Chapter IV.
Rules

In This Chapter

1. EDS rules defined

EDS trading systems are rule based systems. Each system is composed of one or more trading rules. These rules act as filters that screen your AIQ data to find tickers that meet your technical and fundamental criteria. In EDS terminology, tickers that meet the criteria specified by a rule "pass" that rule. Those that don't "fail" that rule.

Trading systems can be quite simple, consisting of just one or two rules, or highly sophisticated, with many rules involving many different filters or conditions. When EDS scans your AIQ database, it finds those tickers that pass your rules and generates a report. This report contains a section for each rule and each section lists the tickers that passed that rule. A Summary Report is also available which lists all tickers that passed one or more rules and which rules each passed.

Translating your screening criteria into rule expressions requires some understanding of how logical expressions are composed as well as familiarity with the EDS Rule Language. The information in this chapter will help the new user in both of these areas.

Although this chapter is devoted mainly to rules, which are the major component of EDS trading systems, several other types of statements are covered. In addition to rule statements, the EDS language provides other statements that allow you to specify constants, define special functions, and insert comments. These statements can be very useful in defining your trading systems.

Rule structure

All rules consist of three basic parts:

- **Goal** - Label or name you assign to the rule. The Goal is also equivalent to the result of the evaluation of the rule. If the conditions defined in the Rule Body expression are true, a ticker passes the rule and the result is True; if not, it is False.
- **IF** - Separates Goal from Rule Body and identifies the statement as a rule.
- **Rule Body** - An expression specifying of a set of conditions or facts that must be satisfied (True) for the goal to be achieved (i.e., for the rule itself to be True).

The following example illustrates the structure of a rule. This simple rule finds tickers with positive Phase indicator values:

PosPhase if [Phase] > 0.

The three parts of the above rule are:

- **Goal: *PosPhase*** Labels up to 64 characters in length are permitted but it is good practice to use a name that succinctly describes the function of the rule.
- **IF: *if*** Simply identifies this statement as a rule.
- **Rule Body: *[Phase] > 0.*** This expression defines one simple condition: Phase indicator (on the date evaluated) greater than zero (i.e., Phase value is positive).

When this rule is evaluated for a ticker with a Phase value greater than zero, the Rule Body expression is True and the Goal is set to a value of True.

The next example shows a more complex rule. This rule finds tickers that meet all of the following conditions:

1) Upside Expert Rating of 90 or higher

2) Within the past 5 days, Delta Trend Score has a value of 80 or higher

3) Positive Trend Score

BuyWhen if [er up] >= 90 and HiVal ([dts],5) >= 80 and [ts] > 0.

The three components of the above rule are:

- **Goal: *BuyWhen***
- **IF: *if***
- **Rule Body: *[er up] >= 90 and HiVal ([dts],5) >= 80 and [ts] > 0.*** This expression defines three separate conditions combined with the *and* logical operator.

Note

The middle term of the Rule Body expression, ***HiVal ([dts]),5)***, involves the use of a special EDS built-in Function called *HiVal* which examines a series of Field values and returns the highest value from the last specified number of periods. A list of EDS built-in Functions available to the user is found in the EDS Reference Guide.

2. The Rule Body

The most significant part of an EDS rule is the Rule Body. The Rule Body is the expression that defines the screening criteria imposed by the rule. Rule Body expressions are *logical expressions* and as such have only two possible results: True or False.

There is no set formula for constructing a Rule Body expression. In its simplest form, it defines a single condition. More complex expressions involve multiple conditions that are linked together with special operators that specify the role that each plays in the logical process.

Filter Expressions

To keep our explanation of the Rule Body as simple as possible, we'll begin the discussion by focusing on expressions that define only one condition. For the purpose of this discussion, we will call an expression that defines only one condition a **filter** or a **filter expression**. How to construct more complex expressions by combining individual filter expressions is discussed later in this section.

Each filter expression specifies a relationship between two values. When a filter expression is evaluated for a ticker, data for that ticker is retrieved, and the relationship of the two values is checked. If a match is found, the result is true and the ticker "passes" the filter criteria.

