cognite-replicator Documentation

Release 0.7.6

Cognite

Mar 27, 2020

Contents

cognite.replicator.**configure_databricks_logger**(*log_level=20*, *logger: logging.Logger = None*, *file_path: str = None*) → logging.Logger

Configure logging for databricks.

Parameters

- **log_level** – the logging level
- **logger** – the logger to use, default is root logger
- **file_path** – the path to a file for storing logs to persistent disk if provided

cognite.replicator.**configure_logger**(*log_level: str = 'INFO'*, *log_path: pathlib.Path = None*) → None

Configure the logging to stdout and optionally local file and GCP stackdriver.

cognite.replicator.assets.**build_asset_create**(*src_asset: cognite.client.data_classes.assets.Asset*, *src_id_dst_map: Dict[int, int]*, *project_src: str*, *runtime: int*, *depth: int*) → cognite.client.data_classes.assets.Asset

Makes a new copy of the asset to be replicated based on the source asset.

Parameters

- **src_asset** – The asset from the source to be replicated to destination.
- **src_id_dst_map** – A dictionary of all the mappings of source asset id to destination asset id.
- **project_src** – The name of the project the object is being replicated from.
- **runtime** – The timestamp to be used in the new replicated metadata.
- **depth** – The depth of the asset within the asset hierarchy.

Returns The replicated asset to be created in the destination.

cognite.replicator.assets.**build_asset_update**(*src_asset: cognite.client.data_classes.assets.Asset*, *dst_asset: cognite.client.data_classes.assets.Asset*, *src_id_dst_map: Dict[int, int]*, *project_src: str*, *runtime: int*, *depth: int*) → cognite.client.data_classes.assets.Asset

Makes an updated version of the destination asset based on the corresponding source asset.

Parameters

- **src_asset** – The asset from the source to be replicated to destination.
- **dst_asset** – The asset from the destination that needs to be updated to reflect changes made to its source asset.
- **src_id_dst_map** – **A dictionary of all the mappings of source asset id to destination asset id.
- **project_src** – The name of the project the object is being replicated from.
- **runtime** – The timestamp to be used in the new replicated metadata.
- **depth** – **The depth of the asset within the asset hierarchy.

** only needed when hierarchy becomes mutable

Returns The updated asset object for the replication destination.

cognite.replicator.assets.**create_hierarchy**(*src_assets: List[cognite.client.data_classes.assets.Asset], dst_assets: List[cognite.client.data_classes.assets.Asset], project_src: str, runtime: int, client: cognite.client._cognite_client.CogniteClient, subtree_ids: Optional[List[int]] = None, subtree_external_ids: Optional[List[str]] = None, subtree_max_depth: Optional[int] = None*)

Creates/updates the asset hierarchy in batches by depth, starting with the root assets and then moving on to the children of those roots, etc.

Parameters

- **src_assets** – A list of the assets that are in the source.
- **dst_assets** – A list of the assets that are in the destination.
- **project_src** – The name of the project the object is being replicated from.
- **runtime** – The timestamp to be used in the new replicated metadata.
- **client** – The client corresponding to the destination project.
- **subtree_ids** – The id of the subtree root to replicate,
- **subtree_external_ids** – The external id of the subtree root to replicate,
- **subtree_max_depth** – The maximum tree depth to replicate,

cognite.replicator.assets.**find_children**(*assets: List[cognite.client.data_classes.assets.Asset], parents: Union[List[None], List[cognite.client.data_classes.assets.Asset]]*) → List[cognite.client.data_classes.assets.Asset]

Creates a list of all the assets that are children of the parent assets.

Parameters

- **assets** – A list of all the assets to search for children from.
- **parents** – A list of all the assets to find children for.

Returns A list of all the assets that are children to the parents.

cognite.replicator.assets.**replicate**(*client_src: cognite.client._cognite_client.CogniteClient, client_dst: cognite.client._cognite_client.CogniteClient, delete_replicated_if_not_in_src: bool = False, delete_not_replicated_in_dst: bool = False, subtree_ids: Union[int, List[int], None] = None, subtree_external_ids: Union[str, List[str], None] = None, subtree_max_depth: Optional[int] = None*)

Replicates all the assets from the source project into the destination project.

