



## Desktop Studio: SQL Editor

---

Intellicus Enterprise Reporting and BI Platform



©Intellicus Technologies  
info@intellicus.com  
www.intellicus.com

---

Copyright © **2011** Intellicus Technologies

This document and its content is copyrighted material of Intellicus Technologies. The content may not be copied or derived from, through any means, in parts or in whole, without a prior written permission from Intellicus Technologies. All other product names are believed to be registered trademarks of the respective companies.

**Dated: - August 2011.**

## **Acknowledgements**

Intellicus acknowledges using of third-party libraries to extend support to the functionalities that they provide.

For details, visit: <http://www.intellicus.com/acknowledgements.htm> .

---

## Contents

---

<b>Working with SQL Editor.....</b>	<b>4</b>
Using the SQL Editor .....	5
Selecting Schema/ Database/ Owner .....	5
Selecting Database Objects .....	5
Creating Join Conditions .....	6
Using SQL Parser .....	7
Designing an SQL Statement .....	7
Writing Power SQL.....	9
Dynamic SQL Queries using parameters .....	10
Dynamic SQL javascript code block .....	10
Stored Procedures .....	12
Compiling SQL .....	13
Viewing SQL Results .....	13
Applying Filters .....	14
Sorting at Run Time.....	16
Removing Tables from SQL Editor .....	17
Removing a Join .....	17

## Working with SQL Editor

This chapter will explain the utility of SQL Editor for creating reports. You can use this editor to simplify the query generation for the reports. You can use SQL Editor to generate a query to retrieve data from the connected database and then use this data to create a report.

Using this editor, you can select different database objects, create queries, check the queries for any errors and save them for later use.

To access **SQL Editor**, click **Tools > SQL Editor**.