For example, consider the following filter expression: **[er up] > 94**

This expression says that we are looking for upside Expert Ratings greater than 94. The two values in this expression are *[er up]* and *94*. (The term in brackets, *er up*, represents the upside Expert Rating field value.) The relationship is specified by the symbol > which is an operator signifying *greater than*.

If this expression is evaluated for a group of stocks, those with Upside Expert Ratings greater than 94 will pass the filter criteria.

Relational Operators

The type of relationship specified by a filter expression is defined by the operator used to relate the two values in the expression. This type of operator is called a *relational operator*. The following relational operators are available in EDS:

=	equal to
<>	not equal to
>	greater than
>=	greater than or equal to
<	less than
<=	less than or equal to

How relational operators are used in logical expressions

The following table demonstrates the use of the different relational operators.

Each of the example logical expressions defines a relationships between two quantities (X and Y). The expressions are evaluated for different values of X and Y. If the values match the relationship stated by the expression, the result is True. If not, the result is False.

Operator	Expression	X	Y	Result
=	X = Y	5	5	True
		5	7	False
<>	X <> Y	5	7	True
		5	5	False
>	X > Y	7	5	True
		5	5	False
>=	X >= Y	5	5	True
		7	5	True
		5	7	False
<	X<Y	5	7	True
		5	5	False
=<	X =< Y	5	5	True
		5	7	True
		7	5	False

Example filter expressions

The best way to gain an understanding of how filter expressions are written is to examine the examples shown in this chapter and those that can be found in the Document files delivered with the EDS system.

Filter expressions are of two general types:

1) Expressions comparing a Field and a numeric Constant. (See next section, *Elements of filter expressions*, for definition of Fields and Constants.)

    [er up] >= 95.

    [volume] > 10000.

2) Expressions comparing two Fields.

    [volume] > [volume esa] * 1.25.

The second Field (volume esa) is multiplied by the factor 1.25. (The symbol * represents multiplication.)

    [close] >= Val([high],1).

This expression involves the use of a special EDS built-in Function called *Val* which returns an historical value for a Field. The returned value is for a specified number of time periods from the present date.

Elements of filter expressions

Filter expressions are constructed from a few basic elements. These basic elements are:

- **Fields** (also known as variables or facts) - Field names represent technical and fundamental values derived from AIQ TradingExpert Pro. Back testing information is also accessible through special application field names. Field names must be enclosed in brackets.
    - **Indicator Fields** - These are the ticker price, volume, Expert Rating, and technical indicator values.
    - **Fundamental Fields -** Fundamental values available through the AIQ Fundamentals module.
    - **Application Fields -** Historical price and other information from positions taken within a back test.
- **Key Words** - A set of arithmetic, relational, and logical operators (+, -, =, >, and, or, not, etc.), a few specially defined words (if, is, and define), and the period (.) used to terminate all Rule Body expressions.

- **Built-in Functions** - A set of predefined routines furnished with EDS which perform mathematical procedures (e.g., square root of a number), return specific quantities (e.g., highest value in a series of numbers), or return a logical value (true/false).
- **User Defined Functions** - Routines defined by the user within a Document.
- **User Constants** - Quantitative values defined by the user within a Document.

> **Note**
> Lists of the built-in Functions, and Prebuilt Routines available in EDS can be found in the EDS Reference Guide.

In addition to the above basic elements, EDS allows the use of predefined rules in filter expressions. There are two types of rule labels that can be inserted into an expression:

- **Pre-built Routines** - A set of predefined rules furnished with EDS.
- **User Rules** - Rules defined by the user within a Document.

Fields

In mathematical terms, Fields are the variables in a rule expression. When EDS evaluates a rule for a ticker, it retrieves quantitative values for all of the fields specified in the rule. The values retrieved for a particular ticker determine whether or not that ticker meets the criteria specified by that rule.

The EDS rule compiler recognizes a Field by [ ] or {} brackets surrounding the Field name. For a list of available Fields, see the EDS Reference Guide.

Indicator Fields

In addition to price and volume data, indicator Fields available in EDS include all of the technical indicator values and the Expert Ratings computed by AIQ TradingExpert Pro. Indicator Field names must be enclosed in [ ] brackets.