Parameters

- **client_src** – The client corresponding to the source project.
- **client_dst** – The client corresponding to the destination project.
- **delete_replicated_if_not_in_src** – If True, will delete replicated assets that are in the destination,
- **no longer in the source project** (*but*) –

- **delete_not_replicated_in_dst** – If True, will delete assets from the destination if they were not replicated
- **the source** (*from*) –
- **subtree_ids** – The id or list of ids of subtree root to replicate,
- **subtree_external_ids** – The external id or list of external ids of the subtree root to replicate,
- **subtree_max_depth** – The maximum tree depth to replicate,

cognite.replicator.assets.**unlink_subtree_parents**(*src_assets: List[cognite.client.data_classes.assets.Asset], subtree_ids: Optional[List[int]] = None, subtree_external_ids: Optional[List[str]] = None*)

Sets the parentId of subtree roots to be None

Parameters

- **src_assets** – A list of the assets that are in the source.
- **subtree_ids** – The id(s) of the subtree root(s).
- **subtree_external_ids** – The external id(s) of the subtree root(s).

cognite.replicator.datapoints.**batch_replicate**(*client_src: cognite.client._cognite_client.CogniteClient, client_dst: cognite.client._cognite_client.CogniteClient, job_id: int, ext_ids: List[str], limit: int, mock_run: bool = False, partition_size: int = 100000, src_datapoint_transform: Optional[Callable[[cognite.client.data_classes.datapoints.Datapoint, cognite.client.data_classes.datapoints.Datapoint]] = None, timerange_transform: Optional[Callable[[Tuple[int, int]], Tuple[int, int]]] = None, start: Union[int, str] = None, end: Union[int, str] = None*)

Replicates datapoints for each time series specified by the external id list.

Parameters

- **client_src** – The client corresponding to the source project.
- **client_dst** – The client corresponding to the destination project.
- **job_id** – The batch number being processed
- **ext_ids** – The list of external ids for time series to copy over
- **limit** – The maximum number of data points to copy per time series
- **mock_run** – If true, only retrieves data points from source and does not insert into destination
- **partition_size** – The maximum number of datapoints to retrieve per request
- **src_datapoint_transform** – Function to apply to all source datapoints before inserting into destination

- **timerange_transform** – Function to set the time range boundaries (start, end) arbitrarily.
- **start** – Timestamp to start replication onwards from; if not specified starts at most recent datapoint
- **end** – If specified, limits replication to datapoints earlier than the end time

cognite.replicator.datapoints.**replicate**(*client_src: cognite.client._cognite_client.CogniteClient, client_dst: cognite.client._cognite_client.CogniteClient, batch_size: Optional[int] = None, num_threads: int = 10, limit: Optional[int] = None, external_ids: Optional[List[str]] = None, mock_run: bool = False, partition_size: int = 100000, src_datapoint_transform: Optional[Callable[[cognite.client.data_classes.datapoints.Datapoint], cognite.client.data_classes.datapoints.Datapoint]] = None, timerange_transform: Optional[Callable[[Tuple[int, int]], Tuple[int, int]]] = None, start: Union[int, str] = None, end: Union[int, str] = None, exclude_pattern: str = None*)

Replicates data points from the source project into the destination project for all time series that exist in both environments.

Parameters

- **client_src** – The client corresponding to the source project.
- **client_dst** – The client corresponding to the destination project.
- **batch_size** – The size of batches to split the external id list into. Defaults to num_threads.
- **num_threads** – The number of threads to be used.
- **limit** – The maximum number of data points to copy per time series
- **external_ids** – A list of time series to replicate data points for
- **mock_run** – If true, runs the replication without insert, printing what would happen
- **partition_size** – The maximum number of datapoints to retrieve per request
- **src_datapoint_transform** – Function to apply to all source datapoints before inserting into destination
- **timerange_transform** – Function to set the time range boundaries (start, end) arbitrarily.
- **start** – Timestamp to start replication onwards from; if not specified starts at most recent datapoint
- **end** – If specified, limits replication to datapoints earlier than the end time
- **exclude_pattern** – Regex pattern; time series whose names match will not be replicated from

cognite.replicator.datapoints.**replicate_datapoints**(*client_src: cognite.client._cognite_client.CogniteClient, client_dst: cognite.client._cognite_client.CogniteClient, ts_external_id: str, limit: Optional[int] = None, partition_size: int = 100000, mock_run: bool = False, job_id: int = 1, src_datapoint_transform: Optional[Callable[[cognite.client.data_classes.datapoints.Da cognite.client.data_classes.datapoints.Datapoint]] = None, timerange_transform: Optional[Callable[[Tuple[int, int]], Tuple[int, int]]] = None, start: Union[int, str] = None, end: Union[int, str] = None*) → Tuple[bool, int]

Copies data points from the source tenant into the destination project, for the given time series.

