChart.ELS
FILE = SALEBYYEAR.ELS
FILE = SalesByCatChartObj.ELS
FILE = Invoice.ELS
[INCLUDE]
FILE = styles.css
FILE = default.css
FILE = OrderForm.css
[IMAGE]
FILE = NWNamelogo.bmp
FILE = NWPictureLogo.bmp
[DATABASE]
[CONNECTION = NWindDB]
[CONNECTION = NWMDB]
[MISCELLANEOUS]
FILE = OldInvoice.ELS
[CONNECTION SETTINGS]
```


Copyright © 2003-2004 by Epsilon-Logic Systems

```
NWindDB = "Provider=SQLOLEDB.1; Persist Security Info=True;User ID=sa; Password=;
                                  Initial Catalog=NORTHWIND; Data Source=Server1"
NWMDB = "Provider=Microsoft.Jet.OLEDB.4.0; Data Source=C:\Data\Northwind.mdb;
                                                  Persist Security Info=False"
```

You may customize the toolbar arrangement by positioning them horizontally or vertically at the top, bottom, left or right side of the *Report Designer* main window. You may also customize the content of each individual toolbar, or arrange them inside another user-defined toolbar. To customize toolbars, select the **Toolbar** menu command from the **View** menu. This will display the **Customize** dialog, as shown in Figure 1.53 below:

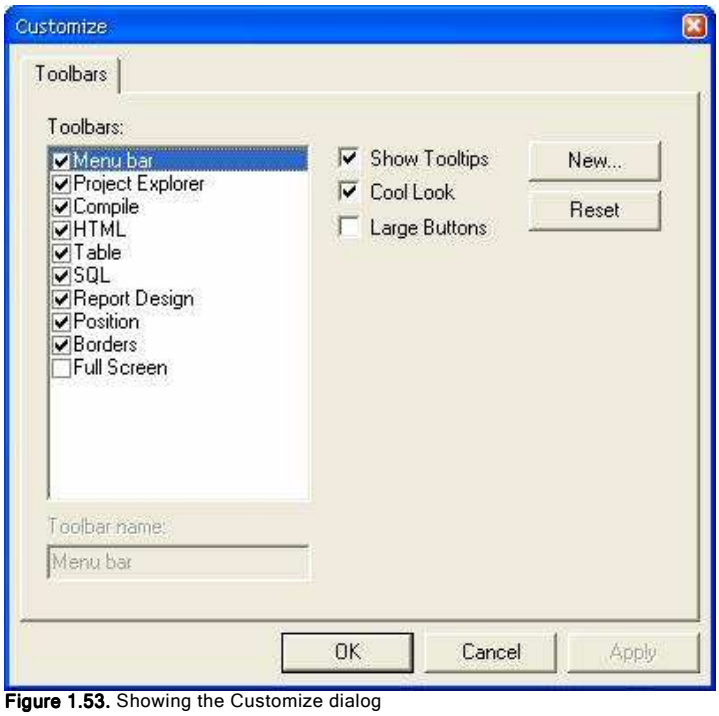


Figure 1.53. Showing the Customize dialog

ELS-Script® package comes with a User's Guide consisting of three chapters (including the current Chapter 1 document), an online query builder help file and references to technical articles and support web sites. In particular, the **Help** menu of the *Report Designer* application contains menu items referencing these support material:

Getting Started	to open Chapter 1 of the User's Guide document,
Basics of SCRIPT Language	to open Chapter 2 of the User's Guide document,
Report Engine Deployment	to open Chapter 3 of the User's Guide document,
Report Engine API Reference	to open the *Report Engine API Reference* help file,
Query Builder Help	to open the *Query Builder Help* file,

Check for Updates	to direct the user to the main ELS-Script web site for new updates,
Support and Technical Articles	to direct the user to the technical article and support web site,
---	
About ELS-Script	to display application information,

Note that the *Report Designer* application, with its powerful **Query Builder**, **Data Shape Builder** and **SQL Editor** modules, may also be used as a database administration tool. Using the commands available under the SQL menu, the user may create all types of queries including *SELECT*, *INSERT*, *INSERT VALUES*, *UPDATE*, *DELETE* and *Make Table* queries. Moreover, the commands of the **Data View** pane combined with the **SQL Dictionary** makes the **SQL Editor** a formidable tool that can be extremely helpful in the creation of SQL scripts, including stored-procedures, *DML* and *DDL* scripts. We should point out that besides SQL commands the **SQL Editor** understands *Data Shape* and *DMX* commands as well. For further information on the **Query Builder**, **Data Shape Builder** and **SQL Editor** modules please click the **Query Builder Help** menu command.

