SUFFIX Function

Finds the suffix value after the domain from a valid URL. Input values must be of URL or String type.This function is part of a set of functions for processing URL data.

- See *DOMAIN Function*.
- See *SUBDOMAIN Function*.
- For more information, see *Structure of a URL* .

Basic Usage

URL literal examples:

```
derive type:single value: SUFFIX('http://www.example.com' ) as: 'myDomain'
```

Output: Generates a column containing the value `com`.

```
derive type:single value: SUFFIX('http://www.exampl.e.com' ) as: 'myDomain'
```

Output: Generates a column containing the value `com`.

Column reference example:

```
derive type:single value: SUFFIX(myURLs) as: 'myDomain'
```

Output: Generates the new `myDomain` column containing the suffix values extracted from the `myURLs` column.

Syntax and Arguments

```
derive type:single value: SUFFIX(column_url)
```

Argument	Required?	Data Type	Description
column_url	Y	string	Name of column or String or URL literal containing the suffix value to extract

For more information on syntax standards, see *Language Documentation Syntax Notes*.

column_url

Name of the column or URL or String literal whose values are used to extract the suffix value.

- Missing input values generate missing results.
- Multiple columns and wildcards are not supported.

Usage Notes:

Required?	Data Type	Example Value
Yes	String literal or column reference (URL)	`http://www.example.com`

Copyright © 2021 Trifacta Inc.

Examples

> **Tip:** For additional examples, see *Common Tasks*.

Example - Domain, Subdomain, Host, and Suffix functions

This examples illustrates how you can extract component parts of a URL using the following functions:

- `DOMAIN` - extracts the domain value from a URL. See *DOMAIN Function*.
- `SUBDOMAIN` - extracts the first group after the protocol identifier and before the domain value. See *SUBDOMAIN Function*.
- `HOST` - returns the complete value of the host from an URL. See *HOST Function*.
- `SUFFIX` - extracts the suffix of a URL. See *SUFFIX Function*.
- `URLPARAMS` - extracts the query parameters and values from a URL. See *URLPARAMS Function*.
- `FILTEROBJECT` - filters an Object value to show only the elements for a specified key. See *FILTEROBJECT Function*.

Source:

Your dataset includes the following values for URLs:

URL
www.example.com
example.com/support
http://www.example.com/products/
http://1.2.3.4
https://www.example.com/free-download
https://www.example.com/about-us/careers
www.app.example.com
www.some.app.example.com
some.app.example.com
some.example.com
example.com
http://www.example.com?q1=broken%20record
http://www.example.com?query=khakis&app=pants
http://www.example.com?q1=broken%20record&q2=broken%20tape&q3=broken%20wrist

Transform:

When the above data is imported into the application, the column is recognized as a URL. All values are registered as valid, even the IPv4 address.

To extract the domain and subdomain values:

```
derive type:single value: DOMAIN(URL) as: 'domain_URL'
```

```
derive type:single value: SUBDOMAIN(URL) as:'subdomain_URL'
```

Copyright © 2021 Trifacta Inc.

```
derive type:single value: HOST(URL) as:host_URL'
```

```
derive type:single value: SUFFIX(URL) as:'suffix_URL'
```

You can use the Trifacta® pattern in the following transform to extract protocol identifiers, if present, into a new column:

```
extract col:URL on:`{start}%*://`
```

To clean this up, you might want to rename the column to `protocol_URL`.

To extract the path values, you can use the following regular expression:

> **NOTE:** Regular expressions are considered a developer-level method for pattern matching. Please use them with caution. See *Text Matching*.

```
extract col: URL on: /[^*:\/\/]\/.*$/
```

The above transform grabs a little too much of the URL. If you rename the column to `path_URL`, you can use the following regular expression to clean it up:

```
extract col:path_URL on:/[!^\/].*$/
```

Drop the `path_URL` column and rename the `path_URL1` column to the dropped one. Then:

```
derive type:single value: URLPARAMS(URL) as: 'urlParams'
```

If you wanted to just see the values for the `q1` parameter, you could add the following:

```
derive type:single value: FILTEROBJECT(urlParams,'q1') as: 'urlParam_q1'
```

Results:

URL	host_URL	path_URL	protocol_URL	subdomain_URL	domain_URL	suffix_URL	urlParams	urlParam_q
www.example.com	www.example.com			www	example	com		
example.com/support	example.com	/support			example	com		
http://www.example.com/products/	www.example.com	/products/	http://	www	example	com		
http://1.2.3.4	1.2.3.4		http://					

Copyright © 2021 Trifacta Inc.

https://www.example.com/free-download	www.example.com	/free-download	https://	www	example	com		
https://www.example.com/about-us/careers	www.example.com	/about-us/careers	https://	www	example	com		
www.app.example.com	www.app.example.com			www.app	example	com		
www.some.app.example.com	www.some.app.example.com			www.some.app	example	com		
some.app.example.com	some.app.example.com			some.app	example	com		
some.example.com	some.example.com			some	example	com		
example.com	example.com				example	com		
http://www.example.com?q1=broken%20record	www.example.com		http://	www	example	com	{"q1":"broken record"}	{"q1":"broken record"}
http://www.example.com?query=khakis&app=pants	www.example.com		http://	www	example	com	{"query":"khakis","app":"pants"}	
http://www.example.com?q1=broken%20record&q2=broken%20tape&q3=broken%20wrist	www.example.com		http://	www	example	com	{"q1":"broken record", "q2":"broken tape", "q3":"broken wrist"}	{"q1":"broken record"}

Copyright © 2021 Trifacta Inc.

Copyright © 2021 Trifacta Inc.