User Guide

Table of Contents

Introduction

BREeze (**B**usiness **R**ules **E**ngine) is a dynamic rules engine that allows you to develop a rules set that can set data in any field, or fields, on the Lead or Case object. BREeze can simplify your complex data rules with an easy to use interface that doesn't require a developer to maintain.

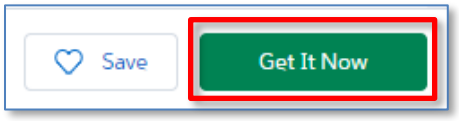
The BREeze application provides the ability to create a rule against either the Lead **or** Case object and define the rules set to meet your business specific needs.

For organizations that would like to extend BREeze to other objects, and leverage advanced functionality such as creating child records, visit our BREeze Premium listing on the AppExchange, or contact gearsdesignsupport@rafter.one, for more information.

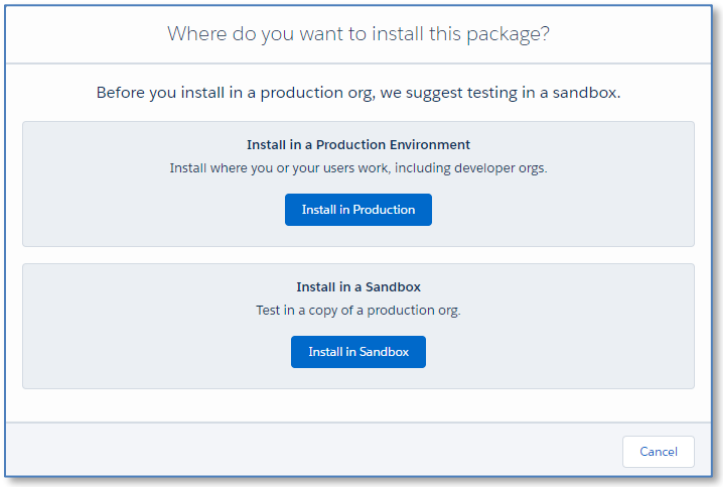
Installation

To install the BREeze application in your Salesforce instance, you must follow these steps:

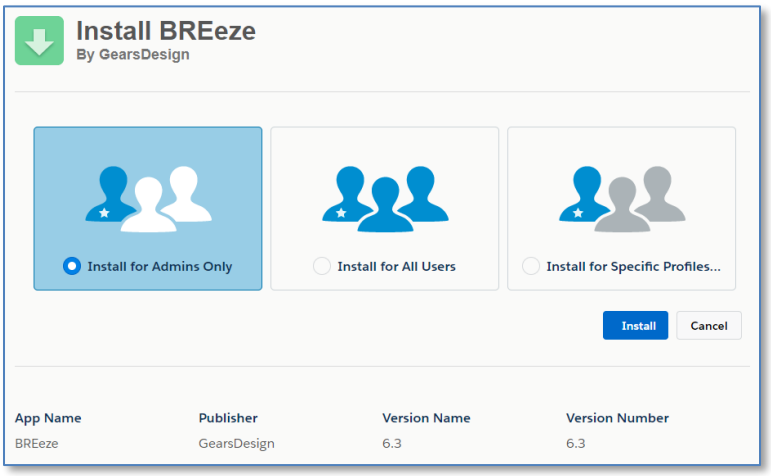
- Navigate to the BREeze application via the AppExchange or go directly there by using the following URL: https://appexchange.salesforce.com/listingDetail?listingId=a0N3A00000ErAwrUAF
- Click on the Get It Now button in the bottom right-hand corner



- Complete any necessary steps to log into your Salesforce org
- Next, you'll be prompted to choose where to install the package
    - If doing an initial installation of the application for trialing purposes, we recommend that it is first installed in a sandbox



- After selecting the location for the installation, you'll be prompted with a screen to confirm the installation
- Once you have confirmed the information for where the application will be installed, check the terms and conditions box and then click the 'Confirm and Install' button
- You may then be prompted to log into your Salesforce instance a final time to verify the installation location
- Next, you'll select the necessary access to the application
    - We recommend that in most instances, only the System Administrator should have access to BREeze so you can leave the default selection



- Once the appropriate access level is selected, click the Install button
- Depending on the number of customizations made to your org, the installation of BREeze can take anywhere between a minute to ten minutes to complete
- If any installation errors occur, please contact gearsdesignsupport@rafter.one for troubleshooting assistance

General Execution

As mentioned above, the BREeze package comes pre-configured with a trigger for Lead & Case rules however only one can be used at a time in the free version of BREeze. Depending on your use case, the triggers are designed to fire either as a before **OR** after insert trigger. If you require the rules to fire on update, please contact GearsDesign to find out more information on the full-featured BREeze Premium application.

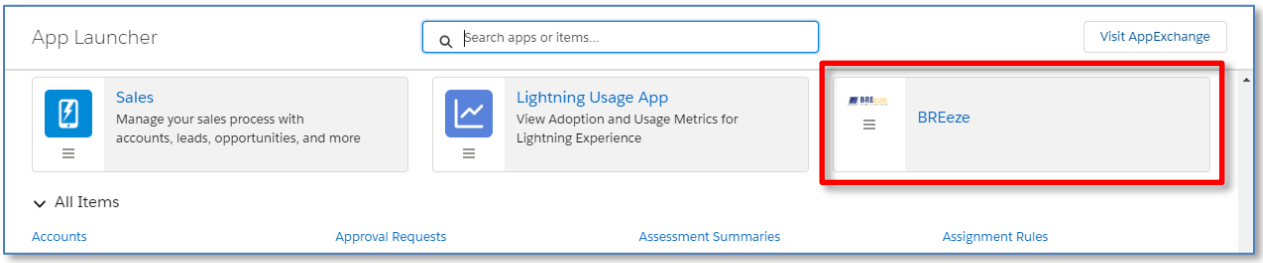
Field Level Security

BREeze adheres to Salesforce object & field level security (FLS). For example, if a BREeze rule is configured to update a field, and the user that triggered the BREeze rule to be fired does not have access to that field via FLS, then BREeze will not update that field.

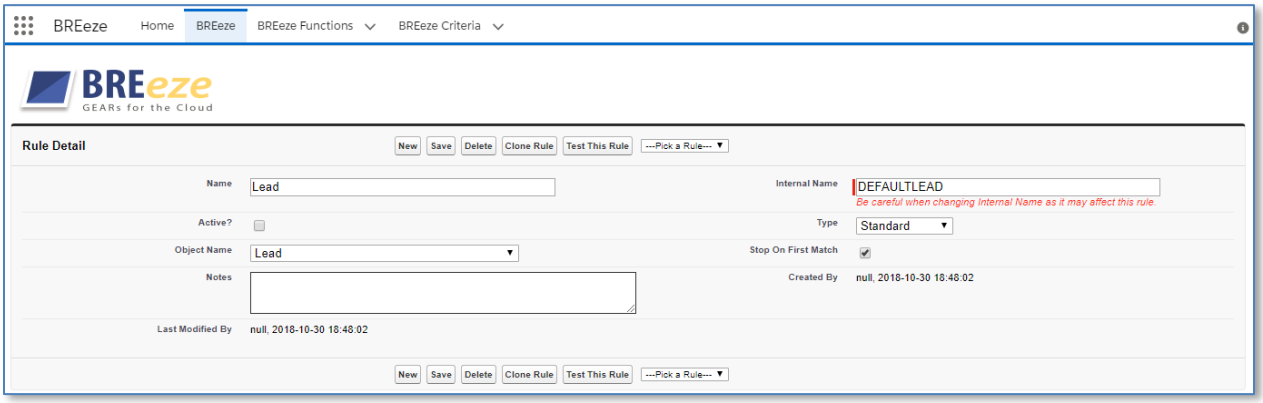
General Setup

Creating a New Rule

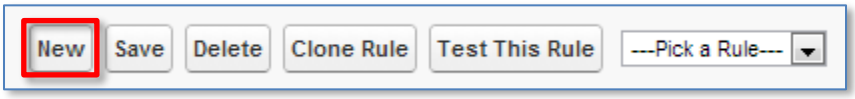
- From your Salesforce App Launcher, navigate to the BREeze app



- Once the app has been selected, click on the BREeze tab
- The following screen should be presented:

