

Summary

This article attempts to de-mystify what queries are, how and why they are used, and to provide insights into how these queries can be executed. An appendix includes useful query examples.

Queries can be used to filter specific objects in Schematic Library, Schematic, PCB Library and PCB documents and then can be used for editing tasks. For this article, any reference to a document refers to those particular documents (unless stated otherwise).

Understanding Queries

Queries are used to determine whether each object within a document is **highlighted** or otherwise. The highlighted objects are normally displayed which is distinctive in comparison with the remaining (non-highlighted) objects, which are either displayed in a less prominent manner, or are not even displayed at all.

Queries can be used to control how a document's objects are displayed. They can also be used to determine whether any objects within a document have particular properties (or particular sets of properties), and to assist in locating such objects. Another important reason for using queries is to qualify which objects have their properties modified during succeeding commands, including during global editing commands.

Which objects within a document are subsequently highlighted following the query application are determined by that query, and in turn is controlled by the designer. The designer also determines whether each of the **Zoom**, **Select**, and **Mask** highlighting options (described in more detail later) are selected with each query, which in turn determine the subsequent selected and masked states of each design object, and how the highlighted and non-highlighted design objects are displayed.

Queries can be applied from *Find Similar Objects* or *Building Query* dialogs, or from the *Filter* panel, or from the *Filter* Toolbar (in the case of PCB documents), or from resources (menu entries or toolbar buttons or shortcut keys).

There are ways to control the query and a query statement consists of a string that always contains at least one keyword. Designers can specify the query statement themselves. A query statement can alternatively be selected from lists of favorite or previously used queries, or automatically generated, or pre-determined by the designer.

Queries are also used for two special purposes within a PCB document; they specify which objects are subject to each defined Design Rule, and they specify the Impedance Formulae that are to be used whenever the Characteristic Impedance Driven option is selected for any (Max-Min) Width Rule.

Data Filtering – facilitating editing tasks

A typical design document contains a number of objects, thus designers are provided with the ability to display each of those objects in more than one way, and a degree of control over which objects are currently displayed. Because some commands change the properties of multiple objects, designers are also provided with the ability to control which objects that do have their properties modified.

This capability to display and modify objects on a selective basis is implemented by a sophisticated **data filtering** feature. Whenever it is used, the properties of each object in the target document are evaluated, and those objects having properties which comply with the specifications of the filtering operation become members of that filtering operation's **result set**. The combination of which **highlight** options were also specified for the filtering operation concerned, and whether each object is a member of that result set or otherwise, determines the subsequent selected and masked properties of each object, and the manner in which it is subsequently displayed.

An example, invoking a Zoom command then invokes the data filtering feature, producing an outcome in which the document's objects are displayed on the screen on a selective basis. The associated result set consists of those objects which are located within the updated region of the document being displayed, and the data filtering feature causes those objects to then be displayed. Conversely, the objects which are located outside the updated region being displayed are not in the result set, and the data filtering feature causes those objects to *not* then be displayed¹.

¹ Even when *all* of the document's objects are displayed following a Zoom command, the data filtering feature has still been invoked: the associated result set then consists of all of those objects.

Another example is that invoking any command which changes the selected state of one or more objects within a document also invokes the data filtering feature, producing a result set of the now currently selected objects. Selected objects and unselected objects are displayed in different ways, so the data filtering feature also controls how each object is subsequently displayed. When a command is invoked to be applied to the currently selected objects, the data filtering feature also controls whether each object has that command applied to it or otherwise.

Another example, the data filtering feature is also used in conjunction with the masking feature. It controls how unmasked objects and masked objects are each displayed, and whether various types of commands are applied to each object within a document.

The data filtering feature is invoked whenever a query is applied. The query statement determines which objects within the target document become members of the associated result set, and the **highlight** options which were selected at the time determine the subsequent states of each object's selected and masked properties, and the manner in which each object is subsequently displayed.

The ability to specify the highlight options, and to control (with a fine degree of precision) the data filtering feature's result set, makes queries so useful to designers.

Data Views and Highlighting – What is displayed, and How

To fully appreciate how to specify and apply queries, it is necessary to understand the concepts of the distinct **data views** which are provided, and of the **highlighting** options which are available when queries (and other types of filtering commands) are applied.

In a nutshell, *what* is displayed is determined by the data view, and *how* different objects are displayed within each data view is determined by the available highlighting options.

Graphical View and List View – two distinct Data Views

Traditionally, a *graphical view* displays these design objects and their properties on a document in a spatial and visual manner.

The *list view* now provides designers with an alternative way of viewing those objects and their properties in a tabular format. Each row within this data view's array displays the properties of one object, and each column displays the value of one particular type of property (as indicated by the top-most Title row) for each of the currently listed objects. It is possible to control which properties (columns) are currently displayed and the sequence in which they are listed; it is also possible to sort the sequence in which the objects (rows) are listed, as determined by the values of one, or more, of their properties.

Within the graphical view, selected objects are displayed in a distinctive manner relative to unselected objects, and the selected state of each object can similarly also be changed from the list view.

The list view facility enables two distinct types of *data views* which are available for viewing the properties of a document's objects. Each type of data view displays those objects' properties in different ways, but they are both displaying the properties of the *same* objects. Thus any changes which are made to any object's properties from either of the data views always results in corresponding changes in how the object is displayed in the other data view. For example, if the selected state of an object is toggled from either of those data views, then the updated state is displayed as such not just in that data view, but in the other data view as well.

The list view is contained within the *List* panel, which is capable of being re-sized and re-positioned. The **Shift+F12** key toggles the *List* panel's visible state. This panel has various controls that determine which objects are listed within it, and whether the properties of those objects can only be viewed from this panel or also edited from it as well.

Zoom, Select, and Mask – three distinct Highlighting options

Whenever a query is applied (or the data filtering feature is otherwise used), each object within the target document becomes a member of that filtering action's result set. However, *how* the objects in the result set, and the objects which are not in the result set, are subsequently displayed depends upon the Highlighting option(s) selected at the time. There are three different highlighting options available, as described below:

Zoom

When this Zoom option is selected, the graphical view of the target document is updated, with the updated view displaying the region occupied by all of the objects which are members of the result set. Zoom commands change the graphical view of documents.

Whether or not each remaining object is displayed in the updated view depends upon its location relative to that region, so each of those objects can end up being totally displayed, or partially displayed, or not even displayed at all.

This option has been provided primarily for updating the target document's graphical view, which displays spatial properties and relationships. But it would be incorrect to state that it has no impact upon the target document's list view; there can also be changes to the set of objects which subsequently have their properties displayed in the list view, and which of those objects are selected.

Select

When this Select option is selected, all of the objects which are members of the result set acquire a (true) selected state, while all of the remaining objects acquire an unselected state. Selected objects are displayed in a more distinctive manner than unselected objects in both the graphical view and list view.

Mask

This highlighting option determines the updated *masked* property of objects in the target document. When selected, all of the objects which are members of the result set acquire an unmasked state, while all of the remaining objects acquire a (true) masked state.

The masking feature is when unmasked objects are displayed in an unqualified (normal) form in both the graphical view and list view. On the other hand, masked objects are displayed in a dimmed form in the list view, and are not displayed in the list view unless the (*List* panel's) *all objects* option is currently selected. The main aspect of masked objects is that they and their properties are not capable of being edited.

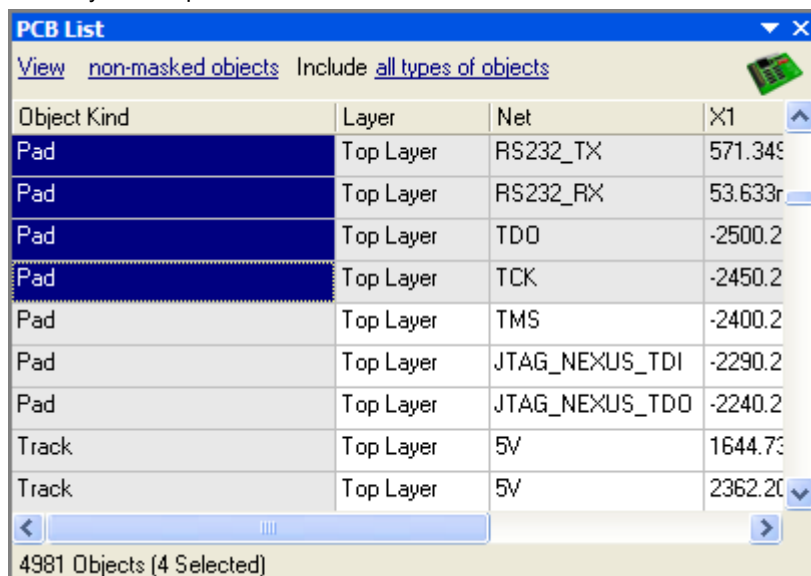
Which options can be selected are totally independent of one another, so any one of those three options can be selected, or any two of them, or all three of them. (It is not impossible to alternatively select *none* of those options, but there would be no difference between how highlighted objects and non-highlighted objects were subsequently displayed.) The purpose of applying a query determines which highlighting options should be selected with it.