If data points already exist in the destination for the time series, only the newer data points in the source are copied over.

Parameters

- **client_src** – The client corresponding to the source project.
- **client_dst** – The client corresponding to the destination project.
- **ts_external_id** – The external id of the time series to replicate datapoints for
- **limit** – The maximum number of data points to copy
- **partition_size** – The maximum number of datapoints to retrieve per request
- **mock_run** – If true, only retrieves data points from source and does not insert into destination
- **job_id** – The batch number being processed
- **src_datapoint_transform** – Function to apply to all source datapoints before inserting into destination
- **timerange_transform** – Function to set the time range boundaries (start, end) arbitrarily.
- **start** – Timestamp to start replication onwards from; if not specified starts at most recent datapoint
- **end** – If specified, limits replication to datapoints earlier than the end time

Returns A tuple of the success status (True if no failures) and the number of datapoints successfully replicated

cognite.replicator.events.**copy_events**(*src_events: List[cognite.client.data_classes.events.Event], src_id_dst_event: Dict[int, cognite.client.data_classes.events.Event], src_dst_ids_assets: Dict[int, int], project_src: str, runtime: int, client: cognite.client._cognite_client.CogniteClient*)

Creates/updates event objects and then attempts to create and update these objects in the destination.

Parameters

- **src_events** – A list of the events that are in the source.
- **src_id_dst_event** – A dictionary of an events source id to it's matching destination object.
- **src_dst_ids_assets** – A dictionary of all the mappings of source asset id to destination asset id.
- **project_src** – The name of the project the object is being replicated from.
- **runtime** – The timestamp to be used in the new replicated metadata.
- **client** – The client corresponding to the destination project.

cognite.replicator.events.**create_event**(*src_event: cognite.client.data_classes.events.Event, src_dst_ids_assets: Dict[int, int], project_src: str, runtime: int*) → cognite.client.data_classes.events.Event

Makes a new copy of the event to be replicated based on a source event.

Parameters

- **src_event** – The event from the source to be replicated to destination.
- **src_dst_ids_assets** – A dictionary of all the mappings of source asset id to destination asset id.
- **project_src** – The name of the project the object is being replicated from.
- **runtime** – The timestamp to be used in the new replicated metadata.

Returns The replicated event to be created in the destination.

cognite.replicator.events.**replicate**(*client_src: cognite.client._cognite_client.CogniteClient, client_dst: cognite.client._cognite_client.CogniteClient, batch_size: int = 10000, num_threads: int = 1, delete_replicated_if_not_in_src: bool = False, delete_not_replicated_in_dst: bool = False, skip_unlinkable: bool = False, skip_nonasset: bool = False*)

Replicates all the events from the source project into the destination project.

Parameters

- **client_src** – The client corresponding to the source project.
- **client_dst** – The client corresponding to the destination project.
- **batch_size** – The biggest batch size to post chunks in.
- **num_threads** – The number of threads to be used.
- **delete_replicated_if_not_in_src** – If True, will delete replicated events that are in the destination,
- **no longer in the source project** (*but*) –
- **delete_not_replicated_in_dst** – If True, will delete events from the destination if they were not replicated
- **the source** (*from*) –
- **skip_unlinkable** – If no assets exist in the destination for an event, do not replicate it
- **skip_nonasset** – If an event has no associated assets, do not replicate it

cognite.replicator.events.**update_event**(*src_event: cognite.client.data_classes.events.Event, dst_event: cognite.client.data_classes.events.Event, src_dst_ids_assets: Dict[int, int], project_src: str, runtime: int*) → cognite.client.data_classes.events.Event

Makes an updated version of the destination event based on the corresponding source event.