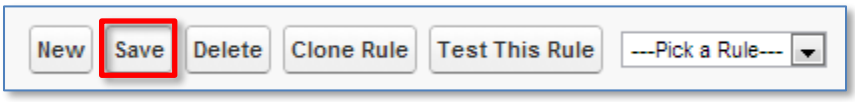


- Next, click on the ***New*** button at the top of the screen

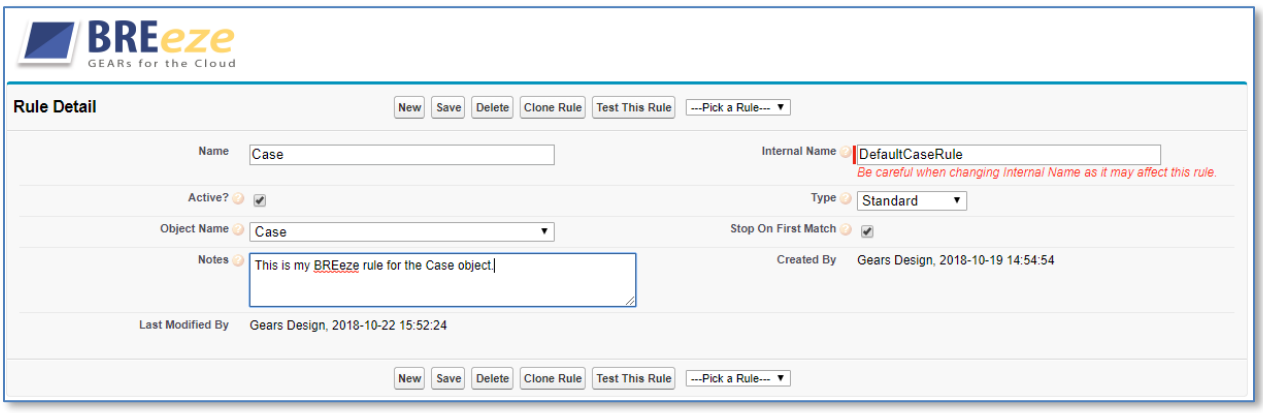


- From the resulting screen, enter the following:
  - **Name**
    - Enter a descriptive name that describes the rule which you are building
  - **Active**
    - While building and testing the rule, this can remain unchecked however when you are looking to fully execute the rule on your data, you must check this box
  - **Type**
    - Select 'Standard'
  - **Object Name**

        - Select either **Lead** or **Case** from the dropdown menu depending on your use case.
            - A BREeze Premium subscription is required to extend the application to other Salesforce objects.
    - **Internal Name**
        - If you are creating a before insert rule, use:
            - Lead Object = "**DefaultLeadRule**"
            - Case Object = "**DefaultCaseRule**"
        - If you are creating an after insert rule, use:
            - Lead Object = "**DefaultLeadRuleAfter**"
            - Case Object = "**DefaultCaseRuleAfter**"
        - **Important Note**: If any other Internal Name value is given to the rule, it will not fire correctly. The value must be one of the four noted above.
    - **Stop On First Match**
        - This indicates whether the rules engine will stop processing when it hits the first entry where all criteria evaluate to true or whether it continues through all rules and evaluates/executes on all rules that evaluate to true.
    - **Notes**
        - Enter a long description of the purpose of the rule or any other pertinent details
- Finally, click the ***Save*** button at the top or bottom of the Rule Detail section

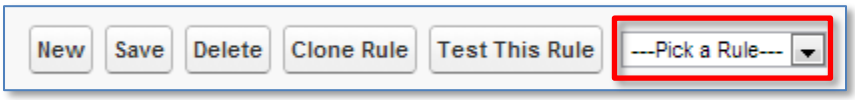


- The result should look like something similar to this with your rule name & associated object populated:

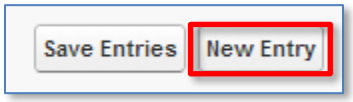


Creating New Rule Entries

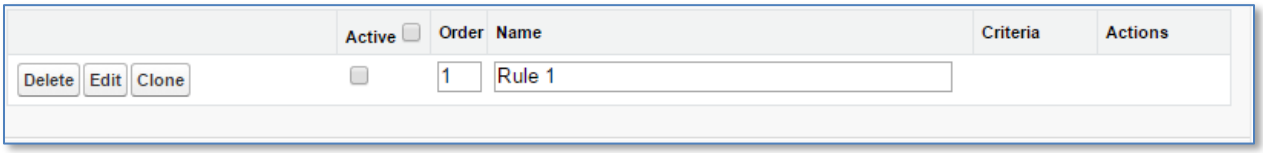
- Navigate to the BREeze tab within Salesforce
- From the dropdown menu, select the Rule you'd like to add a Rule Entry to



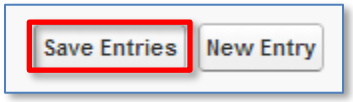
- Under the Rule Entries section, click on the ***New Entry*** button



- A new Rule Entry will appear within the Rules Entries section

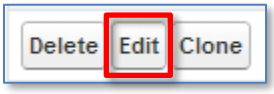


- By default, the entry will be named "Rule 1". In this field, enter a name that describes the criteria and/or action to be evaluated in that entry (not required) and then click on the ***Save Entries*** button

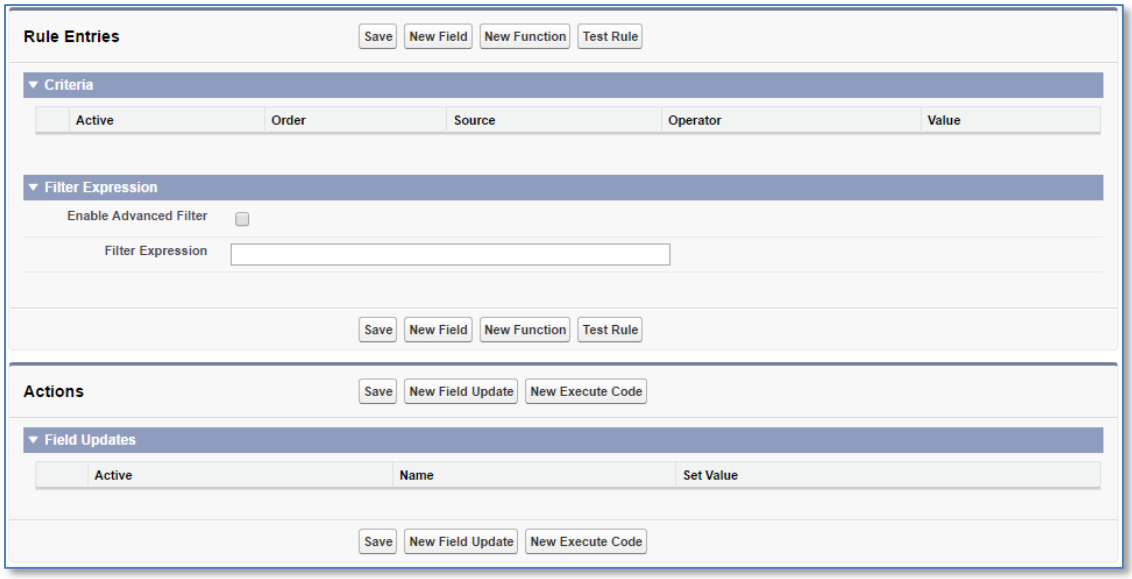


Defining Standard Rule Entries

- Click the ***Edit*** button on the left side of the Rule Entry

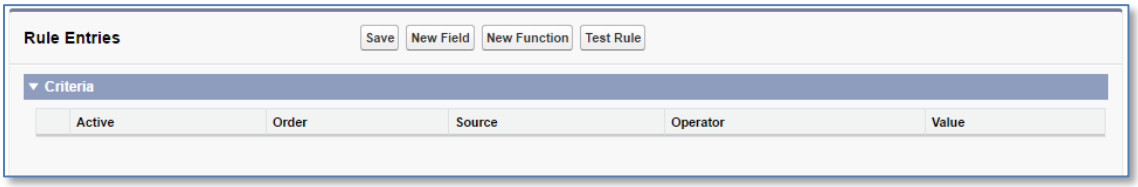


- The resulting screen allows you to define the rule criteria & actions:



