Search Criteria

The search criteria that we have been using up to this point is "*:*"

Querying and returning every node is not what we need to solve our current problem.

Scenario: We want only to return a subset of our nodes--only the nodes that are webservers.

Search Syntax

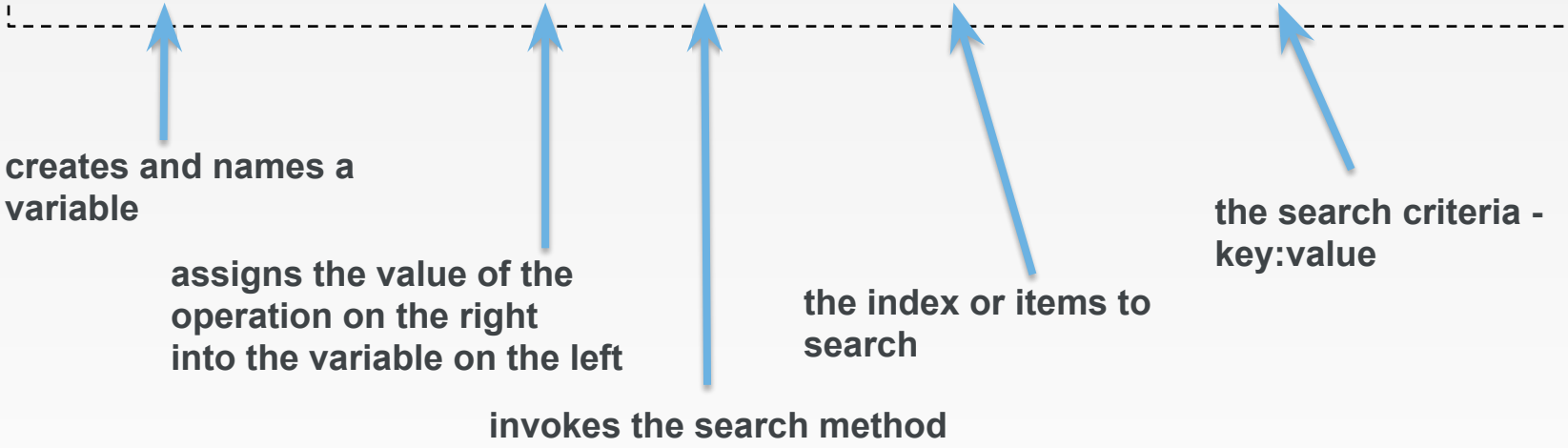
A search query is comprised of two parts: the key and the search pattern. A search query has the following syntax:

key:search_pattern

...where key is a field name that is found in the JSON description of an indexable object on the Chef server and search_pattern defines what will be searched for,

Search Syntax within a Recipe

```
all_web_nodes = search('node','role:web')
```



Search Syntax within a Recipe

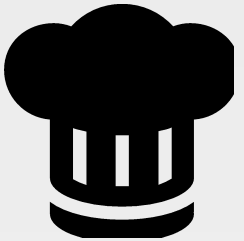
```
all_web_nodes = search('node','role:web')
```

Search the Chef Server for all node objects that have the role equal to 'web' and store the results into a local variable named "all_web_nodes'.

Hard Coding Example

~/chef-repo/cookbooks/myhaproxy/recipes/default.rb

```
node.default['haproxy']['members'] = [{
    'hostname'  => 'web1',
    'ipaddress' => '192.168.10.43',
    'port' => 80,
    'ssl_port' => 80
  },
  {
    'hostname'  => 'web2',
    'ipaddress' => '192.168.10.44',
    'port' => 80,
    'ssl_port' => 80
  }
]
include_recipe 'haproxy::manual'
```



Dynamic Web Load Balancer

Every time we create a web node we need to update our load balancer (myhaproxy) cookbook. That doesn't feel right!