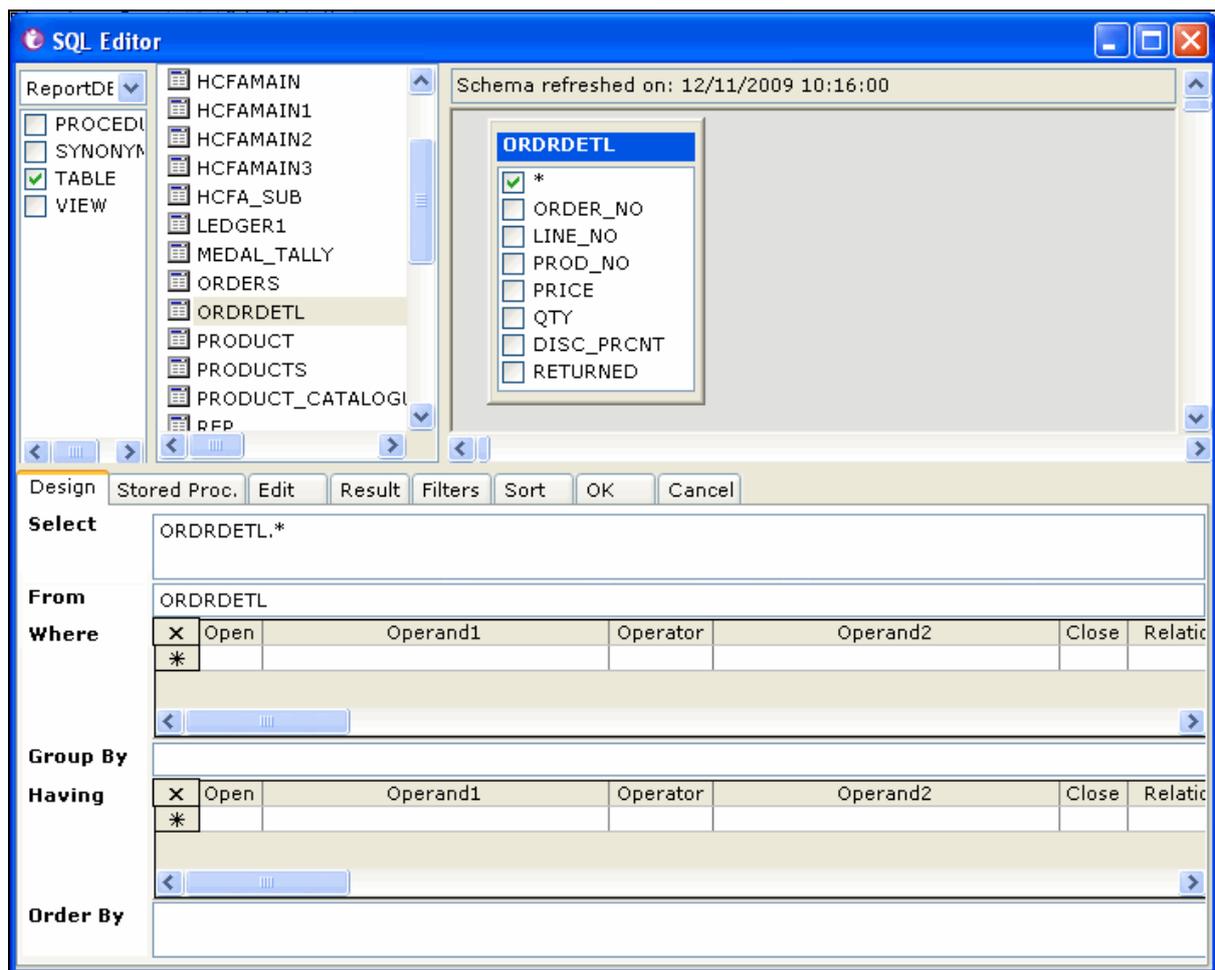


Figure 1: SQL Editor

You can resize the individual panes for a better view by clicking the border between two panes and dragging it vertically or horizontally.

Just below the title bar, SQL Editor displays date and time when the schema was refreshed.



**Warning:** If you do not define an SQL query, you will not be able to create a report through the Layout Editor.

---

## Using the SQL Editor

Using the SQL Editor involves the following tasks:

### Selecting Schema/ Database/ Owner

You can select the owner of the objects to be listed, from a combo box at the top left corner of the SQL Editor dialog box.



Figure 2: SQL Schema

### Selecting Database Objects

Check boxes for selecting Procedure, Synonym, Table, and View are provided in the SQL Editor. On selecting the required (schema) objects, the procedure, table, synonym, and view names will appear in the list pane adjacent to it. You can select these names and drag it to the diagram pane.

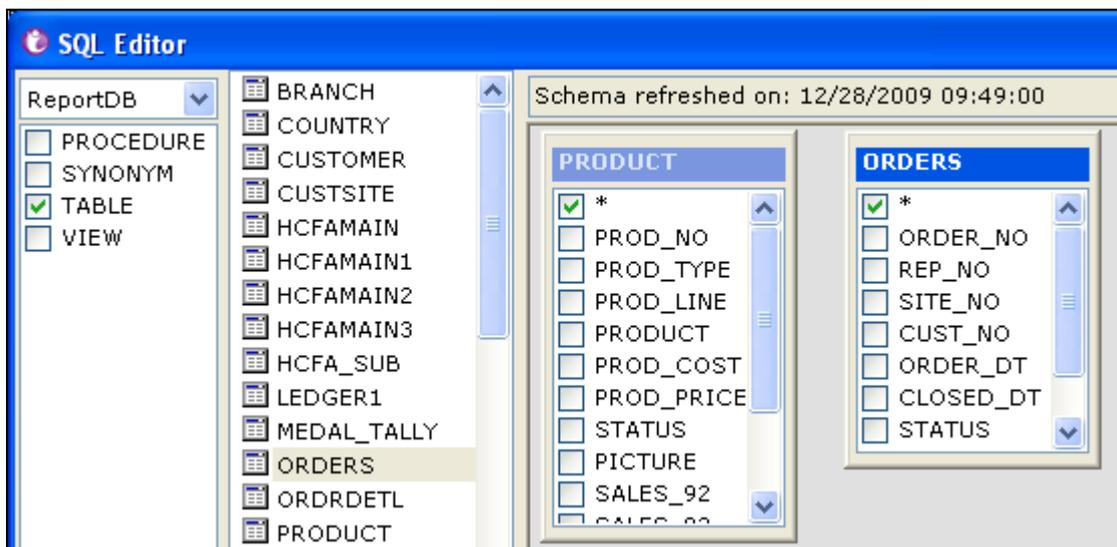


Figure 3: Selecting Database Objects

You can perform multiple selections using the following methods:

- Use <ctrl><click> to select random values
- Use <shift><click> for a continuous range of values, or,
- Simply click and drag individual values.



