Google Cloud Bigtable Documentation

Release 0.0.1

Danny Hermes, Google Cloud Platform

July 31, 2015

Contents

1 List Clusters **3**

2 List Zones **5**

3 Create a new Cluster **7**

4 Check on Current Operation **9**

5 Get metadata for an existing Cluster **11**

6 Update an existing Cluster **13**

7 Delete an existing Cluster **15**

8 Undelete a deleted Cluster **17**
8.1 Documented Modules 17
8.2 Indices and tables 29

Python Module Index **31**

To use the API, the *Client* class defines a high-level interface which handles authorization and creating other objects:

```
from gcloud_bigtable.client import Client
cluster = Client()
```

This will use the Google Application Default Credentials if you don't pass any credentials of your own.

The Cluster Admin API has been fully implemented. Create a *Cluster* to get a high-level interface to cluster management:

```
cluster = client.cluster(zone, cluster_id)
```

CHAPTER 1

List Clusters

If you want a comprehensive list of all existing clusters, make a ListClusters request:

```
clusters = client.list_clusters()
```

This will return a list of *Cluster* s.

CHAPTER 2

List Zones

If you aren't sure which `zone` to create a cluster in, find out which zones your project has access to with a ListZones request:

```
zones = client.list_clusters()
```

This will return a list of `string` s.

CHAPTER 3

Create a new Cluster

After creating the cluster object, make a CreateCluster API request:

```
cluster.display_name = 'My very own cluster'
cluster.create()
```

If you would like more than the minimum number of nodes (3) in your cluster:

```
cluster.serve_nodes = 10
cluster.create()
```

Note: When modifying a cluster (via a CreateCluster, UpdateCluster or UndeleteCluster request), the Bigtable API will return a long-running Operation. This will be stored on the object after each of *create()*, *update()* and *undelete()* are called.

CHAPTER 4

Check on Current Operation

You can check if a long-running operation (for a *create()*, *update()* or *undelete()*) has finished by making a GetOperation request:

```
>>> cluster.operation_finished()
True
```

Note: The operation data is stored in protected fields on the *Cluster*: `_operation_type`, `_operation_id` and `_operation_begin`. If these are unset, then *operation_finished()* will fail. Also, these will be removed after a long-running operation has completed (checked via this method). We could easily surface these properties publicly, but it's unclear if end-users would need them.

CHAPTER 5

Get metadata for an existing Cluster

After creating the cluster object, make a GetCluster API request:

```
cluster.reload()
```

This will load `serve_nodes` and `display_name` for the existing `cluster` in addition to the `cluster_id`, `zone` and `project_id` already set on the *`Cluster`* object.

CHAPTER 6

Update an existing Cluster

After creating the cluster object, make an UpdateCluster API request:

```
client.display_name = 'New display_name'
cluster.update()
```

CHAPTER 7

Delete an existing Cluster

Make a DeleteCluster API request:

```
cluster.delete()
```

CHAPTER 8

Undelete a deleted Cluster

Make a UndeleteCluster API request:

```
cluster.undelete()
```

8.1 Documented Modules

8.1.1 Constants

Constants for Google Cloud Bigtable API.

gcloud_bigtable.constants.**CLUSTER_ADMIN_HOST = 'bigtableclusteradmin.googleapis.com'**
Cluster Admin API request host.

gcloud_bigtable.constants.**CLUSTER_ADMIN_PORT = 443**
Cluster Admin API request port.

gcloud_bigtable.constants.**TABLE_ADMIN_HOST = 'bigtabletableadmin.googleapis.com'**
Table Admin API request host.

gcloud_bigtable.constants.**TABLE_ADMIN_PORT = 443**
Table Admin API request port.

8.1.2 Data Connection

Connection to Google Cloud Bigtable Data API.

gcloud_bigtable.data_connection.**DATA_API_HOST = 'bigtable.googleapis.com'**
Data API request host.

class gcloud_bigtable.data_connection.**DataConnection**(*credentials=None*)
Bases: object

Connection to Google Cloud Bigtable Data API.