The List panel – another data view

Queries can be specified and applied from the *Filter* panel, and depending upon the particular options, the application of a query also determines which objects within a document are subsequently listed (or listed and selected) within the *List* panel.

The *List* panel incorporates the document's list view, which lists the properties of (a subset of) the document's objects in a tabular format. The list view and graphical view both display properties of the same objects, but in different forms, so while some tasks can be undertaken from either view, other tasks are better undertaken from one of these views.

One example of a task that is better undertaken from the graphical view is moving one or more objects; it is easy to see where those objects are being moved to, relative to all of the other objects. Another example of a task that is better undertaken from the list view is inspecting or editing the properties of objects which are currently not displayed in the graphical view, such as currently hidden parameters in Schematic documents.



The screenshot shows a window titled "PCB List" with a toolbar containing "View", "non-masked objects", and "Include all types of objects". Below the toolbar is a table with the following data:

Object Kind	Layer	Net	X1
Pad	Top Layer	RS232_TX	571.349
Pad	Top Layer	RS232_RX	53.633r
Pad	Top Layer	TDO	-2500.2
Pad	Top Layer	TCK	-2450.2
Pad	Top Layer	TMS	-2400.2
Pad	Top Layer	JTAG_NEXUS_TDI	-2290.2
Pad	Top Layer	JTAG_NEXUS_TDO	-2240.2
Track	Top Layer	5V	1644.73
Track	Top Layer	5V	2362.20

At the bottom of the panel, it says "4981 Objects (4 Selected)".

The List panel incorporates the document's list view, and can thus be used to view and edit the properties of the document's objects.

The Zoom commands typically change which region of a document's graphical view that is currently displayed. Those commands are not applicable to the document's list view, but the designer instead has the ability to control the sequence in

which the objects are listed in the list view; these can be sorted by the values of one, or more, of their properties. And by applying queries while the Mask highlighting option is selected, the designer can control exactly which objects are listed in the list view.

By specifying a query, the list view can thus be used to determine if there are any objects in the document having a particular property, or combination of properties. An example is determining if a document contains any Designator or Comment (Text) strings which are currently in a hidden state.

Another important aspect of the *List* panel is that (like the *Inspector* panel) it supports global editing, in which the values of a particular type of property of all the currently selected objects are updated simultaneously².

The *Filter* panel does not provide the only way of specifying and applying queries, but it is still very useful for designers who want to learn how to specify these, as the list view provides instant feedback on the outcomes of applying various queries.

Clicking on the **Clear** button (near the bottom right hand corner of Altium Designer) effectively “clears” or “resets” the target document; all of the objects acquire both an unselected and unmasked state. It is very useful to do this whenever the designer wants the list view to display the properties of *all* of the document’s objects.

For designers who are totally new to specifying their own queries, but who want to learn how, the **Builder** button within the *Filter* panel should be clicked, to invoke the **Query Helper** feature. The left hand section in the resulting *Building Query* dialog contains controls, whose purpose is to assist the designer in the task of specifying what properties are required for each of the document’s objects to be returned by the generated query by that dialog.

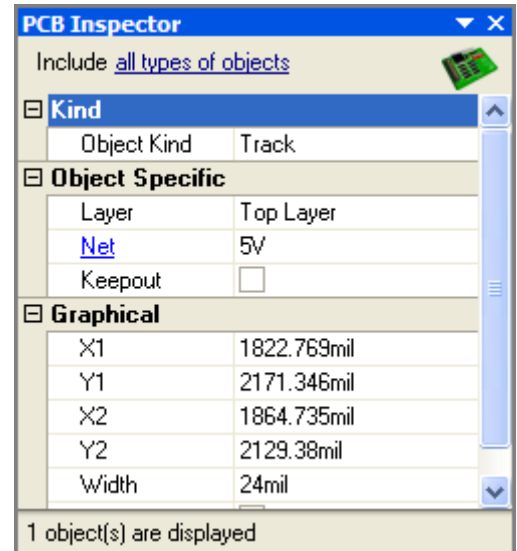
The contents of the corresponding query are updated and displayed in the dialog’s right hand section. If this dialog is then closed by clicking on its **OK** button, those contents will then be copied back to the *Filter* panel. That query can then be applied by clicking on the **Apply** button within this panel, but that should be done only after specifying the Highlighting options.

The **Helper** button within the *Filter* panel could then be clicked on, to invoke the *Query Helper* dialog. The designer is then fully responsible for specifying the query statement, but this dialog provides a Syntax-checking feature and a complete list of keywords (for the type of currently focused document).

Pressing the **F1** key on one of these highlighted keywords obtains information from the on-line help. If this dialog is closed by clicking on its **OK** button, the query statement is then copied back to the *Filter* panel.

The *Filter* panel also contains **History** and **Favorites** buttons. These invoke the *Expression Manager* dialog, which contains a list of previously applied queries and a list of favorite queries. A query can be selected from either of those lists and then applied, and queries can also be copied from the former list to the latter list.

In the case of PCB documents, a **Create Rule** button is also provided in the *Filter* panel. When the designer is satisfied that the result set of the current query matches the requirements of a Design Rule then clicking on that button creates the targeted Design Rule.



The Inspector panel – property viewing and editing tool

The list view from the *List* panel displays the properties of the target document’s currently unmasked objects in a tabular format. The *Inspector* panel also displays the properties of objects but in a condensed form.

This panel displays a “summarized” list of the properties of the target document’s currently selected objects. Its left hand column lists all of the types of properties which are common to all of those objects, while its right hand column indicates whether or not all of those objects have the same value for each of those types of properties. When a particular value is displayed, all of those objects have the same (displayed) value for that particular property, while the absence of the display of any such value alternatively indicates that those objects do *not* all have the same value for that particular property.

²**Important Note:** The selection state property is the *only* property which now determines whether each object within a document has some other property updated during global editing procedures; unlike in previous versions of Protel, it is no longer possible to use the current values of *other* properties to specify which objects are to have their properties updated during global editing procedures.

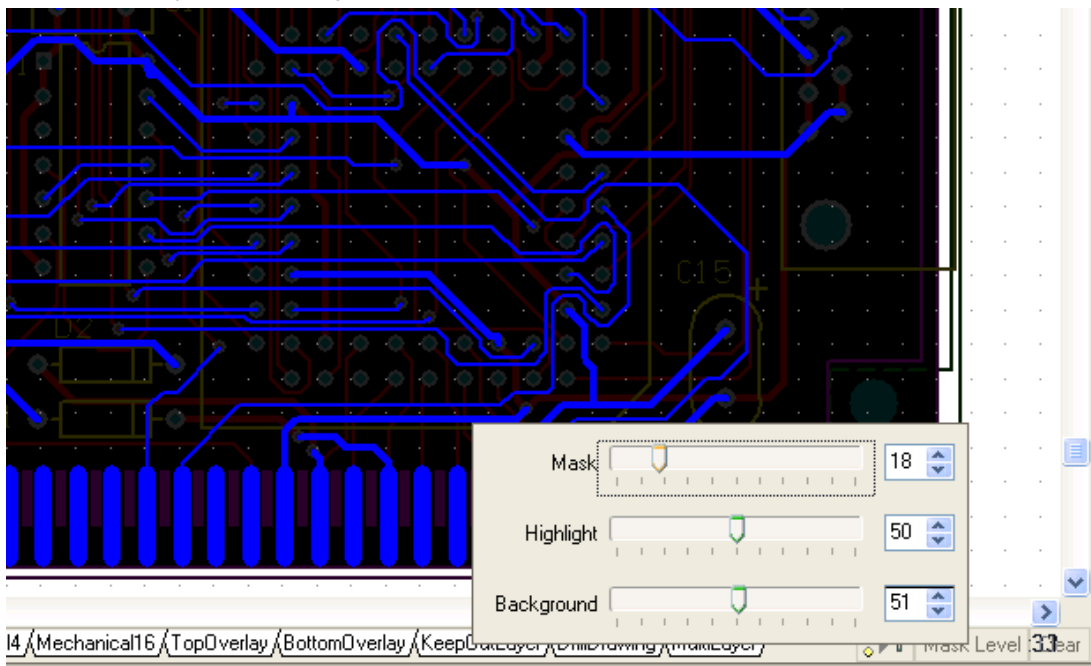
The designer is able to determine which property values are common to all of those objects, and which (remaining) property values are not.