- Starting with the top half of the screen under the Rule Entries header, this is where you'll begin to define the criteria for your Rule Entry:
    - **Criteria**
        - From this section, identify the field and/or function criteria which should be evaluated
    - **Filter Expression**
        - For complex criteria, use this section to build out an expression leveraging AND/OR groupings
            - BREeze assumes that all criteria are joined via "and". Meaning if there are two criteria entries, it would be interpreted by default as: 'Criteria 1' AND 'Criteria 2'. If the criteria should be interpreted otherwise (for example, 'Criteria 1' OR 'Criteria 2'), use this section to define that logic.
- The bottom half of the screen is where you'll define the actions to take place on the record should the criteria be met for the specific Rule Entry
    - **Field Updates**
        - Define the actions (either field update or execute code action) that will take place if the record meets the criteria

Defining Criteria



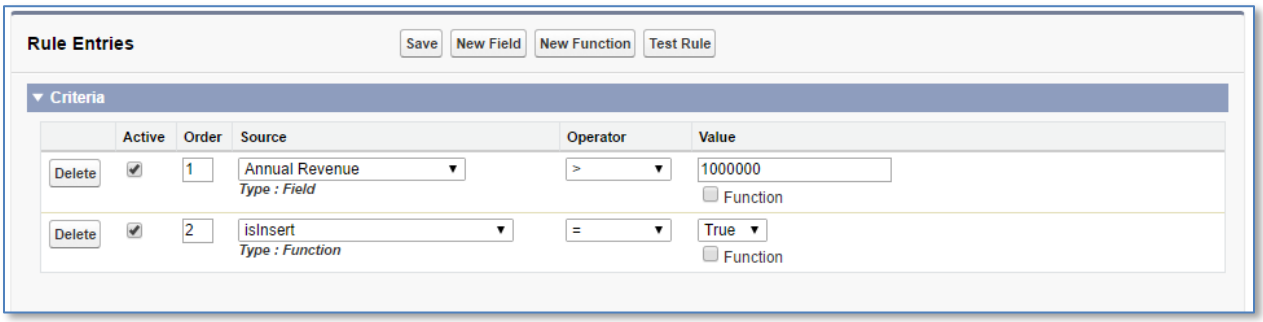
- Click on either the ***New Field*** or the ***New Function*** button

- **New Field**
  - Allows you to select any field from the object which your initial rule was set up against and evaluate that against a single value, multiple values or evaluate that against a function.
    - Example: Annual Revenue > 10000000
- **New Function**
  - Allows you to evaluate a function against a set single value, multiple values or evaluate that against another function.
    - Example: My_Date__c = TODAY

- The standard operators are as follows:

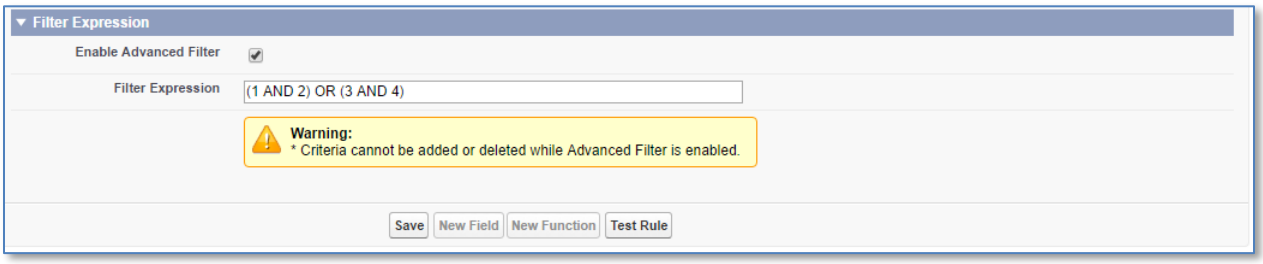
Operator	Definition
=	Equals
!=	Does Not Equal
<	Less Than
<=	Less Than or Equal To
>	Greater Than
>=	Greater Than or Equal To
contains	Determines if the value exists within the selected field
not contains	Determines if the value does not exist within the selected field
in	Allows for multiple values to be entered without using OR and compares the values to the selected field
not in	Allows for multiple values to be entered without using OR and compares the values to the selected field
in (Case Sensitive)	Allows for multiple values to be entered without using OR and compares the values to the selected field and also enforces case sensitivity
isNull	Determines if there is no value in the selected field
StartsWith	Evaluates whether the field begins with the value

- Enter all appropriate fields/functions that should be used in the first evaluation and click the ***Save*** button



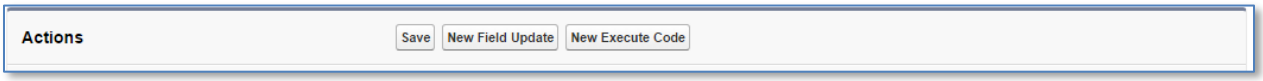
Using Filter Expressions

- Next, determine whether your criteria requires an advanced filter expression
  - If all your criteria should be joined with AND's, then there is no need to use a filter expression as it is assumed
    - Example: 1 AND 2 AND 3
  - If your criteria requires grouping and combinations of AND's & OR's, enter your criteria into the Filter Expression text box and then check the Enable Advanced Filter checkbox
    - Examples: "1 OR 2", "(1 AND 2) OR (3 AND 4)", "(1 OR 2) AND 3"
    - **Important Note:** Once you have enabled a filter expression, you will not be able to add or delete rule criteria. If you find that you need to add or delete criteria after enabling the advanced filter, simply uncheck the Enable Advanced Filter checkbox and you will be able to add or delete as appropriate. Before re-enabling the advanced filter, you'll need to update your Filter Expression to reflect the additions/removals.



Defining Field Actions

Finally, you'll need to select the field updates to take place or functions to call should a record match the criteria defined above. There are two distinct types of field actions that can take place:



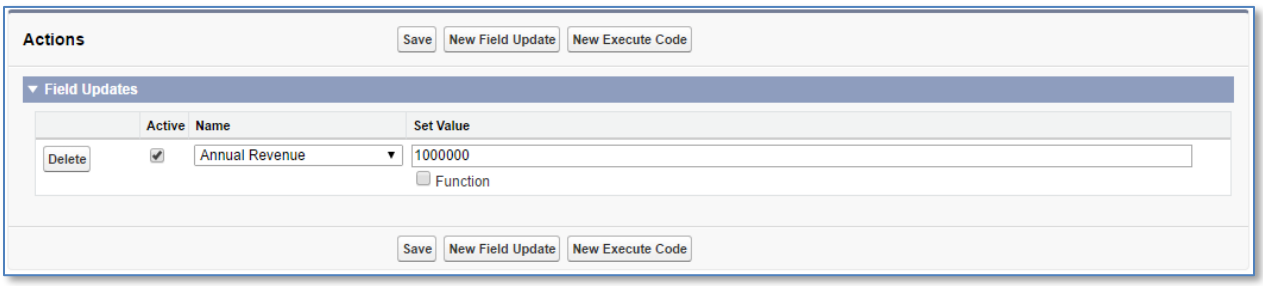
New Field Update

New Field Update should be used in the case that you want to set a specific field on the object with a defined value or have a field set via a function (either standard out of the box or custom)

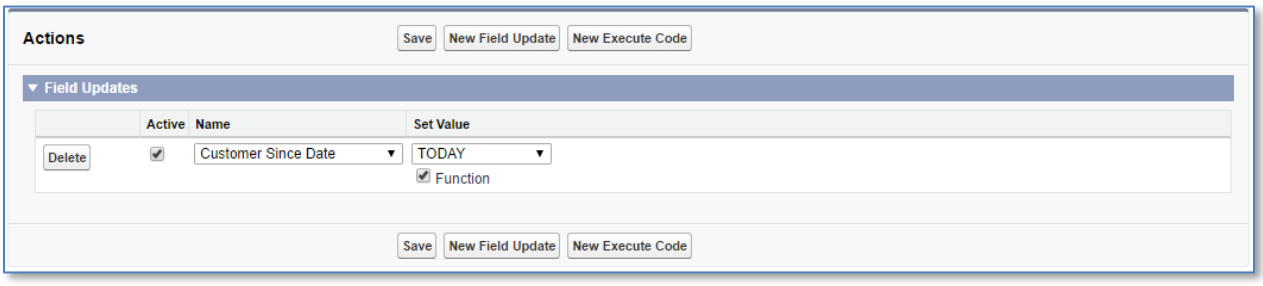
- Under the Actions section, click the ***New Field Update*** button
  - In the Name field, select the field which should be updated
  - In the Set Value field, depending on whether you selected a text/number, picklist (single or multi), enter a static value(s) which the field should be updated to
    - In the case of a lookup type field (Owner, Record Type, etc.), click the magnifying glass icon to the right of the value field and select the appropriate value via the pop-up window

- There is also the ability to set the value of a field to a function by simply checking the Function checkbox under the Set Value field. As an example, by using the "Stamp Rule Text" function, it allows you to set the value of a field to the name of the rule detail where the criteria was met for auditing purposes.