Enables interaction with data in an existing table.

Parameters **credentials** (oauth2client.client.OAuth2Credentials or NoneType) – The OAuth2 Credentials to use for this connection.

READ_ONLY_SCOPE = 'https://www.googleapis.com/auth/cloud-bigtable.data.readonly'
Read-only scope for data API requests.

SCOPE = 'https://www.googleapis.com/auth/cloud-bigtable.data'
Scope for data API requests.

check_and_mutate_row(*table_name*, *row_key*)
Checks and mutates a row.

mutate_row(*table_name*, *row_key*)
Mutates a row.

read_modify_write_row(*table_name*, *row_key*)
Reads, modifies and writes a row.

read_rows(*table_name*, *row_key=None*, *row_range=None*, *filter_=None*, *allow_row_interleaving=None*, *num_rows_limit=None*, *timeout_seconds=10*)
Read rows from table.

Streams back the contents of all requested rows, optionally applying the same Reader filter to each. Depending on their size, rows may be broken up across multiple responses, but atomicity of each row will still be preserved.

Note: If neither `row_key` nor `row_range` is set, reads from all rows. Otherwise, at most one of `row_key` and `row_range` can be set.

Parameters

- **table_name** (*string*) – The name of the table we are reading from. Must be of the form "projects/../zones/../clusters/../tables/.." Since this is a low-level class, we don't check this, rather we expect callers to pass correctly formatted data.
- **row_key** (*bytes*) – (Optional) The key of a single row from which to read.
- **row_range** (`_generated.bigtable_data_pb2.RowRange`) – (Optional) A range of rows from which to read.
- **filter** (`_generated.bigtable_data_pb2.RowFilter`) – (Optional) The filter to apply to the contents of the specified row(s). If unset, reads the entire table.
- **allow_row_interleaving** (*boolean*) – (Optional) By default, rows are read sequentially, producing results which are guaranteed to arrive in increasing row order. Setting "allow_row_interleaving" to true allows multiple rows to be interleaved in the response stream, which increases throughput but breaks this guarantee, and may force the client to use more memory to buffer partially-received rows.
- **num_rows_limit** (*integer*) – (Optional) The read will terminate after committing to N rows' worth of results. The default (zero) is to return all results. Note that if "allow_row_interleaving" is set to true, partial results may be returned for more than N rows. However, only N "commit_row" chunks will be sent.
- **timeout_seconds** (*integer*) – Number of seconds for request time-out. If not passed, defaults to `TIMEOUT_SECONDS`.

Return type `bigtable_service_messages_pb2.ReadRowsResponse`

Returns The response returned by the backend.

sample_row_keys(*table_name*, *timeout_seconds=10*)
Returns a sample of row keys in the table.

The returned row keys will delimit contiguous sections of the table of approximately equal size, which can be used to break up the data for distributed tasks like mapreduces.

Parameters

- **table_name** (*string*) – The name of the table we are taking the sample from. Must be of the form "projects/../zones/../clusters/../tables/.." Since this is a low-level class, we don't check this, rather we expect callers to pass correctly formatted data.
- **timeout_seconds** (*integer*) – Number of seconds for request time-out. If not passed, defaults to `TIMEOUT_SECONDS`.

Return type `messages_pb2.SampleRowKeysResponse`

Returns The sample row keys response returned.

8.1.3 Client

Parent client for calling the Google Cloud Bigtable API.

This is the base from which all interactions with the API occur.

In the hierarchy of API concepts * a client owns a cluster * a cluster owns a table * a table owns column families * a table owns data

`gcloud_bigtable.client.`**ADMIN_SCOPE = 'https://www.googleapis.com/auth/cloud-bigtable.admin'**
Scope for interacting with the Cluster Admin and Table Admin APIs.

class `gcloud_bigtable.client.`**Client**(*credentials=None*, *project_id=None*, *read_only=False*, *admin=False*, *user_agent='gcloud-bigtable-python'*, *timeout_seconds=10*)
Bases: `object`

Client for interacting with Google Cloud Bigtable API.