Parameters

- **src_event** – The event from the source to be replicated.
- **dst_event** – The event from the destination that needs to be updated to reflect changes made to its source event.
- **src_dst_ids_assets** – A dictionary of all the mappings of source asset id to destination asset id.
- **project_src** – The name of the project the object is being replicated from.
- **runtime** – The timestamp to be used in the new replicated metadata.

Returns The updated event object for the replication destination.

cognite.replicator.raw.**copy_rows**(*client_src: cognite.client._cognite_client.CogniteClient, client_dst: cognite.client._cognite_client.CogniteClient, db_tb_dict: Dict[str, List[str]], chunk_size: int*)

Goes through the databases and their tables in order to fetch the rows from source and post them to destination in chunks.

Parameters

- **client_src** – The client corresponding to the source project.
- **client_dst** – The client corresponding to the destination project.
- **db_tb_dict** – A dictionary over the databases and tables within the databases.
- **chunk_size** – The size of the chunks to fetch and post rows in.

cognite.replicator.raw.**create_databases_tables**(*client_src: cognite.client._cognite_client.CogniteClient, client_dst: cognite.client._cognite_client.CogniteClient*) → Dict[str, List[str]]

Creates the databases and tables in the destination and makes a dictionary with the database name as the key and a list of the table names as the value.

Parameters

- **client_src** – The client corresponding to the source project.
- **client_dst** – The client corresponding to the destination project.

Returns A dictionary mapping database names to a list of it's table names.

cognite.replicator.raw.**get_not_created_names**(*src_objects: List[Union[cognite.client.data_classes.raw.Database, cognite.client.data_classes.raw.Table]], dst_objects: List[Union[cognite.client.data_classes.raw.Database, cognite.client.data_classes.raw.Table]]*) → Tuple[List[str], List[str]]

Creates a list of all the source object names and a list of source object names that do not exist in destination.

:param src_objects: A list of all the source objects, either databases or tables. :param dst_objects: A list of all the destination objects, either databases or tables.

Returns A list of all the source object (database or table) names. not_created: A list of all the source object (database or table) names that do not exist in destination.

Return type src_names

cognite.replicator.raw.**insert_rows**(*rows: List[cognite.client.data_classes.raw.Row], db_name: str, tb_name: str, client: cognite.client._cognite_client.CogniteClient*)

Inserts the input rows from database and table from the source to the destination.

Parameters

- **rows** – A list of all the rows that need to be copied over.
- **db_name** – The name of the database that rows are copied from.
- **tb_name** – The name of the table that rows are copied from.
- **client** – The client corresponding to the destination project.

Returns The output of insert.

cognite.replicator.raw.**replicate**(*client_src: cognite.client._cognite_client.CogniteClient, client_dst: cognite.client._cognite_client.CogniteClient, chunk_size: int*)

Replicates all the raw from the source project into the destination project.

Parameters

- **client_src** – The client corresponding to the source project.
- **client_dst** – The client corresponding to the destination project.
- **chunk_size** – The biggest chunk size to fetch and post in.

cognite.replicator.replication.**clear_replication_metadata**(*client: cognite.client._cognite_client.CogniteClient*)

Removes the replication metadata from all resources, so that the replicated tenant will look like an original. Sample use-case: clean up a demo-tenant so that extra data is not present.

Caution: After clearing replication metadata, the delete_replicated_if_not_in_src option will delete everything.

Parameters **client** – The client to strip replication metadata away from

cognite.replicator.replication.**existing_mapping**(**objects, ids: Dict[int, int] = None*) → Dict[int, int]

Updates a dictionary with all the source id to destination id pairs for the objects that have been replicated.

Parameters

- ***objects** – A list of objects to make a mapping of.
- **ids** – A dictionary of all the mappings of source object id to destination object id.