Standard Field



Function



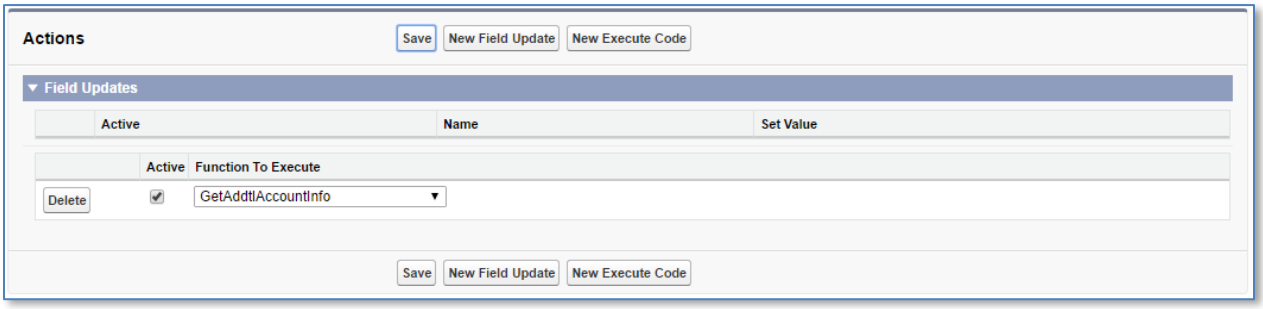
- If you have multiple fields that should be updated, simply click the ***New Field Update*** button again to add as many additional fields as necessary

New Execute Code

Execute Code should be used in the case that you want to have a function called if the criteria are met, but instead of setting a field on the record, you are executing Apex code defined for that function.

Important Note: The BREeze application does not come with any standard functions that can be used in this capacity. The application does, however, give you the ability to create your own custom functions within the application. For more information on how to implement functions within BREeze, see the Custom Function section in this document and the BREeze Custom Function Overview documentation.

- Under the Actions section, click the ***New Execute Code*** button
  - In the Function to Execute field, select the function that should be executed and click *Save*
  - If the function that has been selected requires extensions, click the wrench icon to the right of the function that was selected and enter the appropriate values into the function extension pop-up window and click *Save*. For more information on function extensions, see the Extending Functions section of this document.



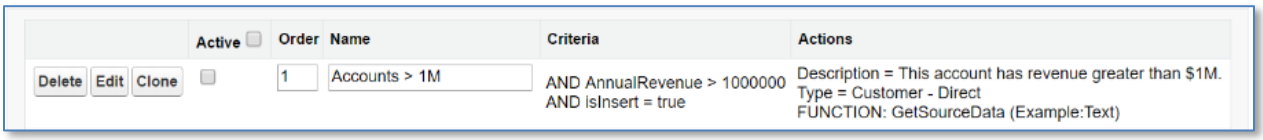
- If you have multiple functions that should be called, simply click the ***New Execute Code*** button again to add as many additional function calls as necessary

Important Note: You can enter any combination of both field updates & execute code entries to a single Actions section. If both field updates and execute code entries are present within the Actions section, should a record meet the criteria for this Rule Entry, all actions will take place.

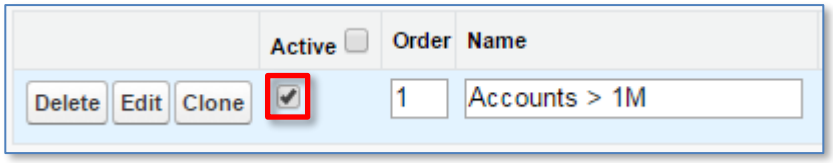
Finalizing Your Field Actions

- Once you have entered all your field updates and/or execute code entries, be sure to click the ***Save*** button at the top or bottom of the Actions section
- Next, click on the ***Back to Rule Maintenance*** link at the top of the screen

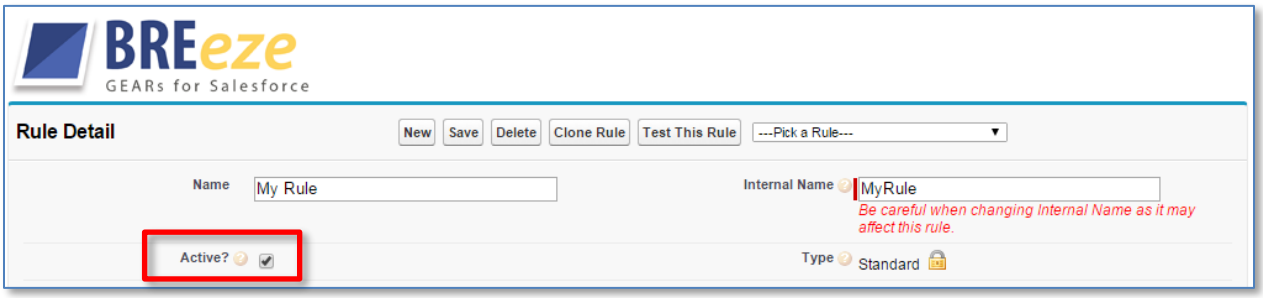
- This will bring you back to your original rule set-up screen, however now you should see the Rule Entry that you just defined with both the criteria and the field actions visible



- From here, you'll want to continue to define all your rule entries in the same manner
  - If you have a rule that is similar to another, simply use the ***Clone*** button to copy the Rule Entry definition. From there, click the ***Edit*** button on the cloned entry and adjust as needed
- Once all your rule entries are complete, be sure to check the Active checkbox within the Rule Entry itself or the checkbox to the right of the Active checkbox which selects all. Finally, be sure to click the ***Save Entries*** button after setting the rule entries to active.

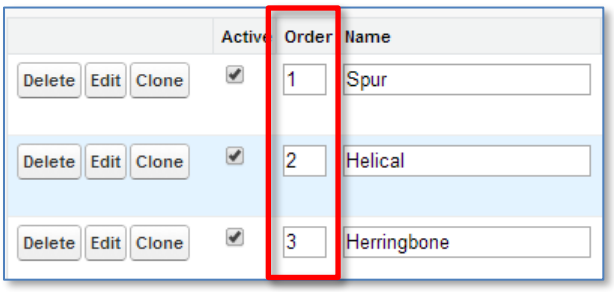


- When you are ready to have records begin flowing through your rule, ensure that the Active checkbox is checked for the overall Rule

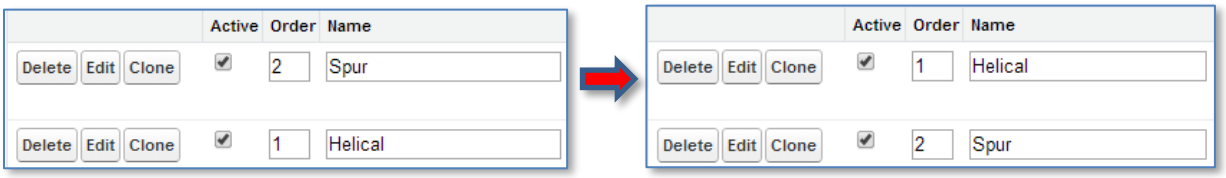


Rule Entry Ordering

BREeze evaluates the rule entries in the order which they are noted from the Rule Detail screen. This order can be altered at any point if you decide that a specific rule or rules should evaluate before others.



To adjust the ordering, simply adjust the number(s) to the desired evaluation sequence and click the ***Save Entries*** button at the top or bottom of the Rules Entries section of the screen



*This can be done one entry at a time or for several records simultaneously.

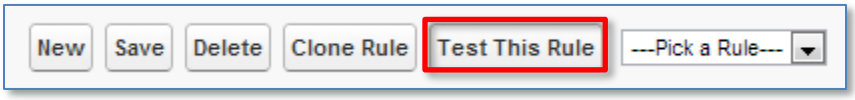
Rule Testing

Before deploying the rules you created in BREeze to a "live data mode", you'll want to test them to ensure they are updating not only the field(s) that you expect but also the populating value(s) you are expecting.