By its nature, this panel can be very useful to have displayed during global editing procedures, which are also only ever applied to the target document's currently selected objects. Designers can see which property values are common to these design objects, and both before and after each type of commonly shared property is globally updated.

Masking – another highlighting option

Designers have the ability to select which objects within a document are currently in a *masked* state. Masked objects are not capable of being selected or of otherwise having their properties edited in any way. Unlike unmasked objects, masked objects do not have their properties displayed in the list view (unless the *List* panel's *all objects* option is currently selected). In a document's graphical view, unmasked objects are always displayed in a totally undimmed form.

To distinguish them from unmasked objects, masked objects can be displayed in a dimmed form, with an associated **Mask Level** control (invoked by clicking on the **Mask Level** button near the bottom right corner of the Altium Designer) controlling how much the display of masked objects is dimmed.



The Mask Level control is used to control how much the display of currently masked objects is dimmed.

The masking feature's power and usefulness is a consequence of the designer having the ability to control which objects in a document are currently masked. One example of how the masking feature can be usefully deployed is to enable a designer to edit just some objects, while the remaining objects are disabled from having their properties edited, but continue to be displayed. Because the masked objects are still displayed, the designer can see where these are relative to the unmasked objects.

In the previous illustration, the only unmasked objects are those on bottom side layers, but all of the objects in the document are still displayed in a dimmed state.

However, the masking feature is far more powerful than just enabling the designer to select which layers are masked, or which types of objects are masked. Because masked objects cannot be selected, designers can also control which objects will be selected when the **Edit » Select » All** command is invoked.

Applications of Queries

When a query is applied to a document, the properties of the design objects within are evaluated, to determine whether its properties comply with the specifications defined by the query's contents. The thus-compliant objects become members of that query's result set, and the combination of the highlighting options also selected at the same time, together with whether each object is a member of the result set or otherwise, determines the updated selected and masked properties of each object, and the manner in which each object is subsequently displayed in each of the document's data views.

Finding, Presenting and Editing Objects

Three distinct highlighting options are available at the time that a query is applied, and it is possible for each of these options to be selected totally independently of the others. At least one of these options is always selected and designers could select two or all of the options.

A brief summary of the nature of each highlighting option follows;

Mask option

If the Mask option is selected when the current query is applied, the objects which are not members of the result set subsequently acquire a masked state, which prevents them from being selected or otherwise edited; the objects which are members of the result set subsequently acquire an unmasked state, and so remain capable of being selected and edited.

The designer also has the ability to control how masked objects are displayed in the graphical view.

The Mask option can be used to prevent particular objects from having any of their properties changed, or to inhibit the display of these. (As described later, hierarchical considerations can sometimes result in some masked objects remaining in an editable and fully displayed state.)

All of the unmasked objects are listed in the list view, whereas masked objects are *not* listed there³. Another reason for selecting the Mask option is to determine whether there are any objects in the document having particular properties (or combinations of properties). If any such objects do exist, they will become members of the result set of an appropriately specified query, and will thus be displayed in the list view. Furthermore, *just* those objects will be displayed there, as all of the remaining objects will not be members of the result set (and thus won't be displayed there as well).

Select option

If the Select option is selected when the current query is applied, all of the objects which are members of the result set subsequently acquire a selected state, while all of the remaining objects (which are not members of the result set) acquire an unselected state instead.

This option would typically be selected just prior to global editing procedures, because only those objects which are currently selected have their properties updated at the time. But there are other occasions when the Select option can be useful, such as when particular objects are to be moved, copied or deleted.

Zoom option

If the Zoom option is selected when the current query is applied, the graphical view of the document is updated. The updated view is one whose boundaries *just* enclose *all* of the objects which are members of the query's result set.

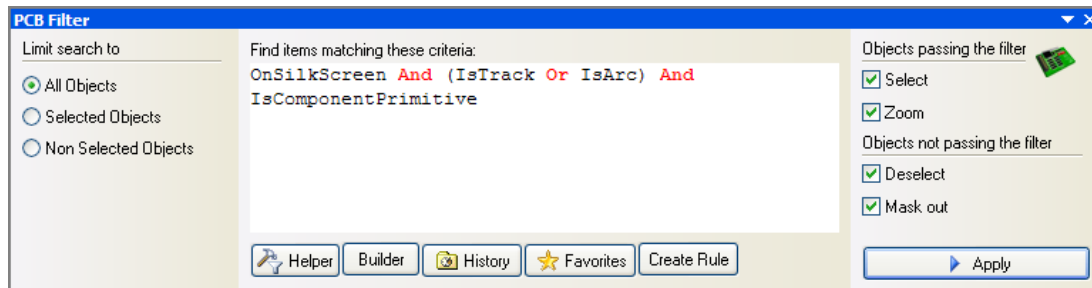
This option would often be selected in conjunction with one or both of the other options; as the objects which are not members of the result set would either be unselected or masked (or both) at the time, designers would often want the graphical view of the document to be updated so that only the objects which have acquired a selected or unmasked state (or both) are displayed in the updated graphical view.

If the Mask option has *not* been selected, objects which are not members of the result set will still be displayed in the graphical view, if they are located within the updated boundaries of this. At first glance, selecting this option by itself might seem pointless, but there could still be times when a designer wants to update the graphical view so that this encompasses particular objects, while not changing the selected state or masked state of any of the document's objects.

³ Hierarchical considerations qualify that situation (again). Some queries retain particular parent objects, while simultaneously filtering out some or all of their child objects. Although not selected by default, an option is available to restore otherwise filtered out child objects (of retained parent objects) to the list view.

Applying Queries

Queries can be applied from the *Filter* panel, or from *Find Similar Objects* dialogs, or from *Query Builder* dialogs, or from the *Filter* toolbar (in the case of PCB (document) documents), or from resources (menu entries or toolbar buttons or shortcut keys). When the *Find Similar Objects* and *Query Builder* dialogs are used, the associated queries statements are automatically generated. This is helpful for those who have yet to master how to specify queries for particular tasks, especially as it is also possible for the contents of those queries to be displayed in the *Filter* panel.



The Filter panel is used to type in a query.

The *Filter* panel is mainly used for designers to build query statements and apply them. In many cases the updated contents of the list view will be of interest, which is a good reason for also having the *List* panel displayed at the time.

There is a much lower level of assistance provided for keying in queries in the *Filter* Toolbar, and the highlighting options it uses are fixed, but it does provide a list of most recently used queries, from which a query can be selected for re-application.

Query Options

When a query is applied, there are options that can be specified at the same time. Three options (Mask, Select and Zoom options) are the highlighting options, but normally at least one other option also needs to be specified.

Mask option

When specified, the objects in the result set are placed in an unmasked state, while the remaining objects are placed in a masked state. Objects in a masked state are not displayed in the document's list view, and are displayed in a dimmed manner in the graphical view. They are also not capable of being selected or having any of their other properties modified.

Select option

When specified, the objects in the result set are placed in a selected state, while the remaining objects are placed in an unselected state. Objects in a selected state are displayed in a distinctive manner in both the document's graphical view and list view, and have their properties updated during global editing commands.

Zoom option

When specified, the objects in the result set are displayed in the updated view of the document's graphical view; whether each of the remaining objects is displayed depends upon its location relative to the updated region that is displayed in the document's graphical view. Whether each object is displayed in the list view depends upon whether various other options were selected.

It is possible to specify that (any) one of those outcomes be selected, or that (any) two of these be selected, or that all three of these be selected. Which outcomes are specified for a particular query will depend upon why the query is being applied, and whether the designer wants to view and edit objects from the document's graphical view or from the list view (or from both of these).

Clear Existing option

This option can usually⁴ also be specified whenever a query is applied. When selected, the document is always "cleared" before the query is applied. (The outcome of "clearing" a document is that every object in the document acquires both an unselected

⁴ This option can be specified when the query is being applied from *Find Similar Objects* or *Building Query* dialogs, or from resources. When the query is being applied from the *Filter* Toolbar (in the case of PCB documents) or from the *Filter* panel though, the document is *always* cleared previously.

state and an unmasked state.) When not selected, any objects which were *already* displayed in the manner matching how objects in the query's result set are *consequently* displayed are unconditionally added to the query's result set, implying that such objects *remain* displayed in the same manner following the query's application.

While this option does enhance the capabilities of queries, to not select this option typically results in outcomes which are unexpected by designers with less experience; hence such designers should always opt for selecting this option.