**Important:** Before you use a procedure, make sure that it does not perform any unwanted data manipulation in the connected data source.



**NOTE:** For OLAP type connections Design tab, Filters tab, Sort tab, Stored Proc. tab will not be available. You can use Edit tab to write the query.

## Creating Join Conditions

Joins can be used to unite some or all of the data from two or more tables into one comprehensive structure. You can perform joins using the diagram pane. To do this:

1. Click the value(s) from a database Field List.
2. Drag it to the required field.
3. When the cursor takes  shape, release the click.

The join will be indicated as an arrow symbol between the selected fields.

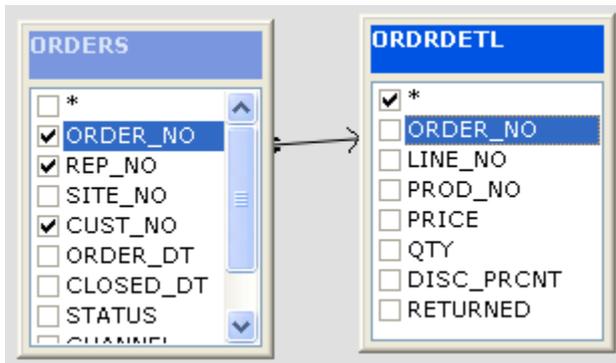


Figure 4: Performing Joins



**Note:** When you select two tables, Intellicus may automatically create join between the two. This would happen because a Primary Key - Foreign Key relation was already created in the database and **Enable Autolink Foreign Key** checkbox (Options > Custom Design) is selected (checked). In this case, respective condition will also be added in WHERE clause of the query.

If **Enable Autolink Foreign Key** checkbox is clear (not checked), Join will not be automatically created.

---

## Using SQL Parser

The formatted SQL statement is provided at the bottom of the SQL Editor dialog box. The values in this formatted SQL statements are populated as per the selections performed in the pane(s) given above it.

This facilitates quicker generation of SQL statements and allows editing the same as per requirement.

×	Open	Operand1	Operator	Operand2	Close	Relatio
*						

×	Open	Operand1	Operator	Operand2	Close	Relatio
*						

Figure 5: SQL Parser

The various options covered under this parser are defined under six categories with the help of tabs.

## Designing an SQL Statement

The **Design** tab contains a complete SQL statement divided into its clauses, according to the objects chosen from the diagram pane.

The SQL clauses available in this tab are:

### Select Clause

In the *Select* clause, the columns selected from the diagram pane are displayed. This allows quick selection and removal of the columns from the select clause. You can also edit this clause by directly writing into that box. Addition of a column using the SQL clause box will reflect on the selections performed in the diagram pane. But removal of a column might not be reflected in the diagram pane.



**Tip:** You can press <ctrl><spacebar> to seek context sensitive list of values in all clauses.

---

## Where Clause

In the *Where* clause, every condition is a row, and the following options are available:

Option	Description
<b>Open</b>	This option helps in grouping the conditions by using opening braces.
<b>Operand 1</b>	This is a combo box with all available column names. This will also display the parameters (report, system) defined in the report.
<b>Operator</b>	This is a combo box, which contains the comparison operators that you may use to compare operand 1 with operand 2.
<b>Operand 2</b>	This is a combo box with all available column names and parameters. It can be used for comparison with the operand 1 chosen earlier.
<b>Close</b>	This option helps in grouping the conditions by using closing braces.
<b>Relation</b>	You can relate the current condition with the next condition using 'AND' or 'OR' logical operators.

## Group by Clause

In the *Group By* clause you can provide grouping criteria for the SQL statement.

## Having Clause

In the *having* clause you can define conditions, similar to those defined in the *Where* clause.

## Order by Clause

In the *Order By* clause you can provide sorting (ascending/ descending) criteria for the SQL statement.