Copyright © 2003-2004 by Epsilon-Logic Systems

The *Report Designer* has a text search feature to search a word or pattern over multiple files. This search feature may be triggered via the **Find In Files** menu command, which will prompt the **Find In Files** dialog. To activate a text search the user must enter some word or pattern in the **Find what** field, for example "`Orders`", then select a file extension type in the **In files/file types** field, set a folder in the **In folder** field, and finally, click the **Find** button. Figure 1.54 shows the **Find In Files** dialogs with some parameter set to search the pattern "`Orders`" over all files with extension type `*.els`.

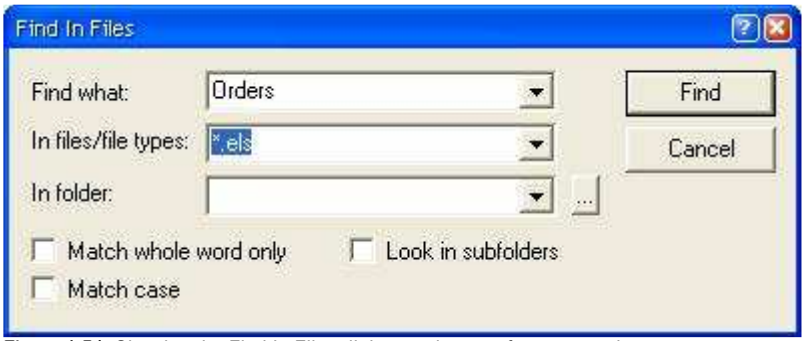


Figure 1.54. Showing the Find In Files dialog ready to perform a search

After running a text search, the result of the search is shown in the **Find in Files** worksheet tab of the **Result** pane, as shown in Figure 1.55 below.

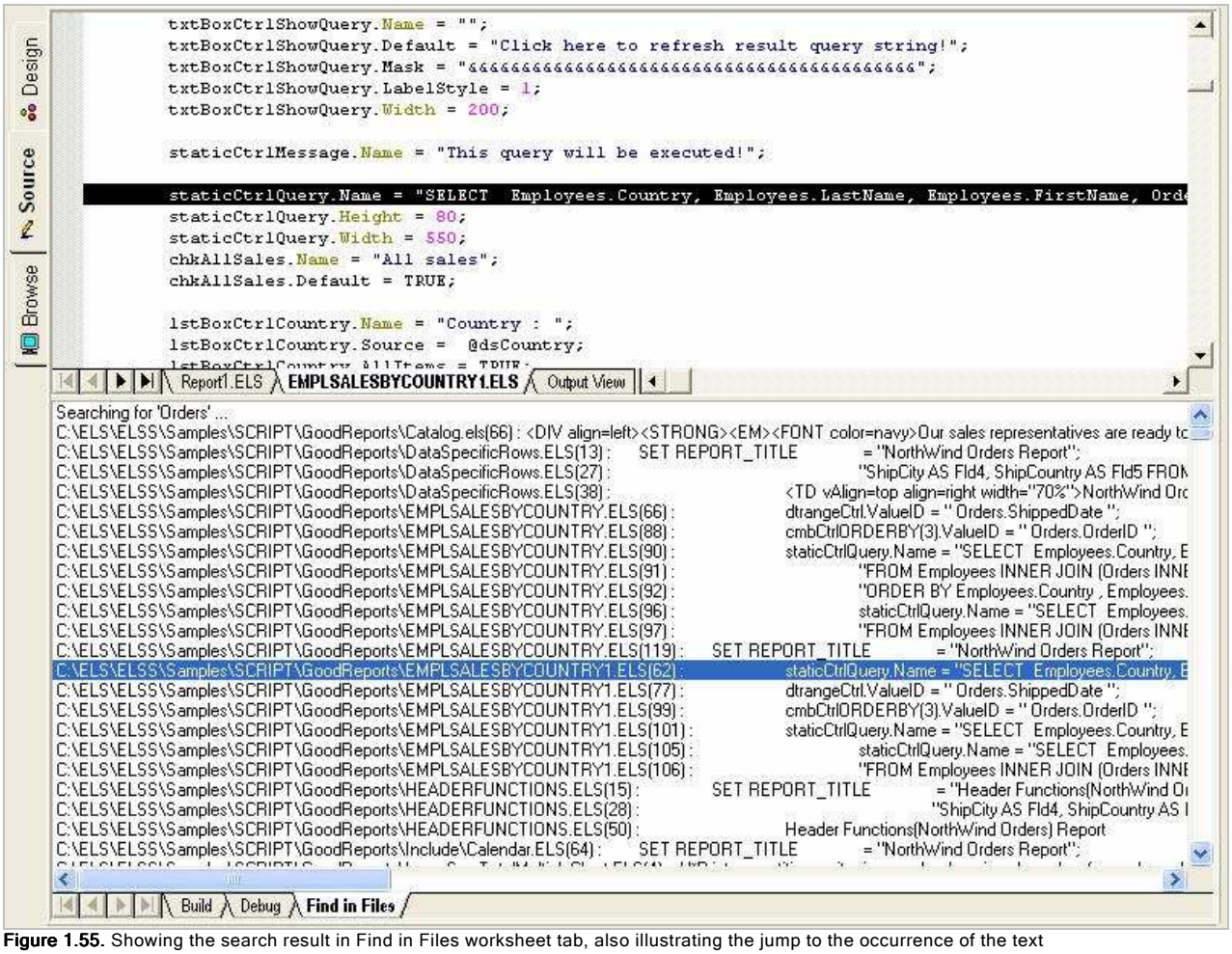


Figure 1.55. Showing the search result in Find in Files worksheet tab, also illustrating the jump to the occurrence of the text

In particular, each line of the result is preceded by the filename and the line number where the occurrence of the search text was encountered. A double-click on a result line will open the corresponding file and scroll to the line in the file where the text was encountered.

Copyright © 2003-2004 by Epsilon-Logic Systems

Note that the **In folder** field defines the scope of the files over which the search process will run. To select a value for this field, click on the 3-dotted button on the right side of the control to display the **Browse for Folder** dialog shown in Figure 1.56 below