Parameters

- **credentials** (`oauth2client.client.OAuth2Credentials` or `NoneType`) – (Optional) The OAuth2 Credentials to use for this cluster. If not provided, defaulst to the Google Application Default Credentials.
- **project_id** (*string*) – (Optional) The ID of the project which owns the clusters, tables and data. If not provided, will attempt to determine from the environment.
- **read_only** (*boolean*) – (Optional) Boolean indicating if the data scope should be for reading only (or for writing as well). Defaults to `False`.
- **admin** (*boolean*) – (Optional) Boolean indicating if the client will be used to interact with the Cluster Admin or Table Admin APIs. This requires the `ADMIN_SCOPE`. Defaults to `False`.
- **user_agent** (*string*) – (Optional) The user agent to be used with API request. Defaults to `DEFAULT_USER_AGENT`.
- **timeout_seconds** (*integer*) – Number of seconds for request time-out. If not passed, defaults to `DEFAULT_TIMEOUT_SECONDS`.

Raises `ValueError` if both `read_only` and `admin` are `True`

cluster(*zone*, *cluster_id*, *display_name=None*, *serve_nodes=3*)
Factory to create a cluster associated with this client.

Parameters

- **zone** (*string*) – The name of the zone where the cluster resides.
- **cluster_id** (*string*) – The ID of the cluster.

- **display_name** (*string*) – (Optional) The display name for the cluster in the Cloud Console UI. (Must be between 4 and 30 characters.) If this value is not set in the constructor, will fall back to the cluster ID.
- **serve_nodes** (*integer*) – (Optional) The number of nodes in the cluster. Defaults to 3.

Return type `Cluster`

Returns The cluster owned by this client.

credentials
Getter for client's credentials.

Return type `oauth2client.client.OAuth2Credentials`

Returns The credentials stored on the client.

classmethod from_service_account_json(*json_credentials_path*, *project_id=None*, *read_only=False*, *admin=False*)
Factory to retrieve JSON credentials while creating client object.

Parameters

- **json_credentials_path** (*string*) – The path to a private key file (this file was given to you when you created the service account). This file must contain a JSON object with a private key and other credentials information (downloaded from the Google APIs console).
- **project_id** (*string*) – The ID of the project which owns the clusters, tables and data. Will be passed to `Client` constructor.
- **read_only** (*boolean*) – Boolean indicating if the data scope should be for reading only (or for writing as well). Will be passed to `Client` constructor.
- **admin** (*boolean*) – Boolean indicating if the client will be used to interact with the Cluster Admin or Table Admin APIs. Will be passed to `Client` constructor.

Return type `Client`

Returns The client created with the retrieved JSON credentials.

classmethod from_service_account_p12(*client_email*, *private_key_path*, *project_id=None*, *read_only=False*, *admin=False*)
Factory to retrieve P12 credentials while creating client object.

Note: Unless you have an explicit reason to use a PKCS12 key for your service account, we recommend using a JSON key.

Parameters

- **client_email** (*string*) – The e-mail attached to the service account.
- **private_key_path** (*string*) – The path to a private key file (this file was given to you when you created the service account). This file must be in P12 format.
- **project_id** (*string*) – The ID of the project which owns the clusters, tables and data. Will be passed to `Client` constructor.
- **read_only** (*boolean*) – Boolean indicating if the data scope should be for reading only (or for writing as well). Will be passed to `Client` constructor.
- **admin** (*boolean*) – Boolean indicating if the client will be used to interact with the Cluster Admin or Table Admin APIs. Will be passed to `Client` constructor.

Return type `Client`

Returns The client created with the retrieved P12 credentials.

list_clusters(*timeout_seconds=None*)
Lists clusters owned by the project.

Parameters **timeout_seconds** (*integer*) – Number of seconds for request time-out. If not passed, defaults to value set on client.