Returns The updated dictionary with the ids from new objects that have been replicated.

cognite.replicator.replication.**filter_objects**(*objects: Union[List[cognite.client.data_classes.events.Event], List[cognite.client.data_classes.files.FileMetadata], List[cognite.client.data_classes.time_series.TimeSeries]], src_dst_ids_assets: Dict[int, int], skip_unlinkable: bool = False, skip_nonasset: bool = False, filter_fn: Optional[Callable[[Union[cognite.client.data_classes.events.Event, cognite.client.data_classes.files.FileMetadata, cognite.client.data_classes.time_series.TimeSeries]], bool]] = None*) → Union[List[cognite.client.data_classes.events.Event], List[cognite.client.data_classes.files.FileMetadata], List[cognite.client.data_classes.time_series.TimeSeries]]

Filters out objects based on their assets and optionally custom filter logic

Parameters

- **objects** – A list of all the objects to filter.
- **src_dst_ids_assets** – A dictionary of all the mappings of source asset id to destination asset id.
- **skip_unlinkable** – If True, excludes objects whose assets haven't been replicated in the destination
- **skip_nonasset** – If True, excludes objects without any associated assets
- **filter_fn** – If specified, is used to filter the objects in addition to the existing asset filters.

Returns A list of objects that meet the criteria.

cognite.replicator.replication.**find_objects_to_delete_if_not_in_src**(*src_objects: List[Union[cognite.client.data_cl cog- nite.client.data_classes.events.Ev cog- nite.client.data_classes.files.FileM cog- nite.client.data_classes.time_seri dst_objects: List[Union[cognite.client.data_cl cog- nite.client.data_classes.events.Ev cog- nite.client.data_classes.files.FileM cog- nite.client.data_classes.time_seri*) → List[Union[cognite.client.data_cl cog- nite.client.data_classes.events.Ev cog- nite.client.data_classes.files.FileM cog- nite.client.data_classes.time_serie

Compare the destination and source assets and delete the ones that are no longer in the source.

Parameters

- **src_objects** – The list of objects from the src destination.
- **dst_objects** – The list of objects from the dst destination.

cognite.replicator.replication.**find_objects_to_delete_not_replicated_in_dst**(*dst_objects: List[Union[cognite.client.data_classes.assets.Asset, cognite.client.data_classes.events.Event, cognite.client.data_classes.files.FileMetadata, cognite.client.data_classes.time_series.TimeSeries]]*) → List[cognite.client.dat

Deleting all the assets in the destination that do not have the "_replicatedSource" in metadata, which means that is was not copied from the source, but created in the destination.

Parameters **dst_objects** – The list of objects from the dst destination.

cognite.replicator.replication.**get_asset_ids**(*ids: List[int], src_dst_ids_assets: Dict[int, int]*) → Optional[List[int]]

Create the list of destination asset ids from the list of source asset ids.

Parameters

- **ids** – A list of the source asset ids from the source object.
- **src_dst_ids_assets** – A dictionary of all the mappings of source asset id to destination asset id.

Returns A list of the destination asset ids for the destination object.

cognite.replicator.replication.**make_id_object_map**(*objects: List[Union[cognite.client.data_classes.assets.Asset, cognite.client.data_classes.events.Event, cognite.client.data_classes.files.FileMetadata, cognite.client.data_classes.time_series.TimeSeries]]*) → Dict[int, Union[cognite.client.data_classes.assets.Asset, cognite.client.data_classes.events.Event, cognite.client.data_classes.files.FileMetadata, cognite.client.data_classes.time_series.TimeSeries]]

Makes a dictionary with the source object id as the key and the object as the value for objects that have been replicated.

Parameters **objects** – A list of objects that are from the replication destination.

Returns A dictionary of source object id to destination object for objects that have been replicated.

cognite.replicator.replication.**make_objects_batch**(*src_objects: List[Union[cognite.client.data_classes.assets.Asset, cognite.client.data_classes.events.Event, cognite.client.data_classes.files.FileMetadata, cognite.client.data_classes.time_series.TimeSeries]], src_id_dst_map: Dict[int, Union[cognite.client.data_classes.assets.Asset, cognite.client.data_classes.events.Event, cognite.client.data_classes.files.FileMetadata, cognite.client.data_classes.time_series.TimeSeries]], src_dst_ids_assets: Dict[int, int], create, update, project_src: str, replicated_runtime: int, depth: Optional[int] = None, dst_ts: Optional[cognite.client.data_classes.time_series.TimeSeries] = None*) → Tuple[List[Union[cognite.client.data_classes.assets.Asset, cognite.client.data_classes.events.Event, cognite.client.data_classes.files.FileMetadata, cognite.client.data_classes.time_series.TimeSeries]], List[Union[cognite.client.data_classes.assets.Asset, cognite.client.data_classes.events.Event, cognite.client.data_classes.files.FileMetadata, cognite.client.data_classes.time_series.TimeSeries]], List[Union[cognite.client.data_classes.assets.Asset, cognite.client.data_classes.events.Event, cognite.client.data_classes.files.FileMetadata, cognite.client.data_classes.time_series.TimeSeries]]]

Create a batch of new objects from a list of source objects or update existing destination objects to their corresponding source object.