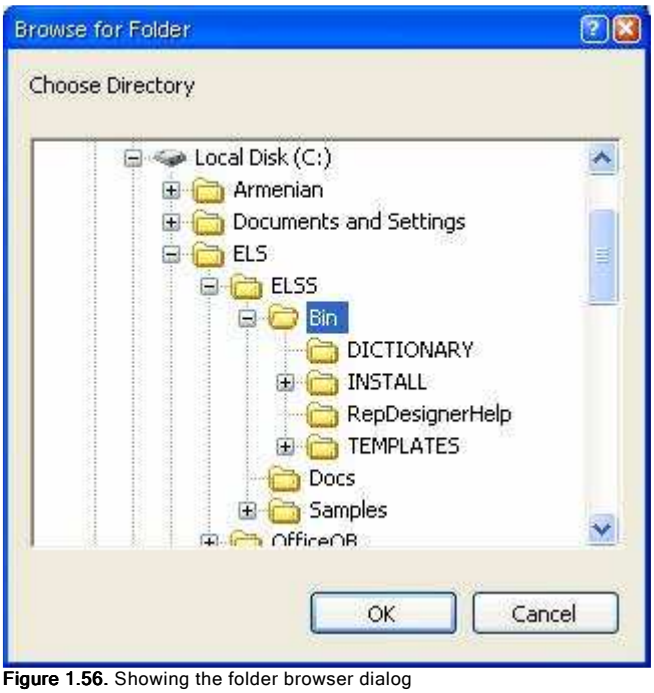


Figure 1.56. Showing the folder browser dialog

Properties / Data View

In the task of project management, the **Data View** pane plays the second major role after the **Project Explorer** window. With this window, the user may explore the structure of the backend databases, and define the elements of data retrieval in queries or data shapes, which eventually will be used inside report scripts.

To populate the **Data View** pane, the user must open an existing connection node from the **Project Explorer** pane. In general, the pane displays a tree-view of database objects corresponding with the connection node. Essentially, the tree has a root node with label **Database**. This root node then will have immediate child nodes, each being a base node representing a database under the selected connection node. Note that a data access connection defined in the project, may apply over multiple databases, so that the **Database** root node may contain more than one child node. Each database node itself may contain **Tables**, **Views**, **Procedures**, and **Functions** folder child nodes. These folder child nodes respectively will contain child nodes representing the tables, the views, the stored-procedures and the user-defined functions of the selected database node. A node representing a table or view database object, will itself contain child nodes representing the fields of, respectively, the table or view object. On the other hand, a node representing a stored-procedure or function will contain **Parameters** and **Returns** child nodes.