Return type tuple

Returns A pair of results, the first is a list of `Cluster` s returned and the second is a list of strings (the failed zones in the request).

list_zones(*timeout_seconds=None*)
Lists zones associated with project.

Parameters **timeout_seconds** (*integer*) – Number of seconds for request time-out. If not passed, defaults to value set on client.

Return type list of strings

Returns The names of the zones

Raises `ValueError` if one of the zones is not in `OK` state.

project_id
Getter for client's project ID.

Return type string

Returns The project ID stored on the client.

project_name
Project name to be used with Cluster Admin API.

Note: This property will not change if `project_id` does not, but the return value is not cached.

The project name is of the form "projects/{project_id}".

Return type string

Returns The project name to be used with the Cloud Bigtable Admin API RPC service.

gcloud_bigtable.client.**DATA_SCOPE = 'https://www.googleapis.com/auth/cloud-bigtable.data'**
Scope for reading and writing table data.

gcloud_bigtable.client.**PROJECT_ENV_VAR = 'GCLOUD_PROJECT'**
Environment variable used to provide an implicit project ID.

gcloud_bigtable.client.**READ_ONLY_SCOPE = 'https://www.googleapis.com/auth/cloud-bigtable.data.readonly'**
Scope for reading table data.

8.1.4 Cluster

User friendly container for Google Cloud Bigtable Cluster.

class gcloud_bigtable.cluster.**Cluster**(*zone*, *cluster_id*, *client*, *display_name=None*, *serve_nodes=3*)
Bases: `object`

Representation of a Google Cloud Bigtable Cluster.

We can use a *`Cluster`* to:

- *reload()* itself
- *create()* itself
- Check if an *operation_finished()* (each of *create()*, *update()* and *undelete()* return with long-running operations)
- *update()* itself
- *delete()* itself
- *undelete()* itself

Note: For now, we leave out the properties `hdd_bytes` and `ssd_bytes` (both integers) and also the `default_storage_type` (an enum) which if not sent will end up as `data_pb2.STORAGE_SSD`.

Parameters

- **zone** (*string*) – The name of the zone where the cluster resides.
- **cluster_id** (*string*) – The ID of the cluster.
- **client** (*client.Client*) – The client that owns the cluster. Provides authorization and a project ID.
- **display_name** (*string*) – (Optional) The display name for the cluster in the Cloud Console UI. (Must be between 4 and 30 characters.) If this value is not set in the constructor, will fall back to the cluster ID.
- **serve_nodes** (*integer*) – (Optional) The number of nodes in the cluster. Defaults to 3.

client

Getter for cluster's client.

Return type *client.Client*

Returns The client stored on the cluster.

create(*timeout_seconds=None*)

Create this cluster.

Note: Uses the `project_id`, `zone` and `cluster_id` on the current *Cluster* in addition to the `display_name` and `serve_nodes`. If you'd like to change them before creating, reset the values via

```
cluster.display_name = 'New display name'
cluster.cluster_id = 'i-changed-my-mind'
```

before calling *create()*.

Parameters **timeout_seconds** (*integer*) – Number of seconds for request time-out. If not passed, defaults to value set on cluster.

credentials

Getter for cluster's credentials.

Return type `oauth2client.client.OAuth2Credentials`

Returns The credentials stored on the cluster's client.

delete(*timeout_seconds=None*)

Delete this cluster.

Parameters **timeout_seconds** (*integer*) – Number of seconds for request time-out. If not passed, defaults to value set on cluster.

classmethod from_pb(*cluster_pb*, *client*)
Creates a cluster instance from a protobuf.

Parameters

- **cluster_pb** (bigtable_cluster_data_pb2.Cluster) – A cluster protobuf object.
- **client** (*client.Client*) – The client that owns the cluster.

Return type *Cluster*

Returns The cluster parsed from the protobuf response.

Raises ValueError if the cluster name does not match _CLUSTER_NAME_RE or if the parsed project ID does not match the project ID on the client.

list_tables(*timeout_seconds=None*)
List the tables in this cluster.