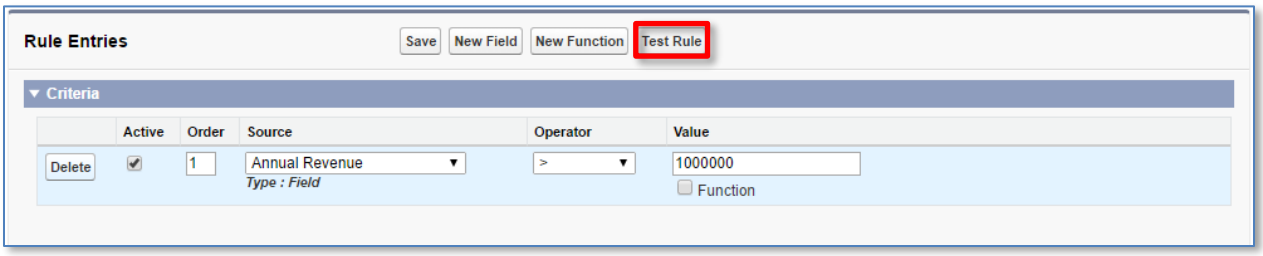
Important Notes:

1. The information and screenshots below are provided via a free add-on package that can be installed on top of the core BREeze package.
2. This is only a test so the record(s) you ran through the test will not actually have the updated information that is outlined in the Actions section. This will not occur until your Rule & Rule Entry have been activated and the record itself meets the criteria to be evaluated by BREeze.

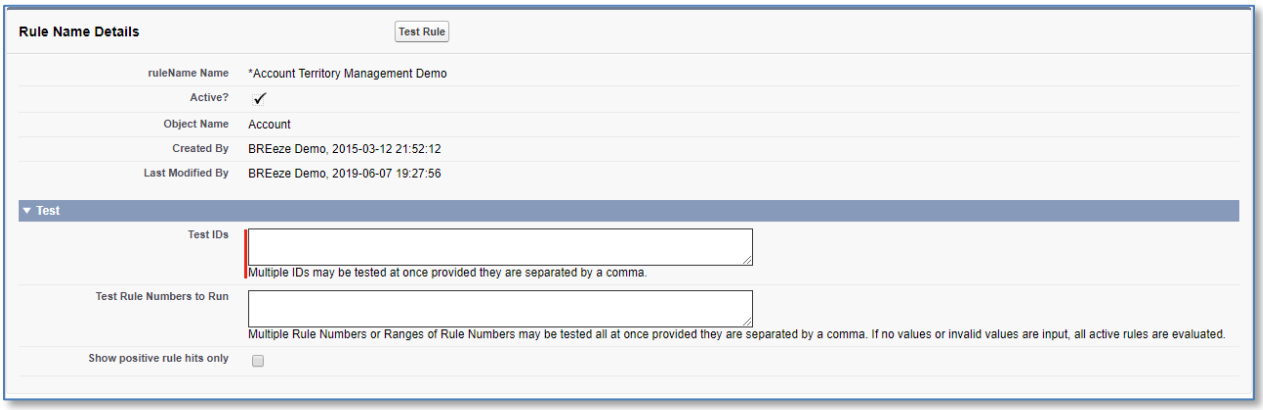
You can test your rule(s) in two places within BREeze. Each test screen will result in the same test executing so it's a matter of convenience where you execute the test from. The first is found on the main Rule Detail section of BREeze:



And the second is located on the Rule Entries screen above where you define the criteria for your rule:

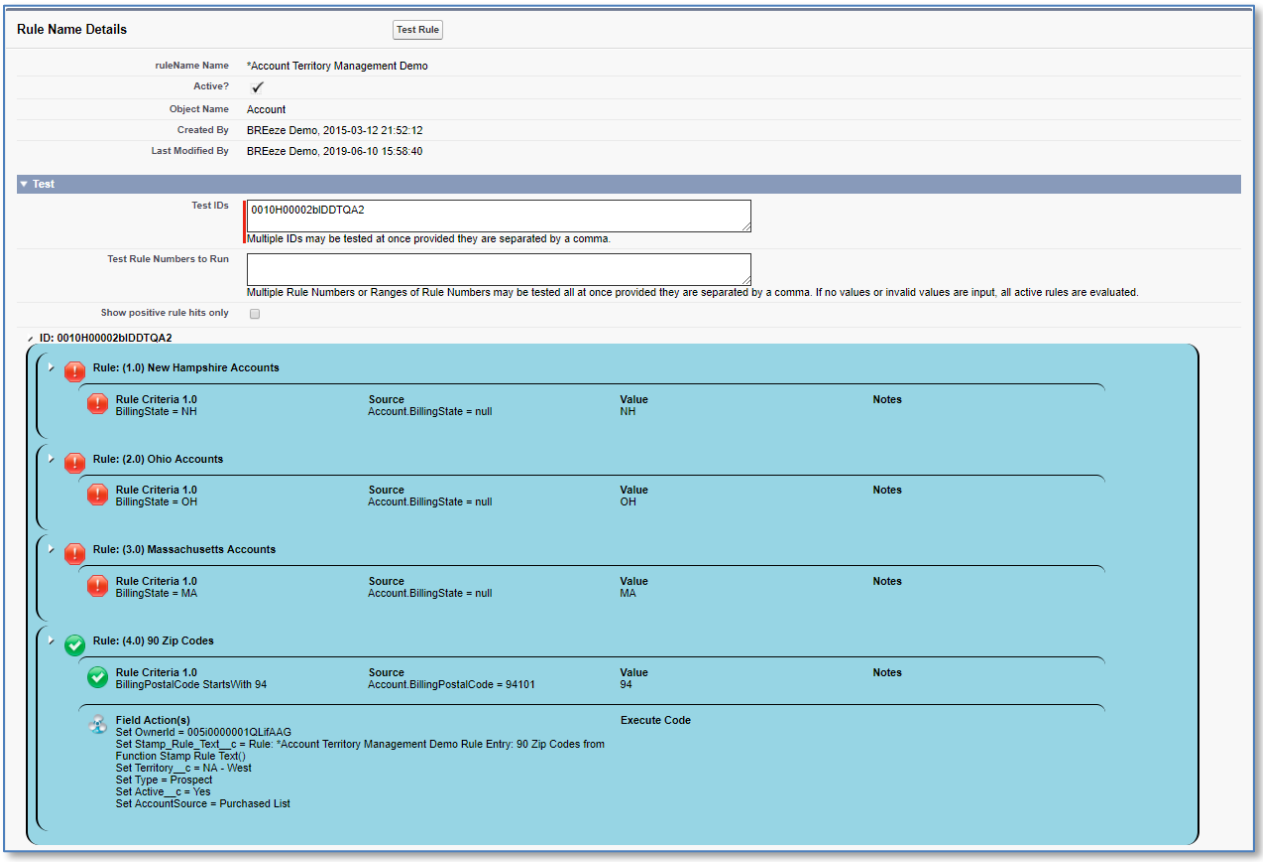


After clicking on either button, you'll be brought to the BREeze Test Rule page

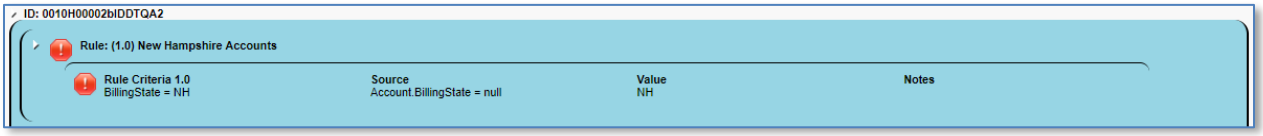


- ***Test IDs***
  - Enter a single Salesforce ID (15 or 18 characters) or multiple Salesforce ID's (separated by commas) for an existing record(s) of the object which you've created the rule for
- ***Test Rule Numbers to Run***
  - If you'd like to evaluate the only a subset of rules for a given record(s), enter the Rule Detail number (Ex. 4), Rule Detail Numbers (comma separated – Ex. 3, 5, 6) or range of Rule Detail numbers (Ex. 4-7)
- ***Show positive rule hits only***
  - If enabled, the results from the Test this Rule will only show where the record(s) evaluated to true and all instances where rules evaluated to false will be removed from view