The *Report Designer* application has many menu commands reserved for the **Data View** pane. For example, most of the menu items under the **SQL** menu apply to the operations defined over the **Data View** pane. These menu commands will be described in more elaborate detail in the query builder and later sections of this chapter. For now, we will describe only the menu commands that are grouped under the popup menu of the **Data View** pane. This popup menu consists of the following menu items:

Refresh	to refresh the tree-view to show the most recent updates to the backend database,
Get Details	to retrieve the structural definition of the selected database object,
----------	
Add Table	to add the selected table into the **Relations** pane of the query builder window,
Allow Docking	to make the window a docking window,
Hide	to hide the window.

In addition to the popup menu, the tool-tips of the **Data View** pane reveal valuable information about the objects pointed by the mouse pointer. For example, if the pointer is moved to be over a field node, the field data type or structure is displayed in the tool-tip, as shown in Figure 1.57 below. Similarly, when moved over a **Parameters** or

Copyright © 2003-2004 by Epsilon-Logic Systems

Returns child node, the data type of the parameter of variable is displayed in the tool-tip.

The **Data View** pane is coupled with the **Properties** pane, which provides an alternative more precise method of editing HTML and *ELS-Row* elements in the **Design** view. Essentially, the **Properties** pane consists of a toolbar with the **Alphabetic** and **Categorized** toggle buttons, the tag selection combo-box and the property evaluation grid, as shown in Figure 1.58 below.

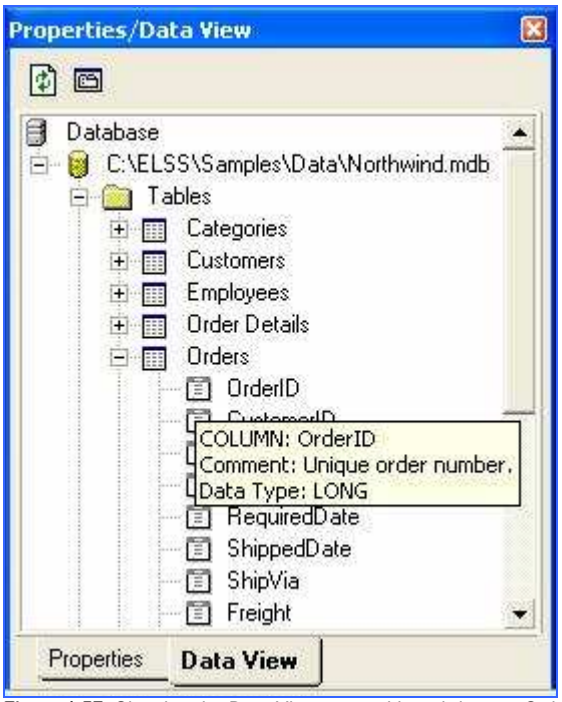


Figure 1.57. Showing the Data View pane with tool-tip over OrderID

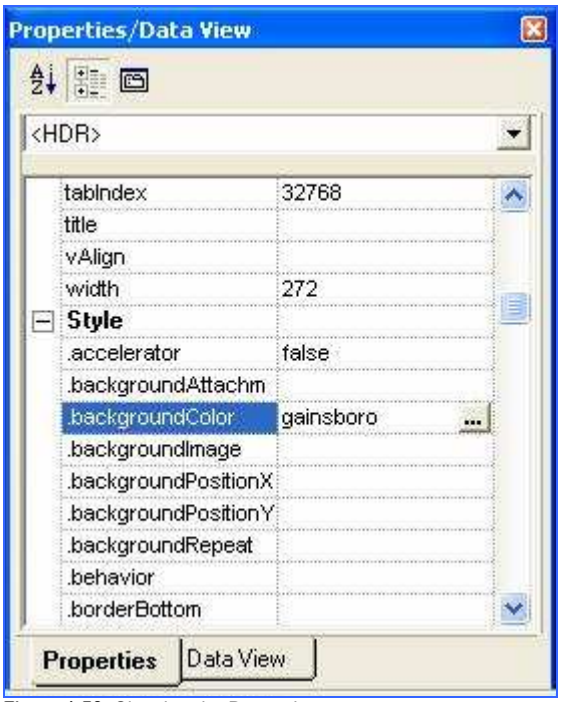


Figure 1.58. Showing the Properties pane

We will describe the features and operations involved with the **Properties** pane in later sections of this chapter.