Parameters **timeout_seconds** (*integer*) – Number of seconds for request time-out. If not passed, defaults to value set on cluster.

Return type list of Table

Returns The list of tables owned by the cluster.

Raises ValueError if one of the returned tables has a name that is not of the expected format.

name
Cluster name used in requests.

Note: This property will not change if zone and cluster_id do not, but the return value is not cached.

The cluster name is of the form "projects/{project_id}/zones/{zone}/clusters/{cluster_id}".

Return type string

Returns The cluster name.

operation_finished(*timeout_seconds=None*)
Check if the current operation has finished.

Parameters **timeout_seconds** (*integer*) – Number of seconds for request time-out. If not passed, defaults to value set on cluster.

Return type boolean

Returns A boolean indicating if the current operation has completed.

Raises ValueError if there is no current operation set.

project_id
Getter for cluster's project ID.

Return type string

Returns The project ID for the cluster (is stored on the client).

reload(*timeout_seconds=None*)
Reload the metadata for this cluster.

Parameters **timeout_seconds** (*integer*) – Number of seconds for request time-out. If not passed, defaults to value set on cluster.

table(*table_id*)
Factory to create a table associated with this cluster.

Parameters **table_id** (*string*) – The ID of the table.

Return type Table

Returns The table owned by this cluster.

timeout_seconds
Getter for cluster's default timeout seconds.

Return type integer

Returns The timeout seconds default stored on the cluster's client.

undelete(*timeout_seconds=None*)
Undelete this cluster.

Parameters **timeout_seconds** (*integer*) – Number of seconds for request time-out. If not passed, defaults to value set on cluster.

update(*timeout_seconds=None*)
Update this cluster.

Note: Updates the display_name and serve_nodes. If you'd like to change them before updating, reset the values via

```
cluster.display_name = 'New display name'
cluster.serve_nodes = 3
```

before calling *update()*.

Parameters **timeout_seconds** (*integer*) – Number of seconds for request time-out. If not passed, defaults to value set on cluster.

8.1.5 Table

User friendly container for Google Cloud Bigtable Table.

class gcloud_bigtable.table.**Table**(*table_id*, *cluster*)
Bases: object

Representation of a Google Cloud Bigtable Table.

Note: We don't define any properties on a table other than the name. As the proto says, in a request:

> The name field of the Table and all of its ColumnFamilies must be left blank, and will be populated in the response.

This leaves only the current_operation and granularity fields. The current_operation is only used for responses while granularity is an enum with only one value.

We can use a *Table* to:

- *create()* the table
- *rename()* the table

- *delete()* the table
- *list_column_families()* in the table

Parameters

- **table_id** (*string*) – The ID of the table.
- **cluster** (*cluster.Cluster*) – The cluster that owns the table.

client
Getter for table's client.

Return type *client.Client*

Returns The client that owns this table.

cluster
Getter for table's cluster.

Return type *cluster.Cluster*

Returns The cluster stored on the table.

column_family(*column_family_id*, *gc_rule=None*)
Factory to create a column family associated with this table.

Parameters

- **column_family_id** (*string*) – The ID of the column family.
- **gc_rule** (*column_family.GarbageCollectionRule*, *column_family.GarbageCollectionRuleUnion* or *column_family.GarbageCollectionRuleIntersection*) – (Optional) The garbage collection settings for this column family.

Return type *column_family.ColumnFamily*

Returns A column family owned by this table.

create(*initial_split_keys=None*, *timeout_seconds=None*)
Creates this table.

Note: Though a `_generated.bigtable_table_data_pb2.Table` is also allowed (as the `table` property) in a create table request, we do not support it in this method. As mentioned in the *Table* docstring, the name is the only useful property in the table proto.

Note: A create request returns a `_generated.bigtable_table_data_pb2.Table` but we don't use this response. The proto definition allows for the inclusion of a `current_operation` in the response, but in example usage so far, it seems the Bigtable API does not return any operation.