Objective:

- ❑ Update the myhaproxy cookbook to dynamically use nodes with the web role

Showing web1 Attributes

```
$ knife node show web1 -a ipaddress
```

```
web1:
  ipaddress: 192.168.10.43
```

Showing web2 Attributes

```
$ knife node show web2 -a ipaddress
```

```
web2:
  ipaddress: 192.168.10.44
```


CHEF

NOTE: Showing web1 CLOUD Attributes

```
$ knife node show web1 -a cloud
```

```
web1:
  cloud:
    local_hostname:  ip-10-197-105-148.us-west-1.compute.internal
    local_ipv4:      10.197.105.148
    private_ips:     10.197.105.148
    provider:        ec2
    public_hostname: ec2-54-176-64-173.us-west-1.compute.amazonaws.com
    public_ips:      54.176.64.173
    public_ipv4:     54.176.64.173
```


CHEF

NOTE: Showing web1 CLOUD Attributes

```
$ knife node show web2 -a cloud
```

```
web2:
  cloud:
    local_hostname:  ip-10-197-105-149.us-west-1.compute.internal
    local_ipv4:      10.197.105.149
    private_ips:     10.197.105.149
    provider:        ec2
    public_hostname: ec2-54-176-64-174.us-west-1.compute.amazonaws.com
    public_ips:      54.176.64.174
    public_ipv4:     54.176.64.174
```


CHEF

GL: Remove the Hard-coded Members

~/chef-repo/cookbooks/myhaproxy/recipes/default.rb

```
node.default['haproxy']['members'] = [{
    'hostname'  => 'web1',
    'ipaddress' => '192.168.10.43',
    'port' => 80,
    'ssl_port' => 80
  },
  {
    'hostname'  => 'web2',
    'ipaddress' => '192.168.10.44',
    'port' => 80,
    'ssl_port' => 80
  }
]
include_recipe 'haproxy::manual'
```

GL: Use Search to Identify the Members

~/chef-repo/cookbooks/myhaproxy/recipes/default.rb

```
all_web_nodes = search('node','role:web')

#TODO: Convert all found nodes into hashes with ipaddress,
#      hostname, port, ssl_port
#TODO: Assign all the hashes to the node's haproxy members
#      attribute.

include_recipe 'haproxy::manual'
```

Creating an Array to Store the Converted Members

~/chef-repo/cookbooks/myhaproxy/recipes/default.rb

```
all_web_nodes = search('node','role:web')

members = []

#TODO: Convert all found nodes into hashes with ipaddress,
#      hostname, port, ssl_port

node.default['haproxy']['members'] = members

include_recipe 'haproxy::default'
```

Populating the Members with Each New Member

~/chef-repo/cookbooks/myhaproxy/recipes/default.rb

```
all_web_nodes = search('node','role:web')

members = []

all_web_nodes.each do |web_node|
  member = {}
  # TODO: Populate the hash with hostname, ipaddress, port, and
  #       ssl_port
  members.push(member)
end

node.default['haproxy']['members'] = members

include_recipe 'haproxy::manual'
```

Populating the Hash with Node Details

~/chef-repo/cookbooks/myhaproxy/recipes/default.rb

```
# ... BEFORE THE LOOP IN THE RECIPE ...

all_web_nodes.each do |web_node|
  member = {
    'hostname' => web_node['hostname'],
    'ipaddress' => web_node[‘ipaddress'],
    'port' => 80,
    'ssl_port' => 80
  }
  members.push(member)
end

# ... AFTER THE LOOP IN THE RECIPE ...
```

The Final Recipe

~/chef-repo/cookbooks/myhaproxy/recipes/default.rb

```
all_web_nodes = search('node','role:web')

members = []

all_web_nodes.each do |web_node|
  member = {
    'hostname' => web_node['hostname'],
    'ipaddress' => web_node[`ipaddress'],
    'port' => 80,
    'ssl_port' => 80
  }
  members.push(member)
end

node.default['haproxy']['members'] = members

include_recipe 'haproxy::manual'
```

NOTE: The Final Recipe - Cloud Instances

~/chef-repo/cookbooks/myhaproxy/recipes/default.rb

```
all_web_nodes = search('node','role:web')

members = []

all_web_nodes.each do |web_node|
  member = {
    'hostname' => web_node['cloud']['public_hostname'],
    'ipaddress' => web_node['cloud']['public_ipv4'],
    'port' => 80,
    'ssl_port' => 80
  }
  members.push(member)
end

node.default['haproxy']['members'] = members

include_recipe 'haproxy::manual'
```