Design, Source and Browse Views

At the beginning of this chapter, we learned that the SCRIPT++ *Report Designer* is essentially a sophisticated SCRIPT editor with tightly integrated *supplementary interfaces*. The main modules or components of these supplementary interfaces are the three views of the report designer's main window, namely the **Design**, **Source** and **Browse** views. Of these three views, the **Design** and **Source** views fit naturally into the dichotomy of the art of report making. Remember that some aspects of report design are better performed via scripting methods, while some other features such as, setting colors or font style, resizing elements and defining borders, can be better controlled via visual methods. Therefore, depending on the report design task that confronts the user, she/he may switch relentlessly between the **Design** and **Source** views. For example, when it comes to editing of the visual elements of the report, the user may switch to **Design** view and utilize visual edit operations, while on the other hand, when the operations involved need more detailed syntactical or scripting control, then the user may switch back to the **Source** view.

The **Design** view essentially consists of the following report sections separated by window splitters:

PH for the *Page Header* report section,
RH for the *Report Header* report section,
D for the *Report Detail* report section,
RF for the *Report Footer* report section,
PF for the *Page Footer* report section.

In addition, the window has a horizontal ruler, and each of the sections have separate vertical rulers, to help the user measure the content of the view in inch page units. Note that the content of the *Report Header* and *Report Footer* will be put respectively at the beginning and at the end of the report output. Such information may contain summaries, totals or grand totals of the elements in the report. The content of *Page Header* and *Page Footer* will

Copyright © 2003-2004 by Epsilon-Logic Systems

be put respectively at the top and bottom of each report output page. This may include report title, pagination, report run-date, as well as other such page header / footer information. The *Report Detail* section is where most of the data presentation will take place when the report output is generated.