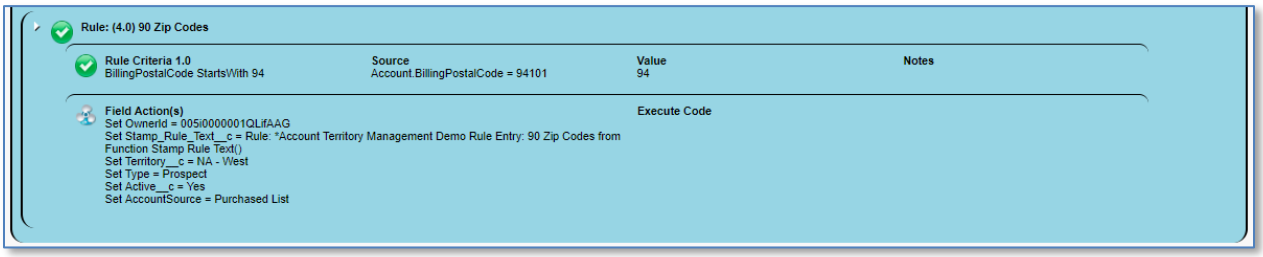
Depending on the number of rule entries, rule criteria, rule actions & when the criteria are first met; the results of your tests will vary. For example, below is a rule which had 6 Rule Entry records – each having 1 line of criteria and several actions.



From the test results, you can see the individual rules which were evaluated as either true or false depending on whether your test record met the criteria. For example, Rule 1 below called 'New Hampshire Accounts' ran first and evaluated whether the Billing State was equal to "NH". The actual data in the Billing State field was "CA" so the rule ultimately resulted in a false evaluation (indicated by the red exclamation mark icon).

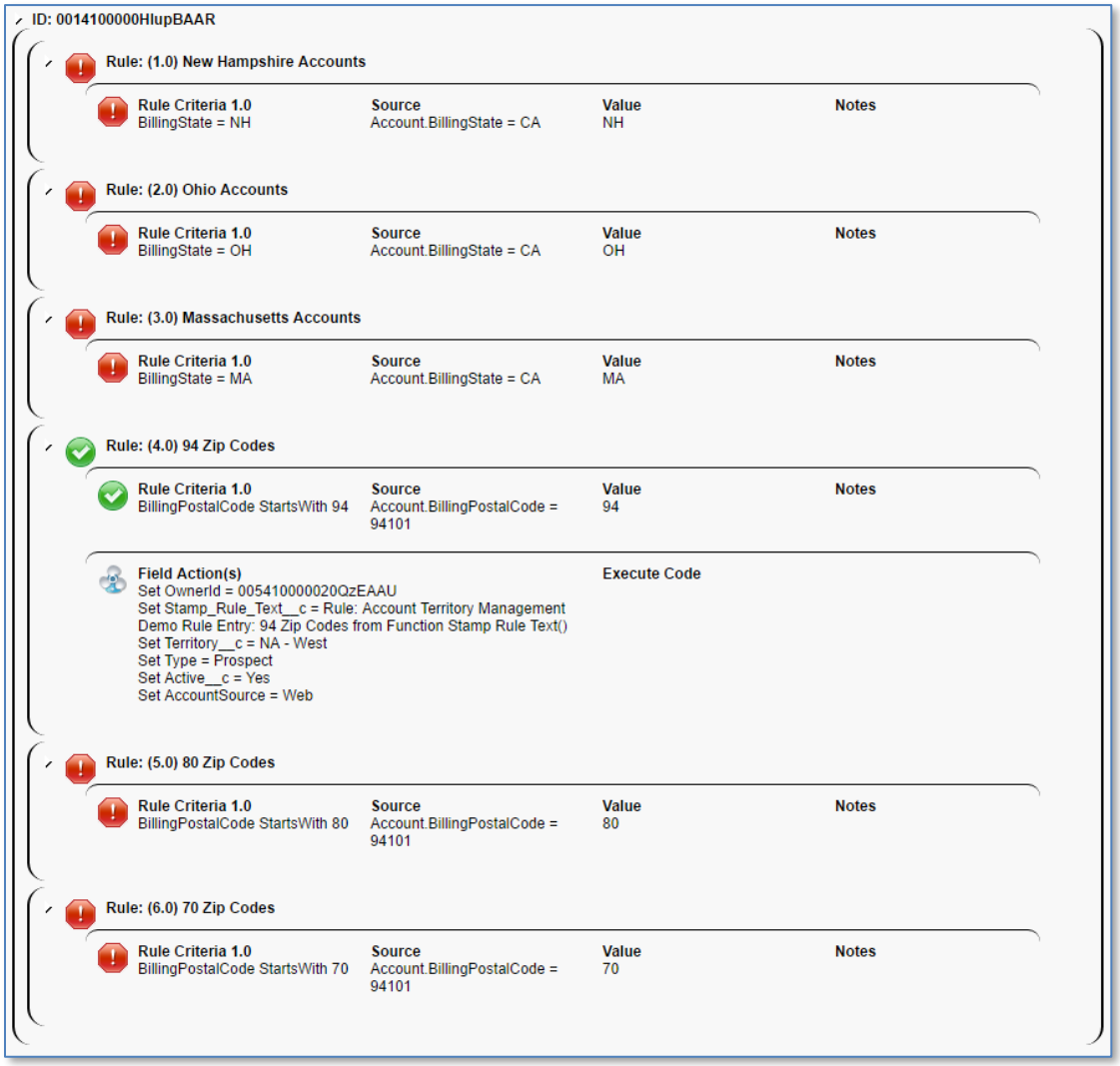


When a rule evaluates to false, BREeze moves to the next rule evaluates that in the same manner until BREeze ultimately hits a rule which evaluates to true (indicated by the green check mark icon).



If the '**Stop On First Match**' field is set to "true" (i.e. checked) on the main Rule screen, BREeze stops evaluating and executes on the actions that are associated to that rule detail entry. So even though there were 6 rule entries in this example, because the 4th rule's criteria were met, BREeze stopped evaluating any further. In this case, it sets a custom field called Territory, the Type, the Source, Active flag, the Owner and finally stamps a text field which the name of the Rule Entry which ultimately the record met the criteria for.

If the '**Stop On First Match**' field is set to "false" on the main Rule itself, BREeze would continue to run through the remainder of the rules and evaluate/assign should any of the other rule criteria evaluate to true. As you can see from the example below, Rule 4 evaluated to true and then BREeze continued to evaluate rules 5 & 6.



Standard BREeze Functions

Below is a list of the current functions which are available in BREeze for both the field criteria & update/assignment sections of the Rule Detail screen.

Criteria

Function	Definition

AddDaysToToday	Returns a date value based on the current date and the extensions which are defined:
	• *Include weekends (true/false)*
	  ○ Determines whether weekends should be included in the date calculation.
	• *End on a weekend (true/false)*
	  ○ Determines whether the date can fall on a weekend or whether the date should automatically be advanced to the following Monday.
	• *Return type (Date/DateTime)*
	  ○ Determines whether the field type to return is a Date or Date/Time. For this use case, it should always be "Date" as this function is only available for Date type fields.
	• *#days (integer)*
	  ○ Number of days (+/-) from the current date which should be evaluated.

AddDaysToTodayDT	Returns a date/time value based on the current date/time and the extensions which are defined:
	• *Include weekends (true/false)*
	  ○ Determines whether weekends should be included in the date calculation.
	• *End on a weekend (true/false)*
	  ○ Determines whether the date can fall on a weekend or whether the date should automatically be advanced to the following Monday.
	• *Return type (Date/DateTime)*
	  ○ Determines whether the field type to return is a Date or Date/Time. For this use case, it should always be "DateTime" as this function is only available for DateTime type fields.
	• *#days (integer)*
	  ○ Number of days (+/-) from the current date which should be evaluated.

NOW	Returns the current date/time

TODAY	Returns the current date
TOMORROW	Returns the current date plus 1 day
YESTERDAY	Returns the current date minus 1 day
getWithinCaseBusinessHours	Evaluates the defined case's business hours
getWithinOrgDefaultBusinessHours	Evaluates the org's default business hours
Lead:getDomain	Returns the domain value from an email address. **Note**: This is only available for 'Lead' object rules.