Parameters

- **src_objects** – A list of objects to be replicated from a source.
- **src_id_dst_map** – A dictionary of source object ids to the matching destination object.
- **src_dst_ids_assets** – A dictionary of all the mappings of source asset id to destination asset id.
- **create** – The function to be used in order to create all the objects in CDF.

- **update** – The function to be used in order to update the existing objects in CDF.
- **project_src** – The name of the project the object is being replicated from.
- **replicated_runtime** – The timestamp to be used in the new replicated metadata.
- **depth** – The depth of the asset within the asset hierarchy, only used for making assets.
- **dst_ts** – List of timeseries in the destination - Will be used for comparison if current timeseries where not copied by the replicator

Returns A list of all the new objects to be posted to CDF. update_objects: A list of all the updated objects to be updated in CDF. unchanged_objects: A list of all the objects that don't need to be updated.

Return type create_objects

cognite.replicator.replication.**new_metadata**(*obj: Union[cognite.client.data_classes.assets.Asset, cognite.client.data_classes.events.Event, cognite.client.data_classes.files.FileMetadata, cognite.client.data_classes.time_series.TimeSeries], project_src: str, replicated_runtime: int*) → Dict[str, Union[int, str]]

Copies the objects metadata and adds three new fields to it providing information about the objects replication.

Fields Created

- **_replicatedSource**: The name of the project this object is replicated from.
- **_replicatedTime**: The timestamp of when the object was replicated, all objects created/updated in the same execution will have the same timestamp.
- **_replicatedInternalId**: The internal id of the source object that the destination object is being replicated from.

Parameters

- **obj** – The source object that is being replicated to the destination.
- **project_src** – The name of the project the object is being replicated from.
- **replicated_runtime** – The timestamp to be used in the new replicated metadata.

Returns The metadata dictionary for the replicated destination object based on the source object.

cognite.replicator.replication.**remove_replication_metadata**(*objects: Union[List[cognite.client.data_classes.assets.A List[cognite.client.data_classes.events.Event], List[cognite.client.data_classes.time_series.Tim*

Removes the replication metadata from the passed resource list, so that the resources will look original. See also clear_replication_metadata.

Parameters **objects** – The list of objects to strip replication metadata away from

cognite.replicator.replication.**retry**(*function, objects: List[Union[cognite.client.data_classes.assets.Asset, cognite.client.data_classes.events.Event, cognite.client.data_classes.files.FileMetadata, cognite.client.data_classes.time_series.TimeSeries, cognite.client.data_classes.raw.Row]], **kwargs*) → List[Union[cognite.client.data_classes.assets.Asset, cognite.client.data_classes.events.Event, cognite.client.data_classes.time_series.TimeSeries]]

Attempt to either create/update the objects, if it fails retry creating/updating the objects. This will retry up to

three times.

Parameters

- **function** – The function that will be applied to objects, either creating_objects or updating_objects.
- **objects** – A list of all the new objects or updated objects.

Returns A list of all the objects that were created or updated in CDF.

cognite.replicator.replication.**thread**(*num_threads: int, copy, src_objects: List[Union[cognite.client.data_classes.events.Event, cognite.client.data_classes.files.FileMetadata, cognite.client.data_classes.time_series.TimeSeries]], src_id_dst_obj: Dict[int, Union[cognite.client.data_classes.events.Event, cognite.client.data_classes.files.FileMetadata, cognite.client.data_classes.time_series.TimeSeries]], src_dst_ids_assets: Dict[int, int], project_src: str, replicated_runtime: int, client: cognite.client._cognite_client.CogniteClient, dst_ts: Optional[cognite.client.data_classes.time_series.TimeSeries] = None*)

Split up objects to replicate them in batches and thread each batch.