**Important:** For a report with grouping, the order by clause must have the columns in the same order as of the respective sections in the layout editor.

## Writing Power SQL

When you switch from the **Design** tab to the **Edit** tab, the SQL in the **Design** tab is constructed and displayed as a complete SQL statement in the later. Using the **Edit** tab, you can view and write complex SQL statements that cannot be defined using the **Design** tab.

During Query writing, make use of **Context Help**. It assists you in correct code (syntax) formation. As you type the code, you have a choice to have context view to get help about what to choose at a place. A window pops up with the right options for you to choose one. Window disappears upon choosing one option. More information on Context Help is available on Online Help.

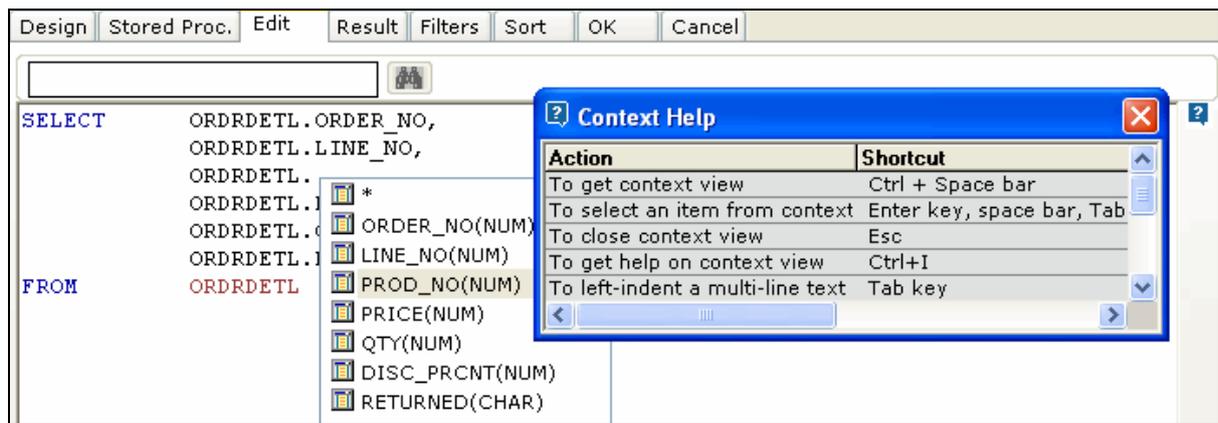


Figure 6: Editing SQL Statement

To prevent a complex query (defined in the **Edit** tab) from being over written, when you switch over to the **Design** tab, make some changes, and get back to **Edit** tab; You will be prompted with a dialog.

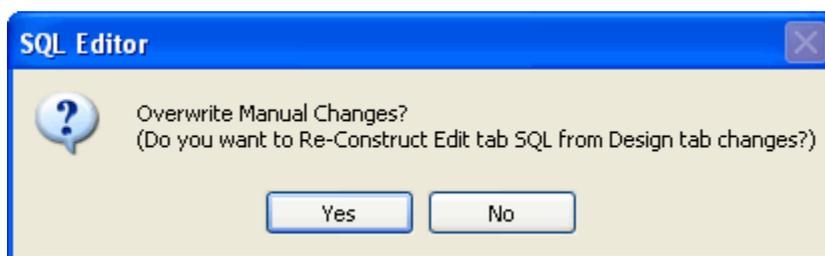


Figure 7: SQL Overwrite Prompt

If you select **Yes** your changes in the **Edit** tab will be over written, because the SQL in the **Design** tab will be reconstructed. If you select 'No' then the SQL in the **Edit** tab will remain intact and shall be used as the final SQL.



**Note:** The SQL statement in the **Edit** tab will be used as the final SQL for compilation.



**Important:** If SQL is manually specified in Edit tab (not constructed using design tab), its field details will not be available in Filter tab.

---

## Dynamic SQL Queries using parameters

To include a parameter in the SQL Query, specify a parameter enclosed within <% and %>.

Example: To dynamically decide where clause:

```
SELECT * FROM EmpTbl WHERE <%PrmWhrCls%>
```

## Dynamic SQL javascript code block

Use javascript code block enclosed by <@% and %@> to create a dynamic SQL Query. After execution, the code block should return a string that will be replaced in the SQL query.