Actions/Assignment

Function	Definition
Stamp Rule Id	Identifies the ID of the rule which fired to update the current record and places that ID in the field specified. **Note**: Can only be used when the field name selected is of type Id.
Stamp Rule Text	Identifies the name of the rule which fired to update the current record and places that text in the field specified. **Note**: Can only be used when the field name selected is a text field.
TODAY	Sets the current date. **Note**: Can only be used when the field name selected is a date field.
TOMORROW	Sets tomorrow's date. **Note**: Can only be used when the field name selected is a date field.
YESTERDAY	Sets yesterday's date. **Note**: Can only be used when the field name selected is a date field.
NOW	Sets the current date/time. **Note**: Can only be used when the field name selected is a date/time field.
Lead:getDomain	Sets the domain from the email address on the record. **Note**: This can only be used on rules for the Lead object.
getWithinOrgDefaultBusinessHours	Sets the instances default business hours.
getWithinCaseBusinessHours	Sets the business hours for the case. **Note**: This can only be used on rules for the Case object.

AddDaysToToday	Sets a date value based on the current date and the extensions which are defined:
	• *Include weekends (true/false)*
	  ○ Determines whether weekends should be included in the date calculation.
	• *End on a weekend (true/false)*
	  ○ Determines whether the value can land on a weekend or whether the date should automatically be advanced to the following Monday.
	• *Return type (Date/DateTime)*
	  ○ Determines whether the field type to return is a Date or Date/Time. For this use case, it should always be "Date" as

	this function is only available for Date type fields. • *#days (integer)* Number of days (+/-) from the current date which should be evaluated.
AddDaysToTodayDT	Sets a date/time value based on the current date/time and the extensions which are defined: • *Include weekends (true/false)* ○ Determines whether weekends should be included in the date calculation. • *End on a weekend (true/false)* ○ Determines whether the date/time can land on a weekend or whether the date should automatically be advanced to the following Monday. • *Return type (Date/DateTime)* ○ Determines whether the field type to return is a Date or Date/Time. For this use case, it should always be "DateTime" as this function is only available for Date Time type fields. • *#days (integer)* ○ Number of days (+/-) from the current date which should be evaluated.

Custom Functions & Function Extensions

Custom functions can be created within BREeze to further extend the out-of-box BREeze functionality. All custom functions have both a development & a configuration component because a function is essentially the execution of an Apex class.

Important Note: Custom functions are advanced functionality and should only be attempted by those who have development experience and extensive Salesforce knowledge. If you would like assistance in building out Custom functions, please contact us at gearsdesignsupport@rafter.one to discuss a support engagement.

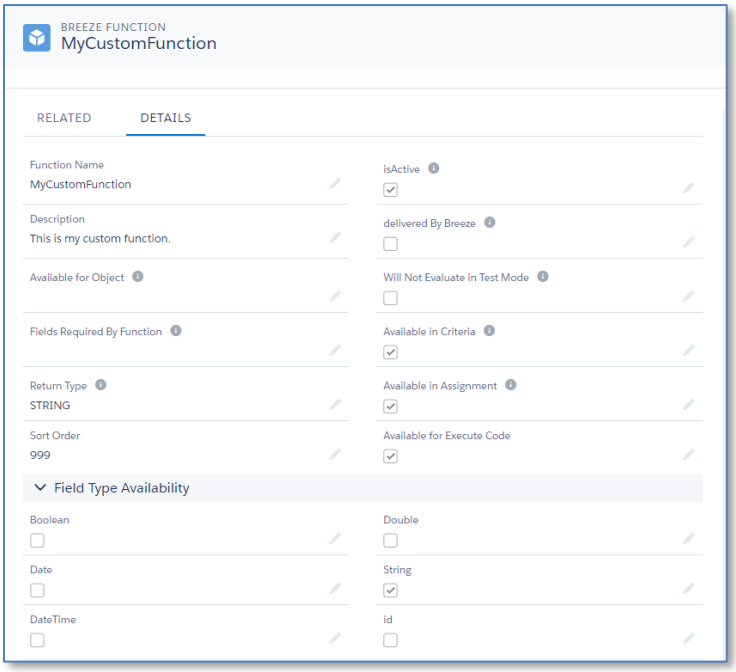
Creating Custom Functions

To create a custom function, follow these steps:

- Navigate to the BREeze Functions tab
- From the BREeze Functions tab, click the ***New*** button
- From the resulting screen, at a minimum enter the following:
    - **Function Name**
        - Enter a descriptive name for the function which you are building
    - **isActive**
        - Indicates that the function is active and can be used in a rule
    - **Return Type**
        - Indicates the field type which is returned by the function (double, string, date, etc.)
    - **Sort Order**
        - Indicates a numeric ordering sequence which controls where the function will appear in a dropdown list
    - **Class Name**
        - Reference to the Apex Class which executes the function logic
    - **Available in Criteria**
        - Indicates that the function will be available for use under the Criteria section of the Rule Details screen
            - This applies to only 'Standard' rules as criteria is not defined for 'Create Object' rules
    - **Available in Assignment**
        - Indicates that the function will be available for use under the Actions (Assignment) section of the Rule Details screen
            - This applies to both 'Standard' & 'Create Object' rules
    - **Available for Execute Code**
        - Indicates that the function will be available for use under the Actions (Assignment) section of the Rule Details screen when clicking on the *Execute Code* button
    - **Field Type Availability Section**
        - Indicates which field types the function will be available for:
            - **Boolean** → Checkbox field type
            - **Date** → Date field type

- **DateTime** → Date/Time field type
- **Double** → Number field type
- **String** → Text field type
- **Id** → Lookup/Id field type

- Optionally, you will likely want to consider populating the following fields:
  - **Available for Object**
    - Indicates which object(s) the function is available for use on
      - **Note**: This can simply be left null if the function can span across all or more than 1 Salesforce objects
  - **Interface PreProcess Required**
    - This should be checked in the case that the function needs to execute a pre-processing step prior to evaluating the criteria.
      - For example, a custom function which does an account name match from a lead record to an account record. To evaluate whether the function returns a true or false value, first the value from the lead record being processed in BREeze must be compared against all existing account records.
    - If this field is selected, you must also select "FunctionCheckerPrePost" in the 'InterfaceName' field.
  - **Interface PostProcess Required**
    - This should be checked in the case that a function needs to execute a post-processing step after the criteria have been met
      - For example, a custom function which adds or removes a public group from a user record. To execute that add/remove of the public group, you use a post-process function to accomplish this.
  - **InterfaceName**
    - Indicates whether the standard or modified Function class template will be used. If 'Interface PreProcess Required' is "True" then, this field must be set to "FunctionCheckerPrePost"
      - FunctionChecker – Uses the standard Function class template
      - FunctionCheckerPrePost – Extends the FunctionChecker class to allow for a pre- and post- process in the template
  - **Store Calculation**
    - If the function is not data specific, performance may be improved by executing the function once, storing the result, and referencing the results on each additional evaluation.
  - **Store Calculation By Record**
    - If the function is data specific, performance may be improved by executing the function once per record, storing the result, and referencing the results on each additional evaluation.
- Finally, click the ***Save*** button
- The result should look like something like this:



Extending Functions

In some cases, you may want to further extend functions by passing variables into them to either evaluate or set a specific value. This allows for greater flexibility when building functions and eliminating the need to create multiple functions for the same purpose where only the variable being passed into the function varies. Function Extensions allow you to not only streamline the functions being created but also allows for greater flexibility when your business requirements change.

An example of this, the out-of-box 'AddDaysToToday' & 'AddDaysToTodayTD' functions leverage extensions by taking the standard TODAY function and manipulating that value based on the variables that are entered.