Parameters

- **initial_split_keys** (*iterable of strings*) – (Optional) List of row keys that will be used to initially split the table into several tablets (Tablets are similar to HBase regions). Given two split keys, `"s1"` and `"s2"`, three tablets will be created, spanning the key ranges: `[, s1)`, `[s1, s2)`, `[s2, )`.
- **timeout_seconds** (*integer*) – Number of seconds for request time-out. If not passed, defaults to value set on table.

delete(*timeout_seconds=None*)
Delete this table.

Parameters **timeout_seconds** (*integer*) – Number of seconds for request time-out. If not passed, defaults to value set on table.

list_column_families(*timeout_seconds=None*)
Check if this table exists.

Parameters **timeout_seconds** (*integer*) – Number of seconds for request time-out. If not passed, defaults to value set on table.

Return type dictionary with string as keys and *column_family.ColumnFamily* as values

Returns List of column families attached to this table.

Raises `ValueError` if the column family name from the response does not agree with the computed name from the column family ID.

name
Table name used in requests.

Note: This property will not change if `table_id` does not, but the return value is not cached.

The table name is of the form

```
"projects/../zones/../clusters/../tables/{table_id}"
```

Return type string

Returns The table name.

rename(*new_table_id*, *timeout_seconds=None*)
Rename this table.

Note: This cannot be used to move tables between clusters, zones, or projects.

Note: The Bigtable Table Admin API currently returns

```
BigtableTableService.RenameTable is not yet implemented
```

when this method is used. It's unclear when this method will actually be supported by the API.

Parameters

- **new_table_id** (*string*) – The new name table ID.
- **timeout_seconds** (*integer*) – Number of seconds for request time-out. If not passed, defaults to value set on table.

timeout_seconds
Getter for table's default timeout seconds.

Return type integer

Returns The timeout seconds default stored on the table's client.

8.1.6 Table Column Families

When creating a *Table*, it is possible to set garbage collection rules for expired data.

By setting a rule, cells in the table matching the rule will be deleted during periodic garbage collection (which executes opportunistically in the background).

The types *GarbageCollectionRule*, *GarbageCollectionRuleUnion* and *GarbageCollectionRuleIntersection* can all be used as the optional `gc_rule` argument in the *ColumnFamily* constructor. This value is then used in the *create* and *update* methods.

These rules can be nested arbitrarily, with *GarbageCollectionRule* at the lowest level of the nesting:

```
import datetime

max_age = datetime.timedelta(days=3)
rule1 = GarbageCollectionRule(max_age=max_age)
rule2 = GarbageCollectionRule(max_num_versions=1)

# Make a composite that matches anything older than 3 days **AND**
# with more than 1 version.
rule3 = GarbageCollectionIntersection(rules=[rule1, rule2])

# Make another composite that matches our previous intersection
# **OR** anything that has more than 3 versions.
rule4 = GarbageCollectionRule(max_num_versions=3)
rule5 = GarbageCollectionUnion(rules=[rule3, rule4])
```

Column Family Module

User friendly container for Google Cloud Bigtable Column Family.

class `gcloud_bigtable.column_family.`**`ColumnFamily`**(*column_family_id*, *table*, *gc_rule=None*)

Bases: `object`

Representation of a Google Cloud Bigtable Column Family.

We can use a *ColumnFamily* to:

- *create()* itself
- *update()* itself
- *delete()* itself

Parameters

- **`column_family_id`** (*string*) – The ID of the column family.
- **`table`** (*table.Table*) – The table that owns the column family.
- **`gc_rule`** (*GarbageCollectionRule*, *GarbageCollectionRuleUnion* or *GarbageCollectionRuleIntersection*) – (Optional) The garbage collection settings for this column family.

`client`

Getter for column family's client.

Return type *client.Client*

Returns The client that owns this column family.

create(*timeout_seconds=None*)
Create this column family.

Parameters **timeout_seconds** (*integer*) – Number of seconds for request time-out. If not passed, defaults to value set on column family.

delete(*timeout_seconds=None*)
Delete this column family.