Figure 1.59 shows a sample report illustrating the content of the report sections. As seen from the **Design** view, the report sample has the report title repeated in the header of each report output page along with the page information, given by the formula: `"P "+Format(PageNum(), "")+" / "+Format(PageCount(), "")`. For example, on the left side of the header of the first page of the report output we should see "Orders/Orders Details", while on the right side we should see "P 1/98". Also, observe that the page footer will contain the "RUN DATE-TIME:" text followed by the actual date-time when the report generation was initiated.

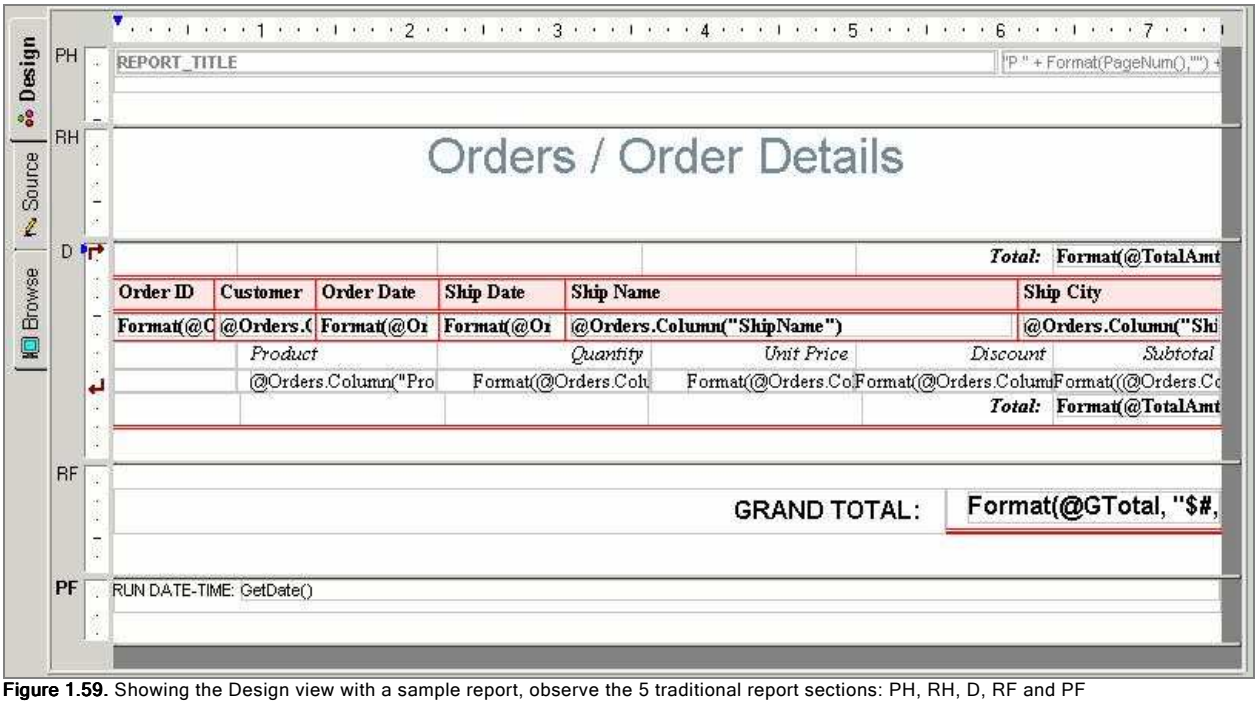


Figure 1.59. Showing the Design view with a sample report, observe the 5 traditional report sections: PH, RH, D, RF and PF

Continuing our observation, we see then that the *Report Header* section contains the "Orders / Order Details" text in centered large font-size, and the *Report Details* section contains several table-like structures with column headers and some sort of fields that resemble formulas. Finally, the *Report Footer* section contains the text "GRAND TOTAL:" followed by yet another formula field. Also, observe that the rulers in the **Design** view contain some sort of arrows and blue pointers resembling markers. We will see later on that these pointers and arrows mark the location where conditional options or iteration starts and ends.

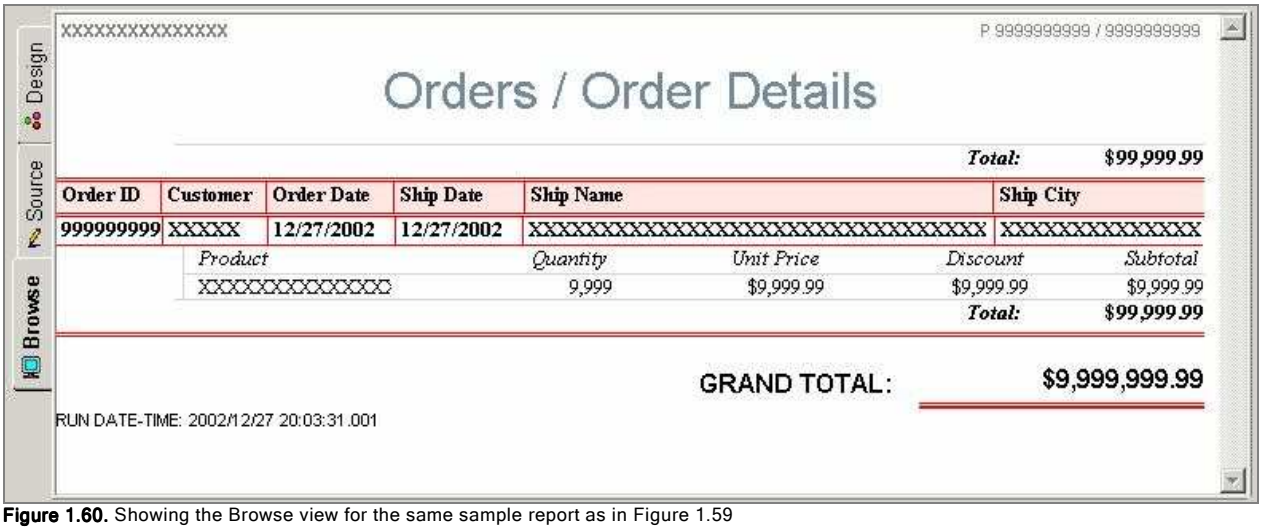


Figure 1.60. Showing the Browse view for the same sample report as in Figure 1.59

Copyright © 2003-2004 by Epsilon-Logic Systems

Now, these formula fields may perhaps be a little confusing to some users, in which case they may switch to the **Browse** view to see how the actual fields may look like when dummy values are substituted. This is illustrated in Figure 1.60 for the same sample report as that of Figure 1.59. Therefore, as we can see, that the **Browse** view is indeed very useful in diagnosing the actual format of the fields, as well as the appearance of the borders and the backgrounds of HTML elements.