Current Component / All Components option (Schematic Library documents)

In the cases of Schematic Library documents, designers can additionally select whether the query returns compliant objects in just the currently focused part, or whether it returns compliant objects in all of the parts contained in the target document.

Whole Library option (PCB Library documents)

In the cases of PCB Library documents, designers can additionally select whether the query returns compliant objects in just the currently focused footprint, or whether it returns compliant objects in all of the footprints contained in the target document.

Current Document / Open Documents / Open Documents of the Same Project option (Schematic documents)

In the case of Schematic documents, designers can additionally select whether the query returns compliant objects in just the target document, or whether it returns compliant objects in all Schematic documents which are currently open, or whether it returns compliant objects in all Schematic documents associated with the target (PCB) Project which are currently open.

Run Inspector (Find Similar Objects and Query Builder dialogs)

When a query is being generated from *Find Similar Objects* or *Building Query* dialogs, designers can additionally specify whether the *Inspector* panel is to be consequently displayed. When this option is selected, that panel will always consequently be displayed, and regardless of whether it was displayed previously or otherwise. When this option is not selected, the displayed state of that panel does not change.

Create Expression (Find Similar Objects and Query Builder dialogs)

When a query is being generated from *Find Similar Objects* or *Building Query* dialogs, designers can additionally specify whether an associated expression is also generated or not. When this option is selected, an expression is generated, and this is listed in the *Filter* panel, which is also consequently displayed. When this option is not selected, no expression is generated or listed in the *Filter* panel, and the displayed state of that panel does not change.

Finding Similar Objects – facilitating global object editing

A longstanding feature is the ability to edit the properties of multiple objects simultaneously. As one example from Schematic documents, designers can change the Color property of all Power Objects having a particular Text property (e.g. 'GND') within a document to the same value, and from just one editing procedure. And as one example from PCB documents, designers can similarly change the Hole Size property of all vias having a particular Via Diameter property (e.g. 30mil) within a document to the same value, and again from just one editing procedure.

However a relatively recent change in the implementation and usage of global editing procedures is that they are now always applied to whichever objects are currently selected.

The implications are that it is first necessary to select whichever objects are to have their properties changed during the subsequent global editing procedure, and that it is also very important to select *all* of the correct objects, and *just* those objects, before undertaking that global editing procedure.

In some situations, the appropriate objects could be selected by the user without having to apply any query. However, in other situations, the potential exists for some objects to incorrectly remain unselected, and / or for some objects to incorrectly be selected, and as such, it would then be desirable to apply an appropriate query, in order to insure that the correct objects are in fact subsequently selected.

As such, a **Find Similar Objects** feature has been provided, and this *automatically* generates a query whose contents produce an outcome matching that desired by the designer.

This feature is invoked by positioning the cursor over one of the objects that is to have its properties modified, and then selecting the **Find Similar Objects...** item from the (right mouse button) popup menu. (Alternatively, the menu item of **Edit » Find Similar Objects** could be selected, or the key sequence of **Shift+F** could be invoked. The cursor then acquires a "busy" state, and one of the objects to have its properties modified should then be clicked on.)

A *Find Similar Objects* dialog is subsequently invoked, and the designer then selects which types of properties need to have same values (as the just clicked-on object). By default, the 'Object Kind' property of the other objects has to have the same value, but typically other types of properties need to be similarly matched as well.

When the objective of using this feature is to select objects for global editing purposes, the **Clear Existing** and **Select Matching** checkboxes should both be checked. Global editing procedures are actually performed from either the *Inspector* panel or the *List* panel, and the former panel is subsequently displayed if the **Run Inspector** checkbox is also checked at the time.

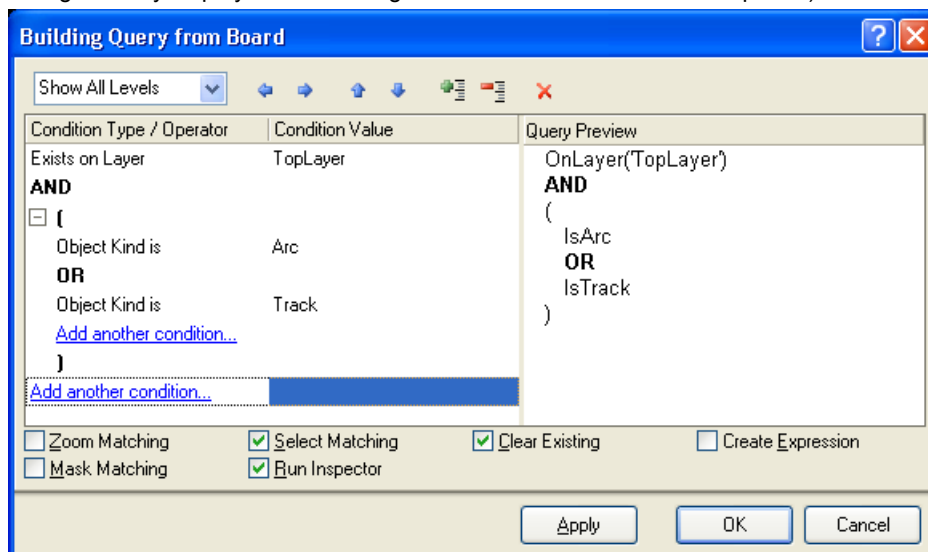
When the **Create Expression** checkbox is checked, the *Filter* panel is subsequently displayed, and the contents of the associated query are displayed in that. Although the queries generated by the **Find Similar Objects** feature are relatively unsophisticated in nature, studying their contents still provides designers with insights into how to specify their own queries.

Designers who have acquired all that they can from studying those queries will typically either not continue to use the **Find Similar Objects** feature, or else will alternatively select the options of checking the **Run Inspector** checkbox, but not the **Create Expression** checkbox. With that set of options, the query generated by the feature is no longer loaded into the *Filter* panel, but the *Inspector* panel is subsequently displayed, which facilitates implementing the succeeding global editing procedure(s).

The Query Builder feature – automating query generation

Although the **Find Similar Objects** feature is useful in many circumstances, the power of the queries which this generates is still rather limited in nature. Thus the **Query Builder** feature permits queries to be automatically generated based upon the options and specifications selected by the designer in the associated *Building Query* dialog.

That dialog can be invoked by clicking on the **Builder** button within the *Filter* panel. It can alternatively be invoked by selecting the menu item of **Edit » Build Query...**, or the key sequence of **Shift+B**. (The checkboxes and the **Apply** button within this dialog are only displayed if the dialog was *not* invoked from the *Filter* panel.)



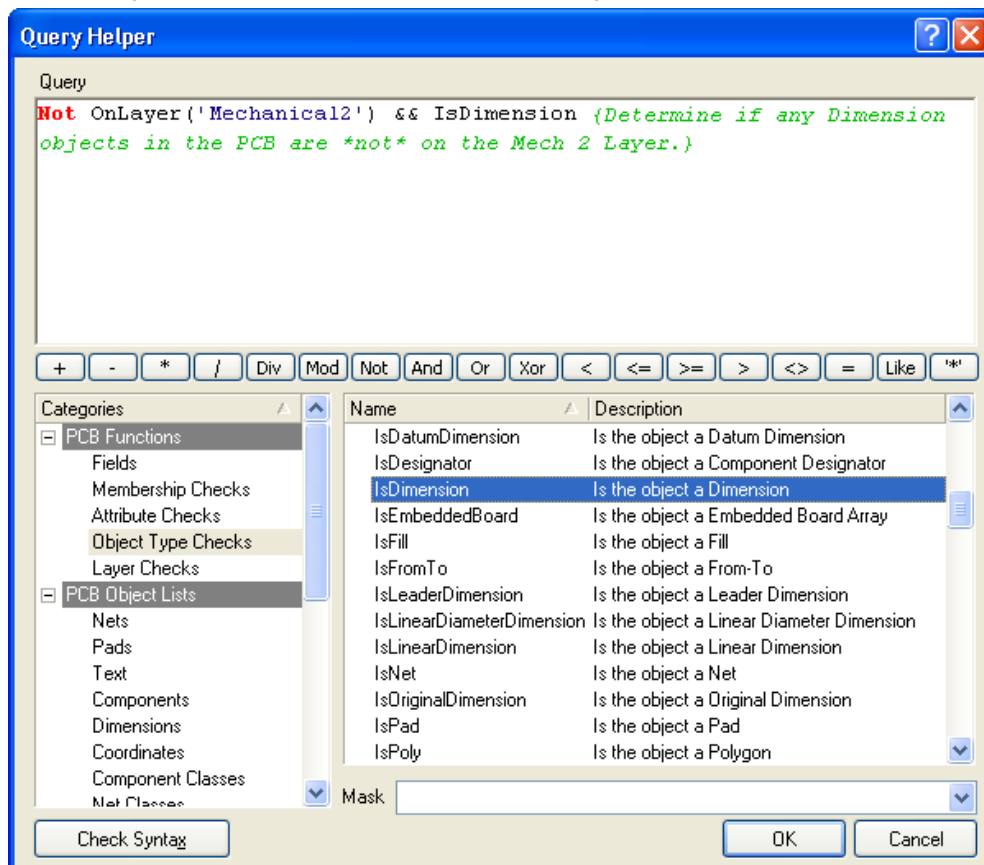
The 'Building Query' dialog (part of the 'Query Builder' feature) assists those who are less experienced in specifying queries; a query is automatically generated whose contents correspond to the conditions specified by the designer.