You can use following in the code block:

- if else
- Parameter value and parameter data type

## Syntax of Code block

```
<@% >  
[Executable block, which returns a string]  
<%@>
```

## Syntax to use parameter in code block

```
params("parameterName").getAttribute()
```

**Attributes:** "Value" to get value of the parameter and "DataType" to get data type of the parameter.

## Examples

Example 1. Check if parameter "prmEmpNo" exists. If it does and its value is not blank, return the value, else return nothing.

```
Select * from emp where 1=1  
<@%  
if(params("prmEmpNo") != null &&  
params("prmEmpNo").getValue() != "")  
{  
return " AND empno in <%prmEmpNo%>";  
}  
else  
{
```

```
return "";
}
%@>
```

**Example 2.** Check if parameter "prmSelectTable" exists. If it doesn't, return table name as "emp" else (if it exists) value of the parameter to be used as table name.

```
Select * from
<@%
if(params("prmSelectTable") == null)
{
return "emp";
else
return params("prmSelectTable").getValue();
}
%@>
```

**Example 3.** Construct WHERE clause dynamically (for LOCATIONTYPEID and REGNID fields).

```
select      LOCTIONID,      LOCTIONNAME, LOCTYPEID, REGNID      from
LOCATIONMASTER where 1=1
<@%
var v Str;
v Str = '';
var v Str1;
var v Str2;
var v Str1='';
var v Str2='';

if(params("prmloctype") != null &&
params("prmloctype").getValue() != "")
{

v_Str1 = " AND LOCATIONTYPEID in (<%prmloctype%>)";
}
else
{
v_Str1 = "";
}

if(params("prmRegion") != null &&
params("prmRegion").getValue() != "")
{
```

```

v_Str2 = " AND REGNID = <%prmRegion%> ";
}
else
{
v_Str2 = "";
}

v_Str = v_Str1 +v_Str2;
return v_Str;
%@>

```

## Find Button

The Edit tab has a find button. For long queries (refer to the figure on previous page), you can search the query for a search string to reach the location quickly.

## Stored Procedures

Use this in case the data comes using a Stored Procedure. If you select Procedure check box from Object list appearing on top left of the SQL Editor, Design tab will automatically get disabled and Stored Procedure tab will be enabled.

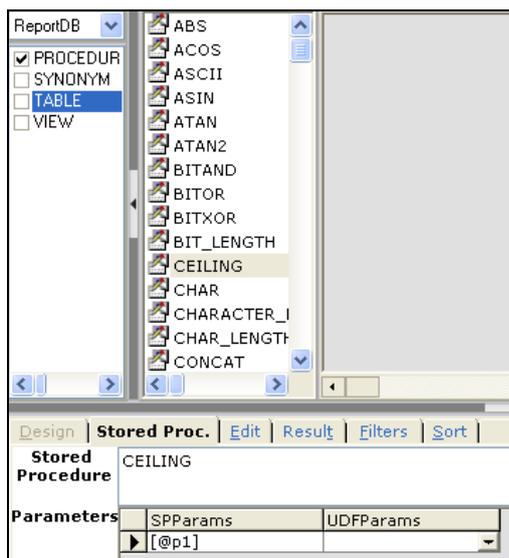


Figure 8: Selecting a Stored Procedure

You will be required to drag selected Stored Procedure into parser.

It will populate the Stored Procedure Parameters. You need to map either value or user defined parameters under UDFParams column.

## Compiling SQL

When you select the **Result** tab, the defined SQL statement will be compiled. Record-set will be displayed in case of successful completion. If there was any error, same tab will display the error. This would help you in finding the exact location of error(s), and rectify them before using the SQL results in the Layout Editor.