Parameters **timeout_seconds** (*integer*) – Number of seconds for request time-out. If not passed, defaults to value set on column family.

name
Column family name used in requests.

Note: This property will not change if `column_family_id` does not, but the return value is not cached.

The table name is of the form `"projects/../zones/../clusters/../tables/../columnFamilies/.."`

Return type string

Returns The column family name.

table
Getter for column family's table.

Return type *table.Table*

Returns The table stored on the column family.

timeout_seconds
Getter for column family's default timeout seconds.

Return type integer

Returns The timeout seconds default.

update(*timeout_seconds=None*)
Update this column family.

Note: The Bigtable Table Admin API currently returns

```
BigtableTableService.UpdateColumnFamily is not yet implemented
```

when this method is used. It's unclear when this method will actually be supported by the API.

Parameters **timeout_seconds** (*integer*) – Number of seconds for request time-out. If not passed, defaults to value set on column family.

class gcloud_bigtable.column_family.**GarbageCollectionRule**(*max_num_versions=None*, *max_age=None*)

Bases: object

Table garbage collection rule.

Cells in the table fitting the rule will be deleted during garbage collection.

These values can be combined via *GarbageCollectionRuleUnion* and *GarbageCollectionRuleIntersection*.

Note: At most one of `max_num_versions` and `max_age` can be specified at once.

Note: A string `gc_expression` can also be used with API requests, but that value would be superceded by a `gc_rule`. As a result, we don't support that feature and instead support via this native object.

Parameters

- **max_num_versions** (*integer*) – The maximum number of versions
- **max_age** (`datetime.timedelta`) – The maximum age allowed for a cell in the table.

Raises `ValueError` if both `max_num_versions` and `max_age` are set.

to_pb()
Converts the *GarbageCollectionRule* to a protobuf.

Return type `data_pb2.GcRule`

Returns The converted current object.

class gcloud_bigtable.column_family.**GarbageCollectionRuleIntersection**(*rules=None*)
Bases: `object`

Intersection of garbage collection rules.

Parameters **rules** (*list*) – List of garbage collection rules, unions and/or intersections.

to_pb()
Converts the intersection into a single gc rule as a protobuf.

Return type `data_pb2.GcRule`

Returns The converted current object.

class gcloud_bigtable.column_family.**GarbageCollectionRuleUnion**(*rules=None*)
Bases: `object`

Union of garbage collection rules.

Parameters **rules** (*list*) – List of garbage collection rules, unions and/or intersections.

to_pb()
Converts the union into a single gc rule as a protobuf.

Return type `data_pb2.GcRule`

Returns The converted current object.

8.2 Indices and tables

- genindex
- modindex

g

gcloud_bigtable.client, 19
gcloud_bigtable.cluster, 21
gcloud_bigtable.column_family, 27
gcloud_bigtable.constants, 17
gcloud_bigtable.data_connection, 17
gcloud_bigtable.table, 24

A

ADMIN_SCOPE (in module gcloud_bigtable.client), 19

C

check_and_mutate_row() (gcloud_bigtable.data_connection.DataConnection method), 18
Client (class in gcloud_bigtable.client), 19
client (gcloud_bigtable.cluster.Cluster attribute), 22
client (gcloud_bigtable.column_family.ColumnFamily attribute), 27
client (gcloud_bigtable.table.Table attribute), 25
Cluster (class in gcloud_bigtable.cluster), 21
cluster (gcloud_bigtable.table.Table attribute), 25
cluster() (gcloud_bigtable.client.Client method), 19
CLUSTER_ADMIN_HOST (in module gcloud_bigtable.constants), 17
CLUSTER_ADMIN_PORT (in module gcloud_bigtable.constants), 17
column_family() (gcloud_bigtable.table.Table method), 25
ColumnFamily (class in gcloud_bigtable.column_family), 27
create() (gcloud_bigtable.cluster.Cluster method), 22
create() (gcloud_bigtable.column_family.ColumnFamily method), 28
create() (gcloud_bigtable.table.Table method), 25
credentials (gcloud_bigtable.client.Client attribute), 20
credentials (gcloud_bigtable.cluster.Cluster attribute), 22