The left hand section in this dialog contains controls, whose purpose is to assist the designer in the task of specifying what properties are required for each of the document's objects to be returned by the query generated by this dialog. As each of those conditions are specified or edited, the contents of the corresponding query are updated and displayed in the dialog's right hand section. If this dialog is then closed by clicking on its **OK** (or **Apply**) button, either the associated query will then be applied, or its contents will then be copied back to the *Filter* panel (depending upon how this dialog was invoked in the first instance).

The Query Helper dialog – support for designer-specified queries

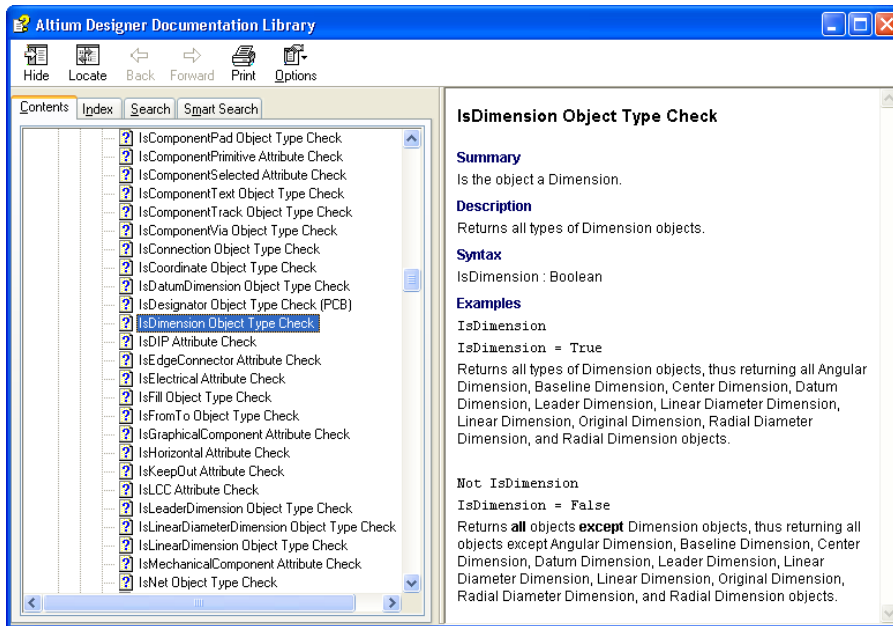
It is possible to enter queries directly into the *Filter* panel, and the designers can take advantage of the associated “keyword popup” feature, which facilitates the entry of recognized keywords. They can still use the *Query Helper* dialog whenever they attempt to specify their own queries.

That dialog is invoked by clicking on the **Helper** button within the *Filter* panel. (It can alternatively be invoked by clicking on the **Query Helper...** button within the *PCB Rules and Constraints Editor* dialog, or by clicking on one of the **Helper...** buttons within the *Impedance Formula Editor* dialog.) This dialog provides assistance to the designer, as it incorporates a Syntax-checking feature and a complete list of valid keywords (for the type of currently focused document). If this dialog is closed by clicking on its **OK** button, the contents of the query within it are then copied back to the *Filter* panel (or to the *PCB Rules and Constraints Editor* dialog, or to the *Impedance Formula Editor* dialog).



The ‘Query Helper’ dialog provides assistance for designers who want to specify their own queries.

A brief description is provided for each keyword that is listed, but the on-line help can be accessed by pressing the **F1** key while a keyword is highlighted. That invokes the *Altium Designer Documentation Library* dialog, which provides details of which objects within a document are returned by the highlighted keyword, how to use that keyword, and one or more examples of its usage.



The Online Help can be accessed by pressing the F1 key while the 'Query Helper' dialog is displayed. Comprehensive details are provided on how each available keyword can be used within queries.

Design Rules in PCB Documents

Queries do have to be used to specify the Scope(s)⁵ of each Design Rule that is defined in a PCB document; the query's result set defines which objects within the PCB document that the Design Rule concerned applies to.

Assistance is available to help designers in specifying these queries. The *Query Helper* and *Building Query* dialogs can both be invoked from the *PCB Rules and Constraints Editor* dialog, which also supports automatic generation of the query contents for some of the simpler (but frequently used) Scopes. That dialog also contains a **Rule Wizard ...** button, which invokes a *New Rule Wizard* dialog, providing designers with assistance when it comes to specifying Design Rule Scopes.

The *Filter* panel contains a **Create Rule** button. The designer can evaluate the result sets of queries from the *List* panel, and once satisfied that the result set of the currently applied query matches the requirements of a Design Rule, click on this button to create the Design Rule.

In general, the result set of Design Rule queries is the *primitive* objects rather than the *group* objects. The primitive objects are of a "child" nature, and include arcs, component bodies, fills, pads, regions, (text) strings, tracks, and vias. The group objects are of a "parent" nature, and include components, coordinates, dimensions, nets, and polygons. Thus a Clearance Rule which specifies a customized clearance between polygons and other objects requires a query whose result set includes the "child" objects of polygons, rather than of the "parent" polygon objects.

Thus an appropriate query to specify is `InPolygon` or `InPoly` (which returns "child" objects of polygons), rather than `IsPoly` (which returns "parent" polygon objects).

Specifying Impedance Formulae

In Width Design Rules, designers can specify either the Width constraints, or specify Characteristic Impedance Driven Width constraints. When the Characteristic Impedance Driven Width option is selected, Minimum, Preferred, and Maximum Impedances are then specified, rather than Minimum, Preferred, and Maximum Widths. The Impedances are then converted into corresponding Widths, with the relationships between Impedances and Widths being determined by the specified Impedance Formulae.

To access and change those formulae, the *Layer Stack Manager* dialog needs to be invoked from the menu entry of **Design » Layer Stack Manager**. The **Impedance Calculation** button in that dialog then needs to be clicked on, which invokes the *Impedance Formula Editor* dialog. Each tab in that dialog contains a formula which defines the (characteristic) impedance as a

⁵ Most types of Design Rules are of a Unary nature, and thus have only one associated Scope. However, Clearance, Short Circuit, Acute Angle, Parallel Segment, and Component Clearance types of Design Rules are of a Binary nature, and thus have two associated Scopes.

function of track width, and another formula which defines the track width as a function of (characteristic) impedance. Each of those formulae is specified by query-like expressions, which can be edited by designers.

Clicking on either of the **Helper...** buttons in that dialog invokes the *Query Helper* dialog, but in this case, all of the **PCB Functions** keywords listed in that dialog are listed under an **Impedance** section. Those keywords define assorted properties of the PCB which have an influence on these formulae, thus permitting designers to specify their own formulae.

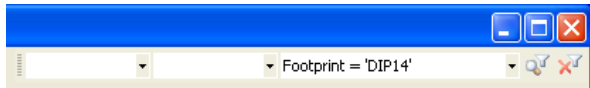
A default pair of formulae has been defined for each tab of the *Impedance Formula Editor* dialog, and each of those formulae can be restored by clicking on the appropriate **Default** button in that dialog.

Unless designers know what they are doing, the default formulae should be used. These formulae have been specified because they do model the relationships between track widths and characteristic impedances to a degree of accuracy.

Filter Toolbar

The *Filter* Toolbar is (currently) only provided for PCB documents. It permits the designer to mask all of the objects within a document except for those having a Net property specified by the designer, or all of the objects within a document except for those forming part of a component specified by the designer. However, designers can also specify queries of their choice, by entering these in the right hand most edit-box provided.

Designers can also select a query from a drop-down list of most recently used queries.



The right hand most edit-box of the *Filter* Toolbar is used to specify the contents of a query.