Fundamental Fields

EDS provides fundamental screening with data derived from the AIQ TradingExpert Pro Fundamentals module. The Fundamental Fields that are available in EDS are determined by the fundamental strategy currently specified in the Fundamentals module. Only those Fields designated for this strategy are accessible in EDS. Fundamental Field names must be enclosed in [ ] brackets.

Application Fields

Special fields used in constructing rules for exiting positions during a back test. The information includes the position high, low, and entry prices plus the number of days the position has been held and the entry date. Application Field names must be enclosed in {} brackets.

Key Words

EDS provides a number of special symbols and words (called Key Words) that are available for use in rule expressions.

Mathematical Operators

+, -, *, /, mod

Relational Operators

=, >, >=, <, <=, <>

Logical (Boolean) Operators

or, and, not, and not, then

When a Rule Body expression consists of multiple conditions, logical (Boolean) operators are used to combine conditions and to define the role that each plays in the logical process. The *then* operator signifies that, if the first condition (condition to the left of *then*) is true, the next condition shall be evaluated and will determine the expression result.

Special Operator

^

Forces a change in the date that an expression is evaluated. When a ^ occurs before an expression, that expression is evaluated using the RuleDate() rather than the ReportDate().

Example:

CloseDiff is ^[close] - [close].
Report if ScanAny([er up] >= 95, 30)

After running *Report*, *CloseDiff* will display the difference between the closing price on the ER date and today's closing price.

Special Words

The EDS compiler recognizes three special words (*if*, *is* and *define*) as statement identifiers. The *if* and *is* identifiers must occur immediately following the statement label (Goal). The *define* identifier must appear first in the statement.

IF Identifies rule

Example: Buy if [er up] >= 95.

IS Identifies User Defined Function

Example: AvgPrice is ([low] + [high] + [close]) / 3.

The *is* after the label identifies this statement as defining a quantity called *AvgPrice*.

DEFINE Identifies User Constant

Example: define erupval 95.

Beginning a statement with the word *define* identifies this statement as defining a constant called *erupval* (the second word in a define statement is the statement label).

Built-in Functions

EDS supplies over 40 different Functions which are available for use in defining filter expressions. These Functions perform various procedures and computations which were selected for their usefulness in these types of expressions. For a list of available Functions, see the EDS Reference Guide. This Guide also provides a complete description and an example for each Function.

Functions fall into several general categories:

- perform mathematical procedures (e.g., square root of a number)
- return specific quantities (e.g., highest value in a series of numbers or a date)
- return a logical value (true/false)

When a Function is entered into an expression, it must be entered according to the specifications listed in the EDS Reference Guide. All parameters required for a Function must be enclosed within parentheses and if multiple parameters are required they must be separated by commas. See the EDS Reference Guide for the obligatory and optional parameters.

The easiest way to use a Function is to paste it from the Rule Builder

(see section 7 in this chapter). When the Rule Builder pastes a Function, it shows you the exact location of all parameters and you simply type them into the indicated spaces.

Function examples:

Log10(number)

The Function Log10 returns the base 10 logarithm of a number.

Example expression: **Log10(100)**

Returns the base 10 logarithm of 100.

HiVal(field, periods, [dateoffset])

The Function HiVal returns the highest value for a Field over a given period of time. The parameter dateoffset is enclosed in brackets [] to indicate that it is an optional parameter. Mandatory parameters are Field (name enclosed in brackets) and periods (an integer number).

Example expression: **HiVal([er up],21)**

Returns the highest Er in last 21 days.

Stock()

The Function *Stock* returns the logical value *True* if the current ticker is a stock and is used to filter tickers based on type. No parameters are required for this Function.

Note:
A User Defined Function can be used to define an indicator which can then be added to TradingExpert Pro's Indicator Library. See EDS Chapter VII.

Example of Function used in a rule:

StockRule if Stock() and [er up] > 90.

Rule finds stocks with ER above 90.

ReportDate()

The Function *ReportDate* returns the date of the report that is being run. No parameters are required for this Function.

Example of Function used in a User Defined Function:

Date is ReportDate()

This statement creates the label *Date* which can be inserted as a column in a report.

User Defined Functions

A User Defined Function is a function created by the user by means of the special EDS language statement that includes the *is* identifier. See Section 3 for information on creating User Defined Functions.

