

# Package: soql (via r-universe)

September 1, 2024

**Type** Package

**Title** Helps Make Socrata Open Data API Calls

**Version** 0.1.1

**Date** 2016-03-06

**Suggests** magrittr

**Author** Zeb Burke-Conte

**Maintainer** ``Zeb Burke-Conte" <zmbc@uw.edu>

**Description** Used to construct the URLs and parameters of 'Socrata Open Data API' <<https://dev.socrata.com>> calls, using the API's 'SoQL' parameter format. Has method-chained and sensical syntax. Plays well with pipes.

**License** MIT + file LICENSE

**Repository** <https://zmbc.r-universe.dev>

**RemoteUrl** <https://github.com/zmbc/soql>

**RemoteRef** HEAD

**RemoteSha** 5aae3b842bdf495ea128539dedc0e6e000181f5c

## Contents

soql-package . . . . .	2
soql . . . . .	3
soql_add_endpoint . . . . .	3
soql_limit . . . . .	4
soql_offset . . . . .	5
soql_queries . . . . .	6
soql_simple_filter . . . . .	7

<b>Index</b>	<b>9</b>
--------------	----------

---

soql-package

*Helps Make Socrata Open Data API Calls*

---

## Description

Used to construct the URLs and parameters of 'Socrata Open Data API' <<https://dev.socrata.com>> calls, using the API's 'SoQL' parameter format. Has method-chained and sensical syntax. Plays well with pipes.

## Details

To create a SoQL URL, or just parameters for one, start with `soql()`. Then chain the result into other functions, such as `soql_where()` or `soql_order()`. When you're done, use `as.character()` to retrieve the finished URL, for use with any networking package.

## Author(s)

Zeb Burke-Conte

Maintainer: "Zeb Burke-Conte" <[zmbc@uw.edu](mailto:zmbc@uw.edu)>

## References

[Documentation for the SODA API](#)

## Examples

```
if (require(magrittr)) {
  # With pipes
  my_url <- soql() %>%
    soql_where("height > 30") %>%
    soql_limit(20) %>%
    as.character()
} else {
  # Without pipes
  soql_chain <- soql()
  soql_chain <- soql_where(soql_chain, "height > 30")
  soql_chain <- soql_limit(20)
  my_url <- as.character(soql_chain)
}
```

---

soql	<i>Create a soql object</i>
------	-----------------------------

---

### Description

This is the constructor for soql objects. It will most often be called with no parameters, and be used at the start of a chain of functions. It is necessary to use this function before any others in the soql package.

### Usage

```
soql(query = "")
```

### Arguments

query	An optional string containing an already-formed URL. This URL is converted into a soql object.
-------	--

### Value

Returns a soql object.

### Examples

```
soql()

soql("a.socrata.endpoint?$select=*&$order=height")
```

---

soql_add_endpoint	<i>Add SODA API endpoint</i>
-------------------	------------------------------

---

### Description

Add an endpoint to an already-existing soql object.

### Usage

```
soql_add_endpoint(soql_list, endpoint)
```

### Arguments

soql_list	The soql object. If you don't have one yet, use the soql() function first. This can be piped in.
endpoint	The endpoint should be the URL of the data, without any parameters.

**Value**

Returns a new soql object, with the endpoint added, for use in other functions.

**References**

[Socrata's documentation on what an endpoint is](#)

**See Also**

[soql](#)

**Examples**

```
if (require(magrittr)) {
  # With pipes
  my_url <- soql() %>%
    soql_add_endpoint("https://fake.soda.api/resource.json") %>%
    as.character()
} else {
  # Without pipes
  soql_chain <- soql()
  soql_chain <- soql_add_endpoint(soql_chain, "https://fake.soda.api/resource.json")
  my_url <- as.character(soql_chain)
}
```

---

soql\_limit

*Limit the number of results from a SODA response*

---

**Description**

Adds a parameter to the SODA URL that limits how many responses the API will send back.

**Usage**

```
soql_limit(soql_list, limit)
```

**Arguments**

soql_list	The soql object. If you don't have one yet, use the soql() function first. This can be piped in.
limit	Number of records desired.

**Value**

Returns a new soql object, with a limit parameter added, for use in other functions.