Unlike when queries are applied from the *Filter* panel, or from *Find Similar Objects* or *Building Query* dialogs, or from resources, it is not possible to specify which options are used when queries are applied from the *Filter* Toolbar; in all cases, any previous query is totally cleared, and the Mask and Zoom options are then used with the current query.

Applying Queries from Resources

It is possible to apply queries from resources (menu entries or toolbar buttons or shortcut keys). The set of default resources for each type of document that supports the use of queries are of a menu entry nature, and are listed under the 'popup' menu that is invoked after pressing the **Y** key.

There are pros and cons to applying queries from resources. The advantage is that queries can be applied faster than by any other method, because designers do not have to key in the query statement, or select the query from the lists of favorite and previously used queries, or specify which properties of similar objects need to match or differ, nor do they have to specify which highlighting options (or any other options) are to be selected when the query is applied; all of those details are specified in the corresponding resource entry.

The disadvantage is that the query statement has been pre-determined, designers are thus unable to change any of them at the time that the query is applied and thus the designer would have to re-edit the resource.

Refer to the *TU0105 Customizing the Altium Designer Resources* tutorial which describes how designers can apply customized queries from customized resources.

Schematic's FilterSelect and PCB's RunQuery processes

To create customized resources that will apply queries in Schematic Library or Schematic documents, the **Sch:FilterSelect** process should be specified (in the **Process** field within an *Edit Command* dialog).

The **PCB:RunQuery** process should be specified to create customized resources that will apply queries in PCB Library and PCB documents.

At least five Parameters also need to be specified for the **FilterSelect** and **RunQuery** processes. An **Expr** Parameter is mandatory to specify the string of the query, and a **ClearExisting** Boolean Parameter is also required. The **Mask**, **Select**, and **Zoom** Boolean parameters also need to be provided as required. (If any of the highlighting option Parameters are not specified, a **False** Parameter Value will be assigned for the corresponding option; thus at least one of those parameters also needs to be specified, and in conjunction with a **True** Parameter value.). The Apply Boolean Parameter also needs to be provided so that the query can be processed.

An example of the text which should be entered into the **Parameters** field within an *Edit Command* dialog is as follows:

```
Expr=IsComment And (Hide = 'True') | Mask=True | Select=False | Zoom=False |
ClearExisting=True | Apply=True
```

The | (pipeline) character is used to separate Parameters.

Thus this Parameter entry stipulates that the contents of the query is the text string `IsComment And (Hide = 'True')` (which will return all Text objects which are also Designator strings of Component objects, and which are also currently in a hidden state), and that the document is to be cleared before that query is applied, and that just the **Mask** highlighting option is to be used. In reality, it was not necessary to additionally specify **False** Parameter Values for the **Mask** and **Select** highlighting options, but the example still demonstrates how five distinct Parameters can be specified in the **Parameters** field.

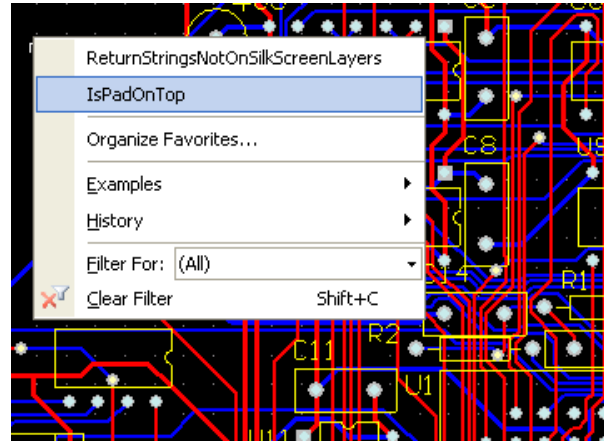
Creating and Using Favorite Queries

To use your favorite queries, click one of the favorites from the **Favorites** list of the *Expression Manager* dialog or by clicking the Y short cut key. This *Expression Manager* dialog is accessed by the **History** or **Favorites** buttons on the *PCB Filter* panel or the *SCH Filter* panel.

To give your favorite queries appropriate names, simply click on a favorite name in the **Favorites** list and click the **Rename** button or click the right hand mouse button to expose the pop up menu and select the **Rename** item.

You cannot directly create favorite queries in the **Favorites** page of the *Expression Manager* dialog you transfer the query expressions from the **History** page to the Favorites page by choosing one of the previously executed queries and then clicking the **Add To Favorites** button. Note, you can use the **Helper** to create an expression and once a new query has been created and executed, it is added to the list of previously executed queries in the **History** page of the *Expression Manager* dialog and then you can transfer them to the **Favorites** page in the *Expression Manager*.

To execute favorite expressions, you can then use the Y short cut key on the PCB or Schematic document or the **Favorites** button from the PCB Filter or the SCH Filter panels. See above illustration where a **IsPadOnTop** favorite query is selected. This query is of `IsPad And OnTop` form.



The Queries pop up menu is invoked on the PCB document when the Y short cut key is pressed.

Inside the Query Language

A query is automatically generated from the *Find Similar Objects* or *Building Query* dialog but designers who want to derive the maximum power of queries need to learn how to define the queries for themselves.

Nature of Queries

The filtering capabilities provided by queries are not confined to any pre-defined list of filtering actions, from which the designer selects an item as required. Instead, the result set of the data filtering feature is determined by the query.

A query consists of a formatted text string suitable for the data filtering mechanism. A query always contains at least one valid keyword, and there is the ability for queries to contain *multiple* keywords.

Clicking the **Helper** button in the *Filter* panel (or the **Query Helper...** button in the *PCB Rules and Constraints Editor* dialog) invokes the *Query Helper* dialog, and that contains the list of keywords.

A brief description is provided for each keyword. The on-line Help (invoked by the **F1** key while a keyword is highlighted) could be consulted for details of which objects are returned by each available keyword, how to use each keyword, and the types and values of parameters if any.

Many of these keywords can be used to specify a valid query. One example of this is the keyword `IsHorizontal` (which is listed in the **PCB Functions** category's **Attribute Checks** section). As the description provided for that keyword indicates (*Is the object a Horizontal track*), a query consisting of just that keyword returns just track objects, and those tracks that are horizontal in nature. (Such tracks have identical **Y1** and **Y2** properties).

Other keywords are not recognized when used in isolation, and need to be used in tandem with recognizable parameters. One example of this is the `ObjectKind` keyword (which is listed in the **Fields** section within the **PCB Functions** category). A valid query which uses that keyword is `ObjectKind = 'Fill'`, and the outcome of using that is that just Fill objects are returned.

An alternative query of `IsFill` (another standalone keyword which is listed in the **Object Type Checks** section within the **PCB Functions** category) would also achieve the same outcome.

Queries and Operators

An operator causes an operation to take place within a statement in a script. The scripting engine supports the following operators.

1. Brackets
2. Not
3. ^, *, /, Div, Mod, And
4. +, -, Or, Xor
5. =, <>, <, >, <=, >=
6. &&, ||

Each line from the list above lists the operators with the same precedence; operator precedence is highest at the start of the list and lowest at the bottom.

Arithmetic Operators

Addition Operator, +

Example:

```
NetPinCount + NetViaCount
```

Subtraction Operator, -

Example:

```
ArcStopAngle - ArcStartAngle
```

Multiplication operator, *

Example:

```
PadXSize_BottomLayer * PadYSize_BottomLayer
```

Division Operator, /

Example:

```
HoleDiameter / ViaDiameter
```

Integral Division operator, Div

Example:

```
Color Div 65536
```

This calculates Color divided by 65536 and the fractional part of the result is discarded.

Mod operator

Mod This is used to obtain the remainder from an integral division operation, equivalent to determining a mathematical modulus.

Example:

```
Color Mod 256
```

This calculates the remainder when Color is divided by 256 without determining the fractional part of the result.

Logic Operators

Logical AND operator, And

Logical AND operator, && This is also used to implement Logical AND, but has a lower order of precedence.

Examples:


```
IsPad And OnMultiLayer
```

```
IsPad && OnMultiLayer
```

Logical Or operator, Or

|| This is also used to implement Logical OR, but has a lower order of precedence.

Example:

```
IsPad Or IsVia
```

```
IsPad || IsVia
```

EXCLUSIVE OR operator, Xor

This is used to implement Logical EXCLUSIVE OR.

Example:

```
OnMultiLayer Xor (HoleDiameter <> 0)
```

To be returned, an object has to either be on the Multi-Layer layer and have a Hole Diameter that is zero, or not be on the Multi-Layer layer and have a Hole Diameter that is not zero.

Not operator

Not This is used to implement Logical NOT.

Example:

```
Not OnMultiLayer
```

To be returned, an object has to not be on the Multi-Layer layer.