When the label assigned to a User Defined Function is inserted into a rule, it serves the same purpose as the function itself. Substituting User Defined Functions for complex terms in expressions can simplify the writing of rules and help prevent errors. A User Defined Function can only be used within the Document in which it was created.

Example

The following statement creates a User Defined Function named *PreviousDaysClose*. This function uses the built-in Function *val* to obtain the value of the *close* Field for the previous day (i.e., one day back from the current day).

PreviousDaysClose is val([close],1).

We can now write a rule called *CloseUp* which is True for tickers that close with a price greater than the previous day's close. The Function *PreviousDaysClose* can be used in this rule as follows:

CloseUp if [close] > PreviousDaysClose.

Pre-built Routines and User Rules

Pre-built Routines are common rules that are supplied with EDS and are easily incorporated into the user's trading systems. These routines are always available (not Document specific) and are designed to help new users and those who are less mathematically inclined develop their personal trading systems.

See the EDS Reference Guide for information on the Pre-built Routines included with EDS.

User Rules are rules that have been created by the user within a particular Document. They are only available from within the Document in which they were created. However, rules can be copied from one Document and pasted into another.

To use a Pre-built Routine or a User Rule in the rule you are defining, you simply insert the label assigned to the Routine or Rule into the Rule Body expression. This label represents the previously defined logical expression. The following example illustrates this concept.

The goal of this example is to find stocks with upside ER of at least 95 and an increasing Phase indicator. Instead of writing one long rule, we can use two of the Pre-built Routines, *erup* and *GoodPhaseUp* which are defined as follows:

erup if [er up] >= 95

GoodPhaseUp if [phase] > 0 and slope([phase],3) > 0.

The rule *GoodPhaseUp* also illustrates the use of a built-in Function. The Function *slope* computes the slope of a given Field for a specified number of periods (3 periods in this example). The value returned by the Function is indicative of the trend of the Field where a positive slope value indicates an upward trend.

We can now use these two rules to construct our final rule:

BuyWhen if erup and GoodPhaseUp.

Separating a long rule into several shorter rules as in the above example is a technique that can greatly simplify the writing of complex expressions. This is especially true where a single rule would result in a long and cumbersome Rule Body in which errors might be difficult to detect.

The goal of this next example is to find stocks with upside Expert Ratings greater than 90 and a spike in volume. Instead of writing this as one long rule, we can first define two intermediate Goals, *UpER* and *VolumeSpike* by writing the following rules:

UpER if [ERUP] > 90.

VolumeSpike if [volume] >= [volume esa]*1.25

We can now use these two User Rules to construct our final rule:

BuyWhen if UpER and VolumeSpike.

User Constants

A User Constant is a special label defined by the user to represent a numeric value. See Section 3 for information on how User Constants are created.

Example

The User Constant *erupval* can be defined as follows:

define erupval 95. (Sets erupval = 95)

This constant may now be used in any rule within the same Document. The following rule, named *erup*, is designed to finds stocks with upside Expert Ratings greater than or equal to the value of the constant *erupval*:

erup if [er up] >= erupval.

> **Note**
> Logical operators are known interchangeably as *Boolean* operators after George Boole who initiated the algebraic study of truth values.

Multiple condition expressions

Rule Body expressions may contain multiple filtering criteria or *conditions*. In a multiple filter expression, each filtering criteria or condition is a separate logical expression with its own True or False result. To combine logical expressions, EDS allows the use of the operators *and*, *or*, and *not*. These operators, called *logical operators*, define the role that each filter or condition plays in the logical process.

A compound expression requiring that both condition A *and* condition B are True is called a conjunction. Conjunctions use the *and* logical operator. A goal requiring only that condition A *or* condition B is True is called a disjunction.

How logical operators are used to combine conditions

The following three logical expressions consist of two conditions (A and B). In each expression, the two conditions are combined with a different operator. The results of evaluating these expressions with different combinations of values for A and B are listed in the rightmost column.

Operator	Expression	Value of A	Value of B	Result
and	A and B	True	True	True
		True	False	False
or	A or B	True	False	True
		False	False	False
not	A not B	True	False	True
		True	True	False

Examples of multiple condition Rule Body expressions:

Following expressions combine two filter expressions with the logical operator *and*:

[er up] = 16 and [er down] =56.