**References**

[Documentation on the SODA website](#)

**See Also**[soql\\_offset](#)**Examples**

```
if (require(magrittr)) {
  # With pipes
  my_url <- soql() %>%
    soql_limit(5) %>%
    as.character()
} else {
  # Without pipes
  soql_chain <- soql()
  soql_chain <- soql_limit(soql_chain, 5)
  my_url <- as.character(soql_chain)
}
```

---

`soql_offset`*Control which records you receive from a SODA API*

---

**Description**

This function adds a parameter to a `soql` object that controls what index the returned records start at. For more information, view the SODA documentation linked in the References section below.

**Usage**

```
soql_offset(soql_list, offset)
```

**Arguments**

<code>soql_list</code>	The <code>soql</code> object. If you don't have one yet, use the <code>soql()</code> function first. This can be piped in.
<code>offset</code>	Desired starting index of responses.

**Value**

Returns a new `soql` object, with an `offset` parameter added, for use in other functions.

**References**

[Documentation on the SODA website](#)

**See Also**[soql\\_limit](#)

**Examples**

```

if (require(magrittr)) {
  # With pipes
  my_url <- soql() %>%
    soql_offset(50) %>%
    as.character()
} else {
  # Without pipes
  soql_chain <- soql()
  soql_chain <- soql_offset(soql_chain, 50)
  my_url <- as.character(soql_chain)
}

```

---

soql\_queries

*SoQL Queries*


---

**Description**

Wrapper functions around the SODA API's queries: select, where, order, group, and q.

**Usage**

```

soql_select(soql_list, select_clause)
soql_where(soql_list, where_clause)
soql_order(soql_list, column, desc = FALSE)
soql_group(soql_list, group_clause)
soql_q(soql_list, q_clause)

```

**Arguments**

soql_list	The soql object. If you don't have one yet, use the soql() function first. This can be piped in.
select_clause, where_clause, group_clause, q_clause	String to be used as the given clause in the query.
column	Column name to be ordered by.
desc	Whether to order descending.

**Value**

Returns a new soql object, with parameters added, for use in other functions.

**References**

[Documentation about these queries on the SODA website](#)

**See Also**

[soql\\_simple\\_filter](#) for an easier method of doing where with equality.

## Examples

```
if (require(magrittr)) {
  # With pipes
  my_url <- soql() %>%
    soql_select("height,weight") %>%
    soql_where("height > 30") %>%
    soql_order("height", desc=TRUE) %>%
    soql_group("type") %>%
    soql_q("a") %>%
    as.character()
} else {
  # Without pipes
  soql_chain <- soql()
  soql_chain <- soql_select(soql_chain, "height,weight")
  soql_chain <- soql_where(soql_chain, "height > 30")
  soql_chain <- soql_order(soql_chain, "height", desc=TRUE)
  soql_chain <- soql_group(soql_chain, "type")
  soql_chain <- soql_q(soql_chain, "a")
  my_url <- as.character(soql_chain)
}
```

---

soql\_simple\_filter      *Create a simple equality filter*

---

## Description

This function adds a parameter to the URL to filter records in a simple way, using equality only. For more advanced filters, see [soql\\_where](#).

## Usage

```
soql_simple_filter(soql_list, column, value)
```

## Arguments

soql_list	The soql object. If you don't have one yet, use the soql() function first. This can be piped in.
column	The column name to filter by.
value	The value the column must be equal to.

## Value

Returns a new soql object, with a filter parameter added, for use in other functions.

## References

[Documentation on the SODA website](#)

**See Also**[soql\\_where](#)**Examples**

```
if (require(magrittr)) {  
  # With pipes  
  my_url <- soql() %>%  
    soql_simple_filter("height", 50) %>%  
    as.character()  
} else {  
  # Without pipes  
  soql_chain <- soql()  
  soql_chain <- soql_simple_filter(soql_chain, "height", 50)  
  my_url <- as.character(soql_chain)  
}
```



# Index

## \* package

soql-package, 2

soql, 3, 4

soql-package, 2

soql\_add\_endpoint, 3

soql\_group (soql\_queries), 6

soql\_limit, 4, 5

soql\_offset, 5, 5

soql\_order (soql\_queries), 6

soql\_q (soql\_queries), 6

soql\_queries, 6

soql\_select (soql\_queries), 6

soql\_simple\_filter, 6, 7

soql\_where, 7, 8

soql\_where (soql\_queries), 6