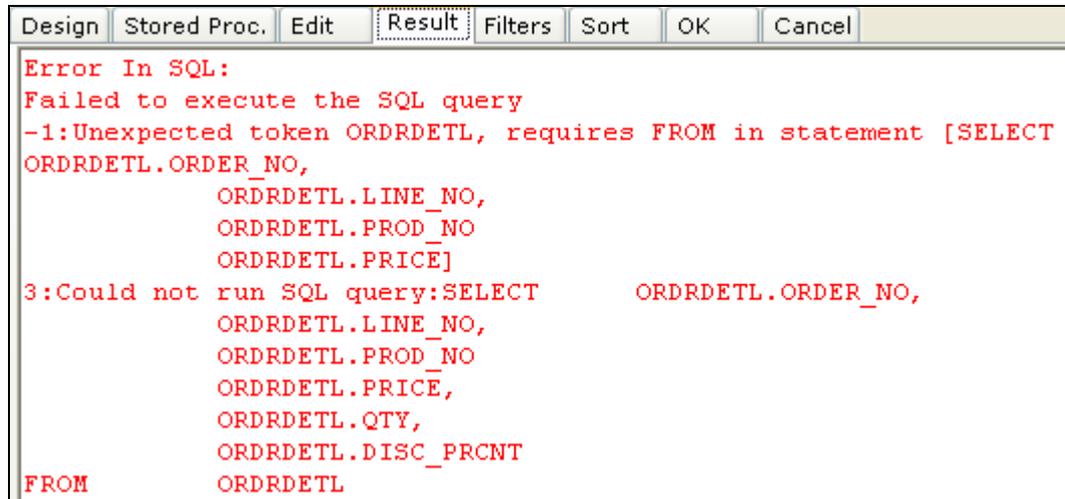


Figure 9: SQL Compilation

If the SQL has used a parameter, you will be prompted to provide the values of the parameter.

## Viewing SQL Results

The **Result** tab will display the result based on the SQL statement in the **Edit** tab. If the SQL statement has used a parameter, you will be prompted to provide the values while viewing the results.



The screenshot shows the SQL Editor interface with the 'Result' tab selected. The text area displays a table of results:

	ORDER_NO	LINE_NO	PROD_NO	PRICE	QTY	DISC_PRCNT
▶	1	1	60401	9	617	21
	1	2	60400	6	247	7
	2	1	60101	12	124	14
	2	2	60400	6	115	13
	2	3	50101	12	32	21
	2	4	40101	15	2	1
	2	5	40100	16	34	1
	2	6	40302	18	1	20
	3	1	60400	6	504	11
	3	2	60301	11	50	4
	3	3	50203	8	32	6

Figure 10: SQL Results

## Applying Filters

At report run time, a user may wish to receive the data that makes sense to him/her at that time. On this tab, you can set filters so that user can provide filter values at report run time.

×	FieldName	Type	DataType	Parameter	FetchData	Criteria	Value1	Value2
	PRODUCT.SALES_92	ADHOC	NUMBER		<input type="checkbox"/>	Between	1000	2000
	PRODUCT.PROD_LINE	ADHOC	CHAR		<input type="checkbox"/>	Is	EarPhone	
*	PRODUCT.PROD_TYPE	ADHOC	CHAR		<input type="checkbox"/>	Is	Mobile Device	

Figure 11: Applying filters on SQL

## To add filters

Select a field by clicking it (Press and hold Ctrl key and click fields to select multiple fields) to from Available Fields. Click > button to move the fields to Selected Fields list.

At report run time, Adhoc tab of Input Parameter Form will have 3 rows having all of the selected fields as as dropdown.

Use >> to move all the fields to Selected Fields list.

To un-select a field, click the field from Selected Fields list and click < button. Click << button to empty Selected Fields list.

- **Qualifier:** When SQL is typed in Edit tab, fields may not be prefixed with table name and so, Available Fields list also don't have table names prefixed. Select a field and specify table name to associate it with a table. This is especially useful when you get the same field name from two tables and wish to differentiate one from another.
- **Max Rows:** The maximum number of records to be fetched from the database. Number specified here would appear by default on input parameter form at run time. User can change it if required at run time.
- **Suppress Duplicates:** Check this checkbox to get only one record if record has multiple records having completely duplicate values.

---

## To setup default value for filter

1. Select the field name from the table.
2. Specify / select values the selected field from other columns. Details are provided after these steps.