[phase] > 0 and slope([phase],3) > 0.

The second expression involves the use of a special EDS Function called *slope* which returns the slope or trend of a Field over a specified number of time periods.

Notes

1. Quantities within parenthesis are evaluated before they are combined with any quantities outside the parenthesis.
2. If an expression contains a label representing a quantitative or logical expression, the expression is evaluated first.

How Rule Body expressions are evaluated

Expressions are evaluated in the following order:

1. Functions.
2. Any arithmetic expressions in the logical expression.
3. Expressions involving relational operators (>, <, >=, =<, =, and <>).
4. Finally the logical operators are applied left to right in the order of their precedence: *not*, then *and*, then *or*.

Evaluation date

When you request an EDS screening report, your rules are evaluated for all tickers specified in the Document's Properties. The first step in this process is to retrieve the required Field values. As all Field values (except Fundamental Fields) are specific both to a ticker and a date, EDS maintains an internal date which it uses for retrieving Field data. Initially, EDS sets this internal date to the Report Date and all Field data is retrieved for this date unless otherwise specified.

The Report Date appears on the EDS toolbar and when a report is initially displayed the date shown is the date when the report was last run. Prior to generating a new report, you can easily adjust this date to the date of your most recent data or any other date that you want the report to reflect.

When your rules require data prior to the Report Date, EDS provides Functions for this purpose. A number of Functions are designed to evaluate a series of data values starting from the most recent date and moving backwards in time. These Functions allow you to specify both a starting date and the number of time periods.

EDS also provides Functions that will alter the internal date to the date of a specified event. For example, you can change the internal date to the date when a ticker issues an up ER of 95 or greater. Another Function is available to reset the date back to the Report Date.

The above does not apply to Fundamental Fields as historical fundamental data is not maintained by AIQ. Only the last Fundamental Field data that was obtained is accessible for any ticker. If a Function requests an historical value for a Fundamental Field, the value returned is always the most recent value.

3. How to create User Defined Functions, User Constants, and Comments

User Defined Functions

User Defined Functions are created by the EDS language statement which includes the *is* identifier. This statement defines an expression and assigns to it a label. The expression can be any sort of equivalence or mathematical routine that you wish to use in defining the rules contained within a Document. Fields, arithmetic operators, User Constants, built-in Functions, and other User Defined Functions are all allowed in these expressions. The result of the expression can be either a quantity or a date.

Examples of User Defined Functions:

1. Expression which sets a label equal to a Field:

   TodaysClose is [close].

2. Expression which calls a Function to derive a quantity from a Field.

   P**reviousDaysClose is val([close],1).**

   In the above statement, the Function *val* returns the value of the *close* Field for the previous day (i.e., one day back from the current day).

3. Expression which mathematically computes a quantity from other User Defined Functions.

   Using the above two Functions, we can first write the following expression to compute the quantity change:

   Change is TodaysClose - PreviousDaysClose.

4. Expression which mathematically computes a quantity from several Field values.

   AvgPrice is ([low] + [high] + [close]) /3.

   The label AvgPrice is defined as the result of dividing the sum of the three Fields (low, high, and close) by 3.

5. Expression that calls a Function to derive a date.

   Date is ReportDate().

   The label *Date* is defined as the date returned by the Function *ReportDate*. This Function returns the date of the report being run.

User Constants

User Constants are labels created to represent numeric values. They are created by the EDS language statement which begins with the key word **define**. Once a Constant has been created in a Document, it can be used in any filter expression or User Defined Function within the same Document.

Examples of statements defining User Constants:

define LongTrend 30.

Defines constant *LongTrend* equal to 30.

define periods 21.

Defines constant *periods* equal to 21.

Comments

User Comments may be placed anywhere within your EDS Documents. Any statement beginning with a ! is treated as a Comment and is ignored by the EDS language compiler.