Comparison Operators

<

This means Less Than.

Example:

```
HoleDiameter < 40
```

<=

This means Less Than Or Equal To.

Example:

```
HoleDiameter <= 40
```

>=

This means Greater Than Or Equal To.

Example:

```
HoleDiameter >= 40
```

>

This means Greater Than.

Example:

```
HoleDiameter > 40
```

<>

This means Not Equal To.

Example:

```
HoleDiameter <> 40
```

To be returned, an object has to have a Hole Diameter which is not equal to 40.

=

This means Equal To.

Example:

```
HoleDiameter = 40
```

To be returned, an object has to have a Hole Diameter which is equal to 40.

Between ... And ...

This means Greater Than Or Equal To the first number, and Less Than Or Equal To the second number.

Example:

```
HoleDiameter Between 30 And 50
```

To be returned, an object has to have a Hole Diameter that is greater than or equal to 30, and less than or equal to 50.

Like

Like Keywords that are used in conjunction with user-specified strings require that these strings be exactly specified whenever these strings are being compared by the = operator. The use of the Like operator permits the provision of strings which contain one or more Wild Card characters, thus supporting the comparison of strings which are not exactly specified. (See the immediately following Section for a description of Wild Card Characters.)

Example:

```
Name Like 'ADDR?*'
```

This returns objects having a Name property whose associated (text) string begins with 'ADDR' and which contains at least one more character; the string property required has thus not been exactly specified, but has been partially specified.

Wild Card Characters

Wild Card characters permit the provision of strings which are not exactly specified. These characters are typically used in conjunction with other characters, resulting in the provision of strings which are partially specified. A few exceptional keywords can accept string parameters that are not exactly specified, but for the most part, strings can only contain Wild Card characters when these are being compared by the Like operator.

?

This can be replaced by a single character of any type.

Example:

```
Footprint Like 'DIP1?'
```

This returns objects which have a Footprint property of 'DIP10', or 'DIP12', or 'DIP14', etc.

*

This can be replaced by any number of characters, each of which can be of any type.

Example:

```
Footprint Like 'SIP*'
```

This returns objects which have a Footprint property of 'SIP1', or 'SIP12', or 'SIP216', etc. (Any objects having a Footprint property of 'SIP' are also returned, because * can also be replaced by no characters.)

Boolean Strings

True

This affirms the meaning of a Keyword.

Example:

```
IsPad = True
```

To be returned, an object has to be a pad.

False

This negates the meaning of a Keyword.

Example:

```
IsVia = False
```

To be returned, an object has to not be a via.

Queries using Multiple Keywords

The Filtering feature has the ability to specify queries which contain more than one keyword. One example of this is the query of `IsHorizontal Or IsVertical`, which returns all tracks which are horizontal, and all tracks which are vertical. Another example is the query of `IsHorizontal And OnMultiLayer`, and that returns just tracks, and those tracks which are both horizontal *and* on the Multi-Layer layer.

Brackets and Order of Precedence

Brackets have the highest precedence within an order of precedence that has been defined for the various operators provided, and which determines how queries are interpreted by the software (whenever the user has not provided brackets). The sequence of this order is as follows:

1. Brackets
2. Not
3. ^, *, /, Div, Mod, And
4. +, -, Or, Xor
5. =, <>, <, >, <=, >=
6. &&, ||

Ambiguities are resolved by working from left to right. Parentheses are evaluated from inside to outside and equal levels are done left to right.

Examples:

`A And B Or C` becomes `(A And B) Or C`

`A * B + C` becomes `(A * B) + C`

This order of precedence is similar to that used in a Pascal type language. A generous usage of brackets removes doubt, and makes the queries easier to read.

For queries in which the brackets are not strictly required, it is still good practice to use them to make the queries more readable.

The next example demonstrates a mix of the **And** and **Or** keywords, and why brackets must be used to avoid ambiguity:

`((IsPad And OnMultiLayer) Or IsVia) And (HoleDiameter < 16)`

However, the use of the same keywords, and in the same sequence, but with brackets in different locations, results in a query with a different meaning:

`(IsPad And OnMultiLayer) Or (IsVia And (HoleDiameter < 16))`

Both of those queries return all vias whose hole diameter is less than 16mil, but each of them differs in which *pads* are *also* returned. The first query returns pads on the Multi-Layer layer whose hole diameter is less than 16mil, but the second query returns *all* pads on the Multi-Layer layer, regardless of their hole diameters.

Finding Objects with particular Properties

While queries can be used to control which objects are being selected or edited, they can also assist in determining if any objects in the document have properties of a particular nature (or particular combinations of various properties).

For example, the query of `(IsComment Or IsDesignator) And Not OnSilkscreen` returns all Designator strings and Comment strings in a PCB document that are on *neither* of the Overlay layers.

Another example is the query of `IsDesignator And (Hide = 'True')` which returns all Designator strings in a PCB document that are currently in a hidden state.

The query of `IsPolygon And OnSignal And (PolygonRemoveDeadCopper = 'True')` will be very useful for anyone who has ever placed a polygon in a PCB document and then “lost” it. That situation can happen when a polygon is placed on a signal (or “positive” copper) layer and the option of removing “dead” copper is selected; if there are no other objects within the area occupied by the polygon which have the same Net property (as the polygon), the polygon will be “invisible” as a consequence of having *no* associated “child” objects. But applying that query will reveal if any such polygons do exist, and if so, the designer can then double-click on the corresponding row within the *List* panel to invoke the properties dialog for the polygon concerned. At that time, some of the properties of the polygon can then be changed so that it is no longer “hidden” following a subsequent repour.

With the “Padstacks” feature for Multi-Layer pads; it is now possible to specify the dimensions and shape of such pads on *each* signal layer. A query of `ObjectKind = 'Pad' And (PadStackMode = 'Full Stack')` would determine if any pads are in the padstacks mode.

The `(ObjectKind = 'Pad') And (PadStackMode <> 'Simple')` query will return *all* pads which are *not* using the **Simple** option; it will return all pads using the **Top-Middle-Bottom** option, *and* any pads which are using the **Full Stack** option.

Finding Objects with calculated properties

The examples provided so far have demonstrated that it is not difficult to specify a query which will return objects having particular properties (or combinations of properties). What makes queries even more powerful is their ability to return or exclude objects based upon *calculated* (rather than *explicit*) properties.

The query of `IsTrack And (Y1 = Y2)` returns tracks which *also* have identical **Y1** and **Y2** properties. Such tracks are consequently horizontal in nature, and the query of `IsHorizontal` actually returns exactly the same objects.

Another example of this is that the query of `ManHAT > 20` can alternatively be implemented by the (somewhat more complex) query of `IsTrack And ((ABS(X1 - X2) + ABS(Y1 - Y2)) > 20)`.

ATan function example

An example of a filtering capability that cannot otherwise be implemented more simply by the use of any “exotic” keywords, but which nevertheless still can be implemented, is the following query, which returns tracks that are on an angle between 49.9 degrees and 50.1 degrees (i.e. an angle of 50.0 +/- 0.1 degrees):

```
IsTrack And (ATAN((Y1 - Y2) / (X1 - X2)) Between (49.9 * PI / 180) And (50.1 * PI / 180))
```

The `ATAN` function returns an angle (which is the inverse tangent of its parameter) measured in units of radians, and to convert such an angle to degrees, it needs to be multiplied by $(\pi / 180)$. Although designers can convert angles into radians themselves, the provision of the keyword of `PI`, that returns a numeric value equal to π , facilitates specifying angles in degrees whenever keywords of a Trigonometric nature are being used.

IIF keyword example

Avoid the use of queries that might have the potential of being divided by zero, and such an outcome would indeed come about whenever a vertical track was being evaluated (for membership of the result set) by the above `ATan` function query. Thus a better query without the possibility of dividing by zero, is as follows:

```
IsTrack And IIF(X1 <> X2 , (ATAN((Y1 - Y2) / (X1 - X2)) Between (49.9 * PI / 180) And (50.1 * PI / 180)) , 0)
```

The `IIF(L,A,B)` function always requires the three (L, A, and B) parameters. If L is True, then return A, otherwise return B. In the above example, the parameter L is the string `X1 <> X2`, which returns a value of True if **X1** is *not* equal to **X2**, but otherwise returns a value of False.

The parameter A for the IIF function is the string `(ATAN((Y1 - Y2) / (X1 - X2)) Between (49.9 * PI / 180) And (50.1 * PI / 180))`, which is thus evaluated when L is True. When L is True though, the value of `(X1 - X2)` cannot possibly be zero, so the possibility of dividing by zero whenever that string is being evaluated is subsequently eliminated.