The **Source** view is where the user has full control over the report script, which essentially is the actual source code of the report itself. The **Source** view comes with full *IntelliSense* support making SCRIPT programming a very intuitive task. We will describe in details about SCRIPT language in Chapter 2 of this manual, for now, we will continue covering the rest of the graphic user interface elements related to the **Design** and **Source** views.

Some of the main menus applicable to both **Design** and **Source** views are the **Insert**, **Format**, **Table** and **Tools** menus. The **Insert** menu consists of the following menu items:

Break		to insert a line break ` ` at the cursor location,
Rule		to insert a horizontal rule `<HR>` at the cursor location,
New Paragraph	**Ctrl+Enter**	to insert new paragraph tags `<P> </P>` at the cursor location,
New Section		to insert new section tags `<DIV> </DIV>` at the cursor location,
Append New Section	**Ctrl+W**	to append new section tags `<DIV> </DIV>` at the end of the report section,
---		
Special Symbol		to call the **Special Symbols** dialog,

Picture		to call the **Insert Image** dialog,
Background Image		to call the **Select Background Image** dialog,
---		
HTML Table		to call the **Insert Table** dialog,
ELS Row		to call the **Insert ELS-Row** dialog,
ELS Shape		to call the **Insert ELS-Shape** dialog,
Expression Builder		to call the **Expression Builder** window,
Data Fields		to call the **Data Fields** window,

ActiveX Control		to call the **Insert ActiveX Control** dialog.

The **Format** menu consists of the following menu items:

Bold	**Ctril+B**		to make the selected text bold,
Italic	**Ctril+I**		to make the selected text italic
Underline	**Ctril+U**		to make the selected text underlined,

Superscript			to make the selected text superscript,
Subscript			to make the selected text subscript,

Alignment	►	**Left**	to align the current content horizontally to the left,
		Center	to center the current content horizontally,
		Right	to align the current content horizontally to the right,
Increase Indent			to increase the indentation of the current content,
Decrease Indent			to decrease the indentation of the current content,
---			
Bulleted List			to make the current selection a bulleted list,
Numbered List			to make the current selection a numbered list.

The **Table** menu consists of the following menu items:

Insert Row	to insert a row before the selected row in an HTML table,
Append Row	to append a row at the end of the rows of an HTML table,
Delete Row	to delete the selected row from an HTML table,

Insert ELS Line	to insert *ELS-Line* before the selected line in an *ELS-Row*,

Copyright © 2003-2004 by Epsilon-Logic Systems

Insert Column Headers		to insert *HDR* line before the selected line in an *ELS-Row*,
Duplicate ELS Line		to duplicate the selected *ELS-Line* in an *ELS-Row*,
Delete ELS Line		to delete the selected *ELS-Line* from an *ELS-Row*,
Move Line Up		to move the current line up one step,
Move Line Down		to move the current line down one step,

Insert Column		to insert a column before the selected column in *HTML table* or *ELS-Row*,
Append Column		to append a column at the end of the columns in *HTML table* or *ELS-Row*,
Delete Column		to delete the selected column from *HTML Table* or *ELS-Row*,
---		
Insert Cell		to insert a cell at the cursor location in *HTML table*,
Delete Cell		to delete the selected cell from *HTML table*,
Merge Cell		to merge the selected cells of *HTML table* into one cell,
Split Cell	► **Vertical**	to split the selected cell of *HTML table* into two rows,
	Horizontal	to split the selected cell of *HTML table* into two columns.

The **Tools** menu consists of the following menu items:

Connection/DataSource List	to call the **Connection/Datasource List** dialog,
Parameter List	to slide-on the **Parameter List** window,
Query Form	to slide-on the **Query Form** designer window,
Precision Resize	to call the **Precision Resize** tool,
Show Report Wizard	to make an active *SRT Wizard* window visible,
---	
Edit SQL Dictionary	to call the **SQL Dictionary** window in edit mode.

This covers the menus of the *Report Designer* application that are related to the **Design** and **Source** views. We will close this section by outlining the toolbars corresponding to these menu commands. In particular, the toolbar that are relevant to the **Design** and **Source** view operations are the **HTML**, **Table**, **Report Design**, **Position** and **Borders** toolbars.