Examples of Comment statements:

! Rule - True if Up Expert Rating and a change in DTS in last 5 days

! Crossover Rule tests for an esa crossover

4. Syntax requirements for EDS rules

The syntactic rules of the EDS rule language are as follows:

General

- Fields must be enclosed in brackets [ ]
- Names of constants and variables (labels) are not enclosed in brackets.
- Terms in the Rule Body may be combined within parenthesis ( ) to specify the order of computation.
- All statements except Comment statements must end with a period.
- Statements beginning with ! are comments and are ignored and not compiled.
- Statements beginning with the word *define* are special statements used to define Constants.

Rules (IF):

- Name (Goal) always occurs first followed by a blank space. Name can contain a maximum of 64 characters.
- The name is followed by *if*.
- The Rule Body which follows the initial *if* contains all conditions to be evaluated.
- Name (Goal) must not contain spaces.
- First character of Name (Goal) cannot be a number.

User Defined Functions (IS):

- Name (name of function) always occurs first followed by a blank space. Name can contain up to a maximum of 64 characters.
- Name is followed by *is*.
- The Body which follows *is* consists of an expression which derives a numeric value or a date.

User Constants (DEFINE):

- Word *define* always occurs first followed by a blank space.
- Name (name of constant) follows *define*. Name can contain up to a maximum of 64 characters.
- The final term is a numeric constant.

5. Examples of EDS rules

The statement, "Find all stocks with an up ER greater than 90" can be written as a rule in the EDS language as follows:

UpER if [ER UP] > 90.

Given this rule, we can proceed to give the system the goal "UpER". The system will find all stocks with ER's greater than 90. This is a rather simple rule.

Tip - We could have called this goal anything up to 64 characters.

For a more complex rule, we will create a conjunctive rule. The goal will be "I want to buy a stock when the ER is greater than 90 and the Phase is positive." We can write this as follows:

UpER if [ERUP] > 90.

PosPhase if [Phase] > 0.

BuyWhen if UpER and PosPhase.

Our goal is BuyWhen. EDS will search through all stocks and try and satisfy the goal BuyWhen. If the goal is satisfied for a particular stock, that stock will be returned as meeting the goal. BuyWhen is satisfied when both the UpER rule and the PosPhase rule return True. This is known as a conjunctive goal.

If we write this rule so that we look for stocks with either an Up ER or a Positive Phase, this would be what is known as a disjunctive rule and can easily be written as:

UpER if [ER UP] > 90.

PosPhase if [Phase] > 0.

BuyWhen if UpEr or PosPhase.

In this case, EDS will return stocks that match either condition stated by the BuyWhen rule. If a stock matches the UpER subgoal, it will be returned as True (pass). If it doesn't, the next subgoal (PosPhase) will be evaluated. If neither rule can be matched, the stock will be returned as False (fail).

Note

All Fields (facts) are contained in brackets ([]). The Field symbols represent the ticker price, volume, and indicator data values as well as fundamental data values. The actual data values are derived from AIQ TradingExpert for the date of the report or, for Fundamental Fields, the last date retrieved.

6. How to create rules in the Rule Library

EDS provides powerful tools for creating trading rules. Rules can be entered directly into your EDS Documents using the built-in text editor or you can use the EDS Rule Builder, an easy-to-use "wizard" function. The Rule Builder allows you to quickly find any of the basic rule components or predefined routines and easily paste them into the rule that you are defining. See next section for information on using the Rule Builder.

 To enter rules directly into a Document's Rule Library:

1. Open EDS and click the **Rule Library** tab if it is not already selected.
2. Type your rules into the Rule Library.

The Rule Library provides a full-featured text editor that allows you to enter text directly into the main window. When entering rules, you must use correct EDS rule syntax. EDS automatically parses and compiles each rule as it is completed. If the compiler doesn't recognize a line of text, it will alert you to the line which contains the error.

7. Using the Rule Builder

The Rule Builder is a "wizard" type function that is evoked from the Rule Library. It provides a quick and easy way to build your rules by providing lists of all the rule elements (Key Words, Fields, Constants, and Functions) available in EDS. You build a rule by finding the element you need from the appropriate list and pasting it into a text string. When you complete a rule, you simply transfer the text directly to the Rule Library.

Opening the Rule Builder

The Rule Builder function can only be accessed from the Rule Library. There are several ways to access the Rule Builder dialog box from the Rule Library:

- Right click your mouse and select **Builder** from the floating menu.
- Click **Builder** command button on tool bar.
- Click **Edit** on menu bar and select **Builder** from *Edit* submenu.