When looking at the 'AddDaysToToday' function, you will see a related list which displays the four extensions being used in this function:

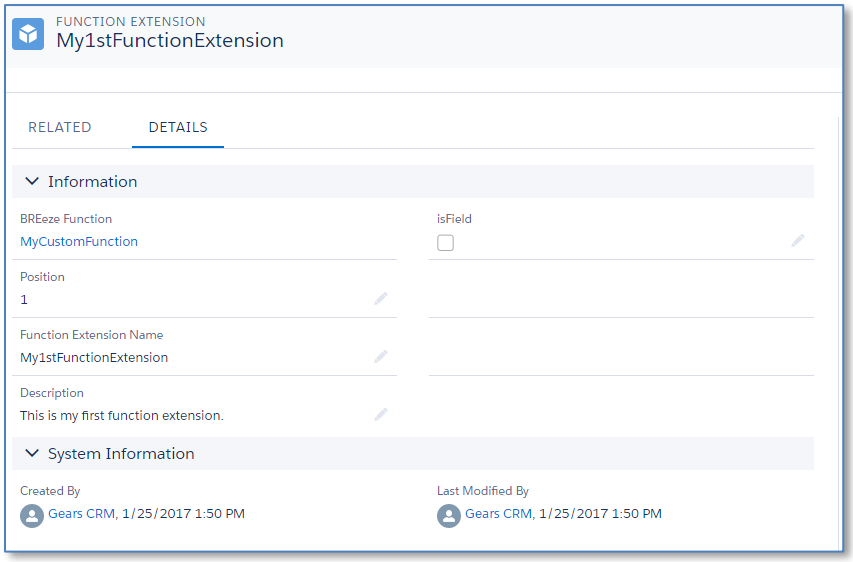
Based on the inputs entered within the BREeze rule itself, you can determine the value that is returned and used in either criteria or assignments.

Creating Function Extensions

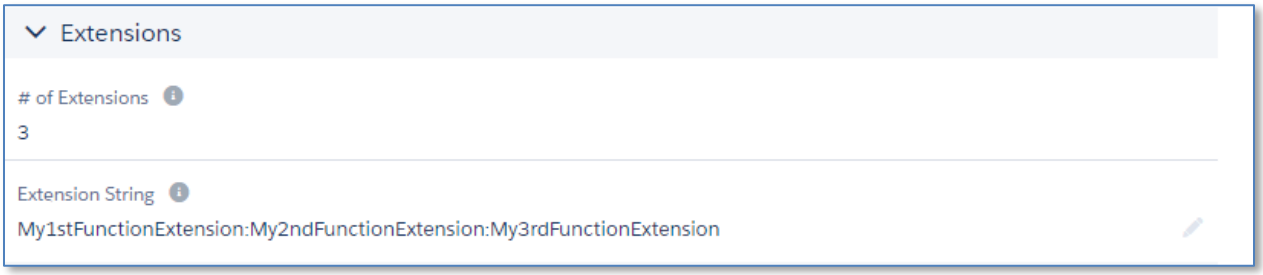
If you've created a function which has the need for additional function extensions to be added, follow these steps:

- Navigate to the custom function which you'd like to add extensions to
- Scroll to the 'Function Extensions' related list
- Click the *New Function Extension* button
- From the resulting screen, enter the following:
  - **Position**
    - This is the position which this extension should be passed into the custom function logic. Additionally, this defines the order in which the extensions will appear when defining the inputs on the rule itself.
  - **Function Extension Name**
    - This is the name of the function extension itself. This value will appear on the function extension entry screen, so you'll want to enter something that is easy to understand when setting up your rules.
    - **Note**: It's also helpful to add information in the name as to the type of data that is expected in this extension. For example:
      - "…(true/false)"
      - "…(Date/DateTime)"
      - "…(integer)"
      - Etc.
  - **Description**
    - This field is optional however it may be helpful to enter information here as to why this extension is being used.
  - **isField**
    - By default this is unchecked however in the case that you want to reference a specific field instead of a value, set this to true
  - **BREeze Function**

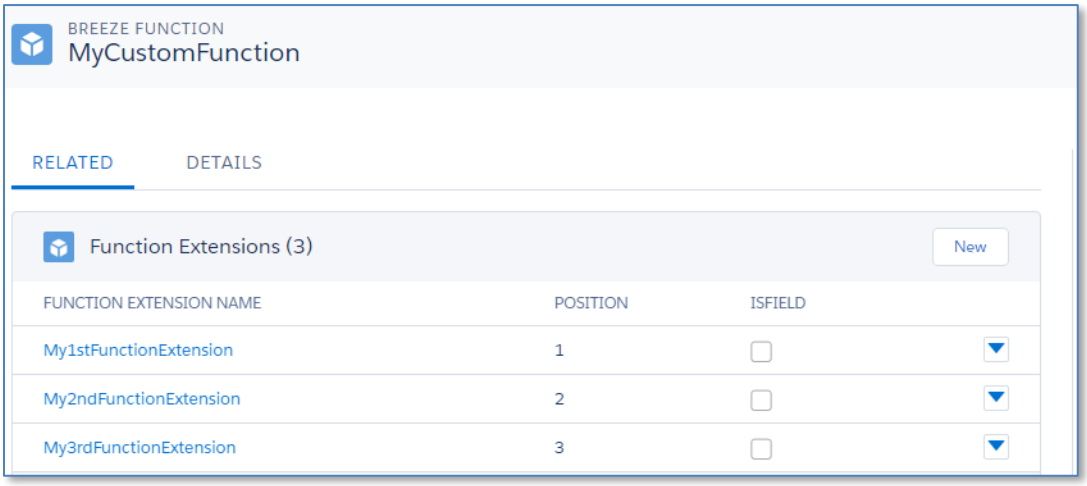
    - This should be pre-populated based on the function which you clicked the ***New Function Extension*** button from
- Finally, click the ***Save*** button or ***Save & New*** button (if you are creating additional extensions)
    - If creating multiple extensions, simply repeat the above process
- The result should look like something similar to this:



- Once all extensions have been created for the function, navigate back to the function itself
- Within the 'Extensions' section of the function itself, the information has been updated to reflect the extensions which were just added

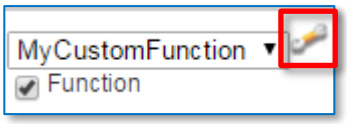


- Additionally, the related list will display all the extensions that are tied to this specific function:

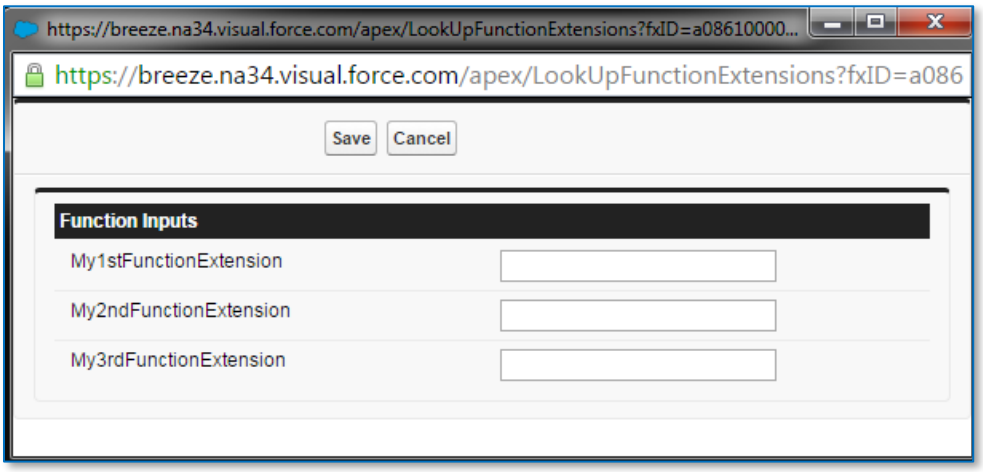


Setting Function Extension Values in Rule Criteria or Assignment

From the Rule Detail screen itself, when you select a function which has extensions available for it, a wrench icon will appear to the right of the value.



Upon clicking that wrench, a subsequent pop-up window will be shown. Within this pop-up window, it will display the function extensions available for that specific function:



By entering values into each of the extensions, these variables will be passed into the function and the rule will either be evaluated using these values or set a specific field based on these values.

About GearsDesign

At GearsDesign, we build products that help companies achieve breakthrough results with their Salesforce implementations. We know first-hand the challenges you face as a Salesforce administrator. Your implementation has to meet your organization's unique requirements. It also needs to stay tightly aligned with Sales and other operations. But your instance also must be able to turn on a dime when business conditions shift or your company's needs change. Sometimes, the changes you need to make go beyond what you're able to do with Salesforce's feature set. Your choices are either a risky and time-consuming customization project, or to just say 'no' and take the heat. We give you better options. GearsDesign solutions make it easy for you to extend and customize Salesforce's capabilities to achieve all your CRM goals. With a GearsDesign solution, any time you're confronted with a tough request, you'll be able to say with confidence, "I can make Salesforce do that."

For more information about GearsDesign or more information on any of our additional applications for Salesforce, visit https://gearsdesign.com/ or contact us at gearsdesignsupport@rafter.one.