Parameters

- **num_threads** – The number of threads to be used.
- **copy** – The function used to copy objects.
- **src_objects** – A list of all the objects in the source to be replicated.
- **src_id_dst_obj** – A dictionary of source object id to destination object.
- **src_dst_ids_assets** – A dictionary of all the mappings of source asset id to destination asset id.
- **project_src** – The name of the project the object is being replicated from.
- **replicated_runtime** – The timestamp to be used in the new replicated metadata.
- **client** – The Cognite Client for the destination project.
- **dst_ts** – List of timeseries in the destination - Will be used for comparison if current timeseries where not copied by the replicator

cognite.replicator.time_series.**copy_ts**(*src_ts: List[cognite.client.data_classes.time_series.TimeSeries], src_id_dst_ts: Dict[int, cognite.client.data_classes.time_series.TimeSeries], src_dst_ids_assets: Dict[int, int], project_src: str, runtime: int, client: cognite.client._cognite_client.CogniteClient, dst_ts: List[cognite.client.data_classes.time_series.TimeSeries]*)

Creates/updates time series objects and then attempts to create and update these time series in the destination.

Parameters

- **src_ts** – A list of the time series that are in the source.
- **src_id_dst_ts** – A dictionary of a time series source id to it's matching destination object.

- **src_dst_ids_assets** – A dictionary of all the mappings of source asset id to destination asset id.
- **project_src** – The name of the project the object is being replicated from.
- **runtime** – The timestamp to be used in the new replicated metadata.
- **client** – The client corresponding to the destination project.
- **dst_ts** – List of timeseries in the destination - Will be used for comparison if current timeseries where not copied by the replicator

cognite.replicator.time_series.**create_time_series**(*src_ts: cognite.client.data_classes.time_series.TimeSeries, src_dst_ids_assets: Dict[int, int], project_src: str, runtime: int*) → cognite.client.data_classes.time_series.TimeSeries

Make a new copy of the time series to be replicated based on a source time series.

Parameters

- **src_ts** – The time series from the source to be replicated to the destination.
- **src_dst_ids_assets** – A dictionary of all the mappings of source asset id to destination asset id.
- **project_src** – The name of the project the object is being replicated from.
- **runtime** – The timestamp to be used in the new replicated metadata.

Returns The replicated time series to be created in the destination.

cognite.replicator.time_series.**replicate**(*client_src: cognite.client._cognite_client.CogniteClient, client_dst: cognite.client._cognite_client.CogniteClient, batch_size: int = 10000, num_threads: int = 1, delete_replicated_if_not_in_src: bool = False, delete_not_replicated_in_dst: bool = False, skip_unlinkable: bool = False, skip_nonasset: bool = False, target_external_ids: Optional[List[str]] = None, exclude_pattern: str = None*)

Replicates all the time series from the source project into the destination project.

Parameters

- **client_src** – The client corresponding to the source project.
- **client_dst** – The client corresponding to the destination project.
- **batch_size** – The biggest batch size to post chunks in.
- **num_threads** – The number of threads to be used.
- **delete_replicated_if_not_in_src** – If True, will delete replicated assets that are in the destination,
- **no longer in the source project** (*but*) –
- **delete_not_replicated_in_dst** – If True, will delete assets from the destination if they were not replicated
- **the source** (*from*) –

- **skip_unlinkable** – If no assets exist in the destination for a time series, do not replicate it
- **skip_nonasset** – If a time series has no associated assets, do not replicate it
- **target_external_ids** – List of specific time series external ids to replicate,
- **exclude_pattern** – Regex pattern; time series whose names match will not be replicated

cognite.replicator.time_series.**update_time_series**(*src_ts: cognite.client.data_classes.time_series.TimeSeries, dst_ts: cognite.client.data_classes.time_series.TimeSeries, src_dst_ids_assets: Dict[int, int], project_src: str, runtime: int*) → cognite.client.data_classes.time_series.TimeSeries

Makes an updated version of the destination time series based on the corresponding source time series.

Parameters

- **src_ts** – The time series from the source to be replicated.
- **dst_ts** – The time series from the destination that needs to be updated to reflect changes made to its source time series.
- **src_dst_ids_assets** – A dictionary of all the mappings of source asset id to destination asset id.
- **project_src** – The name of the project the object is being replicated from.
- **runtime** – The timestamp to be used in the new replicated metadata.

Returns The updated time series object for the replication destination.

Python Module Index

C

Index

T

U