Using the Rule Builder

In the following example, we will use the Rule Builder to create the example rule *PosPhase* which is discussed above:

Rule Builder dialog box

1. From the Rule Library, open the Rule Builder dialog box.
2. From *Select category* list box (1), select **KeyWords**. A list of Key Words will appear in the *Select item to paste* list box (2).
3. From *Select item to paste* list box (2), select **if.** The following text will appear in the text box immediately below boxes 1 and 2: **<<rule>> if <<boolean>>**.
4. Click on **Paste** command (3). The above text string is now transferred to the lower text box (4).
5. In text box (4), click on <<**rule**>> and type the rule goal **PosPhase** in the highlighted space. The text string in box (4) is now as follows: **PosPhase if <<boolean>>**.
6. In text box (4), click on <<**boolean**>>. Again the space between the << >> is highlighted.
7. In list box (2), select the KeyWord **>** (**greater than)** and click **Paste**. In text box (4), the following text string now appears: **PosPhase if <<number>> >= <<number>>.**

8. In text box (4), click on the first <<number>> term. Again the space between the << >> is highlighted.
9. In list box (1), select the **Fields** category. A list of Fields will appear in list box (2).
10. In list box (2), select **[Phase]** (Phase indicator Field) and click **Paste**. In text box (4), the text string is now as follows: **PosPhase if [Phase] = <<number>>**.
11. In text box (4), click on the remaining <<number>> term. The space between the << >> is highlighted. In this space, type the number **0**. Text box (4) now contains our completed rule: **PosPhase if [Phase] = 0**.
12. To paste this rule into the Rule Library, click **OK**.

The Rule Builder allows complete flexibility in building your rules. You can use the Rule Builder to build all or part of a rule. Instead of building the entire rule in the Rule Builder, you can transfer a portion of a rule from the Rule Builder to the Rule Library and then complete it by typing additional text in the Rule Library.

You can also use the Rule Builder to edit an existing rule.

To edit an existing rule:

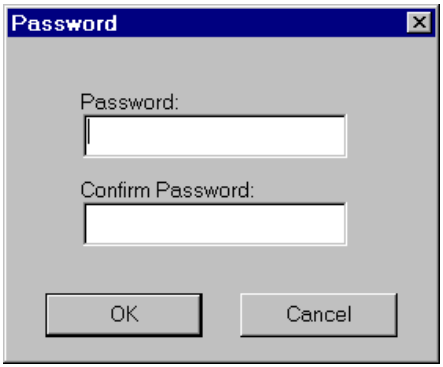
1. In the Rule Library window, highlight the rule that you want to edit.
2. With the rule still highlighted, open the Rule Builder function. When the Rule Builder dialog box opens, the highlighted rule will appear in the Rule Builder text box (4).
3. To change an element, such as a Field or Function, within the rule, highlight the element you want to change.
4. To replace the highlighted element, select a new element from the appropriate list in box (2) and click the **Paste** command. The selected element will replace the highlighted element in box (4).
5. When you complete your changes, click **OK**. In the Rule Library, the edited rule now appears in place of the original rule.

8. How to hide your rules

EDS provides a special function that can be used to hide your trading rules while still allowing access to all other EDS functions. When evoked, this function removes the Rule Library tab so that the Rule Library can not be displayed and, therefore, the code for your rules can not be viewed. A password is required to unhide the Rule Library.

 To hide the Rule Library:

1. Open EDS and click **File** on the menu bar.
2. From the File menu, click **Hide Rule Code**. The *Password* dialog box will appear.



3. Type a Password of your choice into the upper text box. Be sure to use a Password that you can easily remember.
4. Type the identical Password into the text box labeled *Confirm Password* then click **OK.** The Rule library tab will be removed from the screen.

 To unhide the Rule Library:

1. Open EDS and click **File** on the menu bar.
2. From the File menu, click **Unhide Rule Code**. The *Password* dialog box will appear.
3. Type your Password in the text box provided.
4. Click **OK** to unhide the Rule library tab and restore access to the Rule Library.