D

DATA_API_HOST (in module gcloud_bigtable.data_connection), 17
DATA_SCOPE (in module gcloud_bigtable.client), 21
DataConnection (class in gcloud_bigtable.data_connection), 17
delete() (gcloud_bigtable.cluster.Cluster method), 22
delete() (gcloud_bigtable.column_family.ColumnFamily method), 28
delete() (gcloud_bigtable.table.Table method), 25

F

from_pb() (gcloud_bigtable.cluster.Cluster class method), 23
from_service_account_json() (gcloud_bigtable.client.Client class method), 20
from_service_account_p12() (gcloud_bigtable.client.Client class method), 20

G

GarbageCollectionRule (class in gcloud_bigtable.column_family), 28
GarbageCollectionRuleIntersection (class in gcloud_bigtable.column_family), 29
GarbageCollectionRuleUnion (class in gcloud_bigtable.column_family), 29
gcloud_bigtable.client (module), 19
gcloud_bigtable.cluster (module), 21
gcloud_bigtable.column_family (module), 27
gcloud_bigtable.constants (module), 17
gcloud_bigtable.data_connection (module), 17
gcloud_bigtable.table (module), 24

L

list_clusters() (gcloud_bigtable.client.Client method), 21
list_column_families() (gcloud_bigtable.table.Table method), 26
list_tables() (gcloud_bigtable.cluster.Cluster method), 23
list_zones() (gcloud_bigtable.client.Client method), 21

M

mutate_row() (gcloud_bigtable.data_connection.DataConnection method), 18

N

name (gcloud_bigtable.cluster.Cluster attribute), 23
name (gcloud_bigtable.column_family.ColumnFamily attribute), 28
name (gcloud_bigtable.table.Table attribute), 26

O

operation_finished() (gcloud_bigtable.cluster.Cluster method), 23

P

PROJECT_ENV_VAR (in module gcloud_bigtable.client), 21
project_id (gcloud_bigtable.client.Client attribute), 21
project_id (gcloud_bigtable.cluster.Cluster attribute), 23
project_name (gcloud_bigtable.client.Client attribute), 21

R

read_modify_write_row() (gcloud_bigtable.data_connection.DataConnection method), 18
READ_ONLY_SCOPE (gcloud_bigtable.data_connection.DataConnection attribute), 17
READ_ONLY_SCOPE (in module gcloud_bigtable.client), 21
read_rows() (gcloud_bigtable.data_connection.DataConnection method), 18
reload() (gcloud_bigtable.cluster.Cluster method), 23
rename() (gcloud_bigtable.table.Table method), 26

S

sample_row_keys() (gcloud_bigtable.data_connection.DataConnection method), 18
SCOPE (gcloud_bigtable.data_connection.DataConnection attribute), 17

T

Table (class in gcloud_bigtable.table), 24
table (gcloud_bigtable.column_family.ColumnFamily attribute), 28
table() (gcloud_bigtable.cluster.Cluster method), 24
TABLE_ADMIN_HOST (in module gcloud_bigtable.constants), 17
TABLE_ADMIN_PORT (in module gcloud_bigtable.constants), 17
timeout_seconds (gcloud_bigtable.cluster.Cluster attribute), 24
timeout_seconds (gcloud_bigtable.column_family.ColumnFamily attribute), 28
timeout_seconds (gcloud_bigtable.table.Table attribute), 26
to_pb() (gcloud_bigtable.column_family.GarbageCollectionRule method), 29
to_pb() (gcloud_bigtable.column_family.GarbageCollectionRuleIntersection method), 29
to_pb() (gcloud_bigtable.column_family.GarbageCollectionRuleUnion method), 29

U

undelete() (gcloud_bigtable.cluster.Cluster method), 24
update() (gcloud_bigtable.cluster.Cluster method), 24
update() (gcloud_bigtable.column_family.ColumnFamily method), 28