- **FieldName:** The field name for which default value(s) is being set.
- **Type:** Select Adhoc to use the fieldName for accepting the value. Select UserParameter get value using a pre-set user parameter.
- **Data Type:** Specify data type of the filter parameter.
- **Parameter:** If Type is UserParameter, specify the parameter name.
- **FetchData:** Select this check box for the field if filter type is Adhoc and you want data should be fetched from the database. In this case value box for this field will appear as a drop-down box.
- **Criteria:** Select criteria for the field.
- **Value1, Value2:** Based on selected criteria, specify value1 or value1 and Value2. For example, for "Between" as criteria, you need to provide two values.
- **Mandatory:** Select this checkbox if it is mandatory for user to provide value for this field at runtime.

The screenshot shows the 'Report Parameters' page for a report named 'ProductList'. At the top, there are buttons for 'Run Now' and 'Reset', and a dropdown menu for '(Select Form)'. Below this is a 'Select Filter Criteria' section with a '3 Selected' indicator. The section includes a 'Max. Rows' input field and a 'Suppress Duplicates' checkbox. The main area contains a table with three rows of criteria:

Field	Criteria	Value
PRODUCT.PROD_TYPE	Is	Mobile Devices
PRODUCT.PROD_LINE	Is	EarPhone
PRODUCT.SALES_92	Between	1000 and 25000

Figure 12: Report parameters page during report run time

## Sorting at Run Time

This tab is used to specify levels of sort at run time.

Field Name	Sort	
PRODUCT.PROD_TYPE	ASC	X
PRODUCT.PROD_LINE	ASC	X
*		X

Figure 13: Options for sorting at runtime

- **Prompt:** Select this check box to get sort dialog box at report-run time. User will be able to provide sorting options on the dialog box.
- **Count:** Select the number of sort levels to be made available to user. For example you want user to be able to provide three sort choices at run time (e.g. by Country, State and county), select 3.
- **Qualifier:** When SQL is typed in Edit tab, fields may not be prefixed with table name and so, Available Fields list also don't have table names prefixed. Select a field and specify table name to associate it with a table. This is especially useful when you get the same field name from two tables and wish to differentiate one from another.
- **Disable Forced Sorting:** If due to any reason, result-set received from data-source is not sorted as per need (like groping), Intellicus server will sort the data. Check this checkbox to stop Intellicus from doing so.



**Important:** Data sorting performed by Intellicus becomes significantly time-intensive process when a record-set has large volume of records. In that case, we recommend not to **Prompt** and check **Disable Forced Sorting**. This will make sure Intellicus uses data as it is received from data-source.

The fields listed in **Available Fields** are the fields that will come from database. The fields listed in **Selected Fields** will be available for sorting at run time.

- To move a field from **Available Fields** list to **Selected Fields** list, select a field from **Available Fields** list and click > .
- Click >> to move all the fields to **Selected Fields** list.

- To move a field from **Selected Fields** list to **Available Fields** list, select a field from **Selected Fields** list and click < .
- Click << to remove all the fields from **Selected Fields** list.

Based on the number selected in **Count** check box, that many dropdown boxes will appear on an input form at the run time. You can select fields that should appear by default along with default order (ascending or descending).

**In Default Fields** select the field, and the sort order. Click 'x' button to delete that row. Pressing enter key in last row will append a row at the bottom. Select a row and Use Up or Down arrow button to modify the sequence of field-appearance.

## Removing Tables from SQL Editor

To remove a table from the SQL Editor, right-click the table and select **Delete Table**. If you have defined a join on the tables, you need to first delete the join.

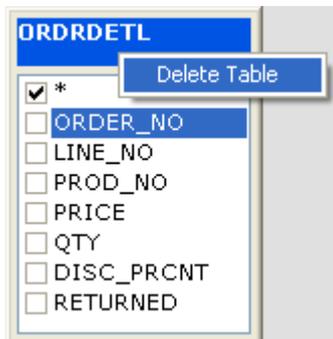


Figure 14: Delete Table

## Removing a Join

To remove a join condition, select a join condition by clicking  under the *where* clause and press <delete> on the keyboard.

Where	Open	Operand1	Operator	Operand2	Close	Relation
		ORDRDETL.PROD_NO	=	PRODUCT.PROD_NO		
						

Figure 15: Removing a Join

The Join will be removed.