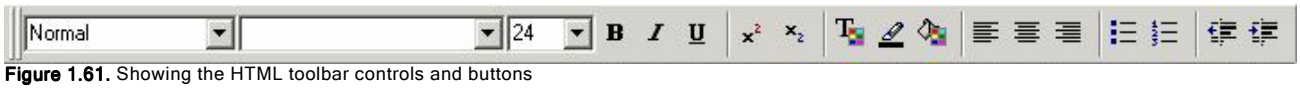


Figure 1.61. Showing the HTML toolbar controls and buttons

The **HTML** toolbar is shown in Figure 1.61 above, and consists of the following controls or buttons:

HTML Style	combo-box control, to define an HTML style for the selected text in the **Design** view (not recommended for use with paged-mode reports, instead use absolute font information, for example, the features available via the next controls),
Font	combo-box control, to define the font type for the selected text in the **Design** view,
Font size	combo-box control, to define the font size for the selected text in the **Design** view,
---	
Bold	button, to make the selected text bold,
Italic	button, to make the selected text italic,
Underline	button, to underline the selected text,

Superscript	button, to make the selected text superscript,
Subscript	button, to make the selected text subscript,

Fore Color	dropdown button, to define the fore color of the selected text,
Highlight	dropdown button, to define the highlight color and highlight the selected text,
Back Color	dropdown button, to define the background color of the container of the selection,
---	
Align Left	button, to make the selection left aligned,
Center	button, to make center the selection,
Align Right	button, to make the selection right aligned,

Bulleted List	button, to form the selected content into a bulleted list,

Copyright © 2003-2004 by Epsilon-Logic Systems

Numbered List button, to form the selected content into a numbered list
---
Decrease Indent button, to decrease the indentation by one step,
Increase Indent button, to increase the indentation by one step.

The **Table** toolbar is shown in Figure 1.62, and consists of the following buttons:

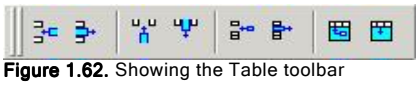


Figure 1.62. Showing the Table toolbar

Insert Row button, to insert a new row into HTML table,
Delete Row button, to delete the selected row of HTML table,

Insert Column button, to insert column into HTML table or ELS-Row,
Delete Column button, to delete the selected row from HTML table or ELS-Row,
---
Insert Cell button, to insert a new cell into HTML table,
Delete Cell button, to delete the selected cell from the HTML table,
---
Merge Cells button, to merge the selected cells into a single cell,
Split Cell Vertically button, to split the selected cell into to columns.

Figure 1.63. Showing the Report Design toolbar

The **Report Design** toolbar is shown in Figure 1.63, and consists of the following buttons:

Design View button, to switch or open the report in **Design** view,
Source View button, to switch or open the report in **Source** view,
---
Open button, to open the selected report in the default view,
---
Insert ELS Row button, to call the **Insert ELS-Row** dialog,
Insert ELS Shape button, to call the **Insert ELS-Shape** dialog,
Insert HTML Table button, to call the **Insert Table** dialog,
Insert Image button, to call the **Insert Image** dialog,
Expression Builder button, to call the **Expression Builder** window,
Insert Data Field button, to call the **Data Fields** window.

Figure 1.64. Showing the Position toolbar

The **Position** toolbar is shown in Figure 1.64, and consists of the following buttons:

Visible Borders button, to toggle the visibility of the outlines of the borders,
---
Absolute Position button, to toggle the mode of positioning between absolute and relative,
Snap to Grid button, to snap to the virtual grid of the view,
Lock button, to lock the object's position,

Bring to Front button, to bring the object in front of another object,
Send to Back button, to send the object to the back of another object,
---
Show/Hide Grid-guide button, to toggle the visibility of background grid-guide,
---
Precision Resize button, to call the **Precision Resize** tool.

Copyright © 2003-2004 by Epsilon-Logic Systems

Figure 1.65. Showing the Borders toolbar

Finally, the **Borders** toolbar is shown in Figure 1.65, and consists of the following buttons:

Border Type	button, to define the border line type of the selected border (e.g. solid, dotted, etc.),
Border Weight	button, to define the weight of the selected border in points,
Border Color	button, to define the color of the selected border,
Border	button, to select (or remove) the border.

Copyright © 2003-2004 by Epsilon-Logic Systems