The parameter B is the string `0`, which is evaluated when L is False. That happens when **X1** is equal to **X2**, but as that only happens when the track being evaluated is vertical, its associated angle is thus 90 degrees. However, that angle is outside the range of interest (of 49.9 degrees to 50.1 degrees), so returning a result of 0 (or False) results in such a track being unconditionally outside the result set.

Hierarchical aspects

Both Schematic and PCB documents have primitive objects and group objects. Group objects incorporate one or more primitive objects. Primitive objects are either “free” in nature, or “owned” by a group object.

The hierarchical relationships exist between different objects and as an example, it is very useful to be able to move or edit a component object as a single entity, rather than process each of its child objects on an individual basis. However, these hierarchical relationships do have implications for queries, especially when the Mask option is selected during the application of a query.

Parent objects and child objects are distinct entities, and it is possible for a query to mask a particular parent object, while not masking some or even all of its associated child objects. The converse is also possible; a query can assign an unmasked state to a particular parent object, while masking some or even all of its associated child objects.

It is normal to edit the properties of a group object by editing the properties of its parent object, rather than by *directly* editing the properties of any of its child objects. In fact, it is often not even possible to directly edit the properties of child objects.

Masked Parent and Child Objects

When a query is applied and one or more parent objects subsequently acquire a masked state, any child objects of the group object(s) concerned, which have *not (also)* been masked by the currently applied query, often can't have any of their own properties edited at the time, and because their parent object has been masked, it is also not possible to edit the properties of that object either. The implication is that those objects are displayed in a "view only" state, rather than additionally being in an "editable" state.

Even though it is generally not possible to directly edit the properties of child objects, changing the properties of a parent object usually does change at least some properties of at least some of its child objects. That situation is the reason for the graphical view to display all of the (non-hidden) child objects of an unmasked parent object, and regardless of whether each of those child objects is currently masked or otherwise.

When a particular parent object has been masked, thus preventing its properties from being viewed or modified, it is likely that designers still want to be able to view any of its child objects which are not currently masked. Conversely, whenever it is currently possible to edit a parent object's properties, there is still at least some merit in continuing to display all of its child objects in the graphical view, because in many situations changing the properties of a parent object also changes the properties of at least some of its child objects. Thus even when some (or even all) of the child objects are currently masked, the potential remains that at least some of their properties could change, because their parent object is still in an editable state.

In a nutshell, the graphical view can display child objects as though these are unmasked, even though they are not displayed in the list view. And that has implications whenever queries are applied to control how objects are displayed in the target document's graphical view.

As one example, a pad object can be a child object of a parent object and of a net object. If for whatever reason it is desired to display that pad as though it is a masked object in the document's graphical display, it would be necessary to specify a query which would assign a masked property *not just* to that pad itself, but *additionally* to its parent component object, *and* to its parent net object. Failure to mask all three of those objects would result in the pad being displayed in the graphical view as though it was an unmasked object.

It is not without reason that child objects are sometimes displayed in the document's graphical view as though they are unmasked, even though they are not also displayed in the document's list view.

Tricks of the Trade

How to "re-filter" queries

There can be times when designers might want to "re-filter" the objects in a document, in which the objects returned by one query (and *just* those objects) have their properties evaluated by a following query, so that all of the objects which were *not* returned by the first query are all also unconditionally not returned by the second query. To achieve that outcome, all of the objects which are to be evaluated by the second query should be placed in a selected state, and the contents of the second query should be qualified by the use of the `IsSelected` keyword, in conjunction with the `And` operator.

As an example, if the second query would otherwise be `IsPad`, then a query of `IsSelected And IsPad` should be specified instead. That way, all of the Pad objects which are *not* in a selected state at the time that the second query is applied will still be "filtered out" after the second query has been applied.

How to comment a query

Any text within a query's contents which is surrounded by opening and closing curly brackets is ignored by the data feature's parsing software. Thus curly brackets can be used to add comments to queries. An example is as follows;

```
Not OnLayer('Mechanical2') && IsDimension {Determine if any dimension objects in the PCB are
*not* on the Mech 2 Layer.}
```

Where to now?

The designers who have read this article will have learnt enough to at least feel quite confident to experiment with queries. Ultimately, that is how designers will become true masters of using queries. It is the usage and experience that determines your level of expertise.

Appendix: Examples of Useful Queries

The following lists are examples of queries which can be used to accomplish particular objectives.

Queries to use with PCB documents

```
IsComponent And (Copy(Name,1,2) Between 'C0' And 'C9')
```

Returns Component objects whose Name property starts with 'C' and which immediately follows with a number, e.g. C1, C22, etc.

```
InComponent('C*') And HasFootprint('0603') And (IsDesignator Or IsComment)
```

Returns Designator and Comment strings which belong to components which have a Footprint property of '0603' and a Name property which starts with 'C'.

```
InComponent('C*') And HasFootprint('0603') And IsPad
```

Returns pads which are child objects of components, and which parent objects have a Footprint property of '0603' and a Name property which starts with 'C'.

```
(StringType ='Free') And (Component <> 'Free')
```

Returns strings which are child objects of components, but which are neither Designator strings nor Comment strings.

```
OnTopSolderMask Or OnTopSilkscreen
```

```
OnBottomSolderMask Or OnBottomSilkscreen
```

Useful for checking whether any overlaps occur between the objects on one of the Silkscreen layers and the objects on the Solder Mask layer on the same side of the PCB. (When used, the Silkscreen layer selected should be set to be the current layer, and the Solder Mask layer being checked should also be set visible at the time.)

Queries to use with Schematic documents

```
Object_Designator(Parent) Like 'R*'
```

Returns child objects of Part objects whose (Component) Designator property starts with 'R'. E.g. 'R1', 'R2', 'RA1', 'RV12', etc.

```
IsParameter And (Object_Designator(Parent) Like 'R*')
```

As above, but returning just those child objects which are Parameter objects.

```
IsParameter And (Object_ObjectKind(Parent) = 'Part')
```

Returns Parameter objects which are child objects of Part objects.

```
IsPart And (PartId = '')
```

Returns all single-part Part objects.

```
PartId Like '?* '
```

```
PartId <> ''
```

Returns all multiple-part Part objects.

```
(PartDesignator Like 'R*') And (Copy(PartDesignator,2,1) Between '0' And '9')
```

```
Copy(PartDesignator,1,2) Between 'R0' And 'R9'
```


Returns Part objects whose (Component) Designator property starts with 'R' and which immediately follows with a number, e.g. R1, R22, etc.

```
((StringText > '') And (Not IsPowerObject)) Or (ParameterValue > '')
```

Returns all objects incorporating a string, when that string can have its Font property changed, and contains at least one character.

```
IsParameter And OwnerName = ''
```

```
IsParameter And (Object_ObjectKind(Parent) = '')
```

Returns Sheet Parameters.

```
OwnerName = ''
```

```
Object_ObjectKind(Parent) = ''
```

Returns Sheet Parameters, and other objects which are not child objects of any object.

Revision History

Date	Version No.	Revision
9-Dec-2003	1.0	New product release.
24-Aug-2005	1.1	Updated for Altium Designer 2004.
13-Oct-2005	1.2	Revised for Altium Designer 2004.
8-Nov-2005	1.3	Updated for Altium Designer 6.
10-Nov-2005	1.4	Creating and using Favorite Queries sub section added. Revised for Altium Designer 6.
28-Feb-2006	1.5	Updated for Altium Designer 6 – Information on Operators and Brackets added.
1-Mar-2006	1.6	Revised information.
14-Mar-2008	1.7	Updated Page size to A4.
09-Aug-2011	-	Updated template.

Software, hardware, documentation and related materials:

Copyright © 2011 Altium Limited.

All rights reserved. You are permitted to print this document provided that (1) the use of such is for personal use only and will not be copied or posted on any network computer or broadcast in any media, and (2) no modifications of the document is made. Unauthorized duplication, in whole or part, of this document by any means, mechanical or electronic, including translation into another language, except for brief excerpts in published reviews, is prohibited without the express written permission of Altium Limited. Unauthorized duplication of this work may also be prohibited by local statute. Violators may be subject to both criminal and civil penalties, including fines and/or imprisonment.

Altium, Altium Designer, Board Insight, DXP, Innovation Station, LiveDesign, NanoBoard, NanoTalk, OpenBus, P-CAD, SimCode, Situs, TASKING, and Topological Autorouting and their respective logos are trademarks or registered trademarks of Altium Limited or its subsidiaries. All other registered or unregistered trademarks referenced herein are the property of their respective owners and no trademark rights to the same are claimed.