Zendesk Import Integration

Treasure Data allows you to directly importing data from your organization's Zendesk account.

For sample workflows on importing data from Zendesk, view Treasure Boxes.

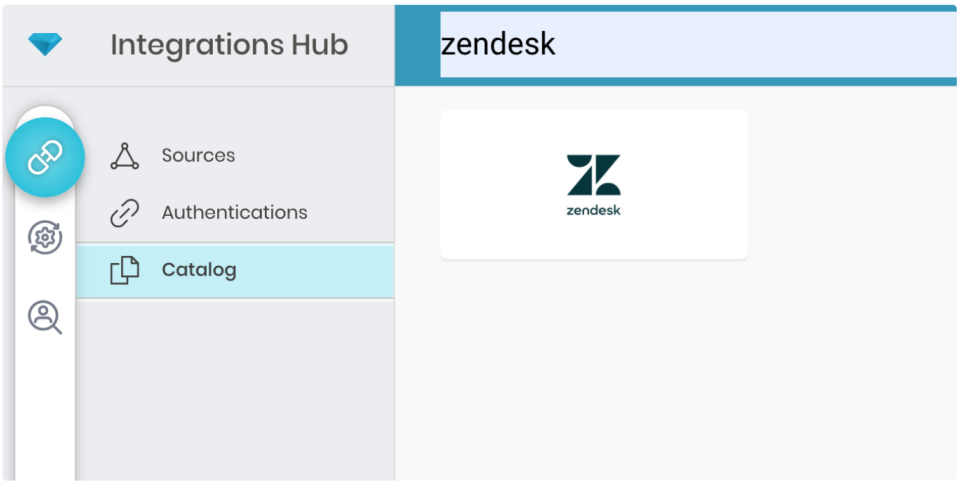
Prerequisites

- Basic knowledge of Treasure Data
- Zendesk account
- Zendesk Zopim account to retrieve Chat data

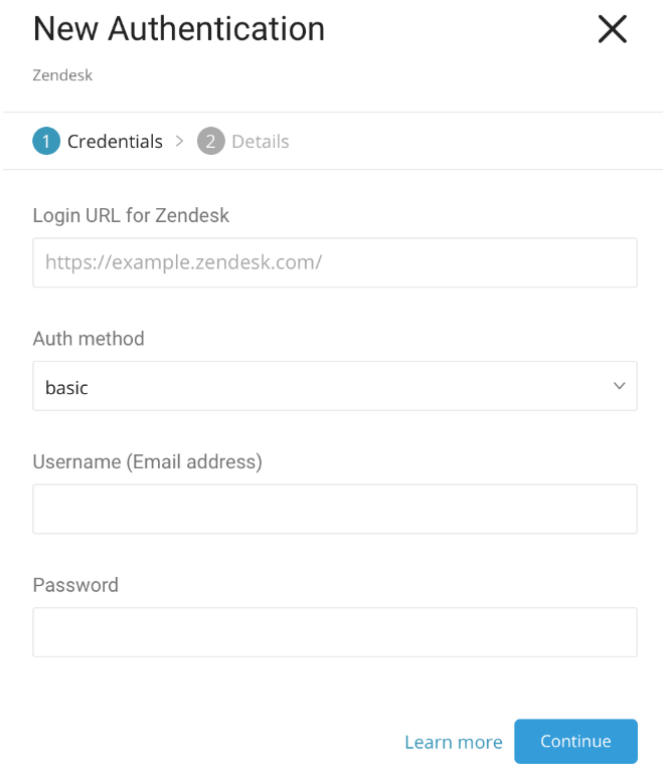
Use the TD Console to Create Your Connection

Create a New Connection

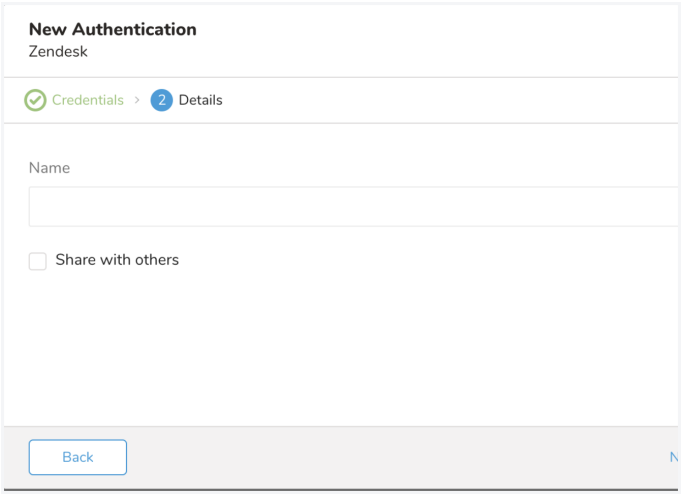
1. Open the TD Console.
2. Navigate to Integrations Hub > Catalog.
3. Search and select Zendesk.



4. Click **Create.** You are creating an authenticated connection. The following dialog opens.



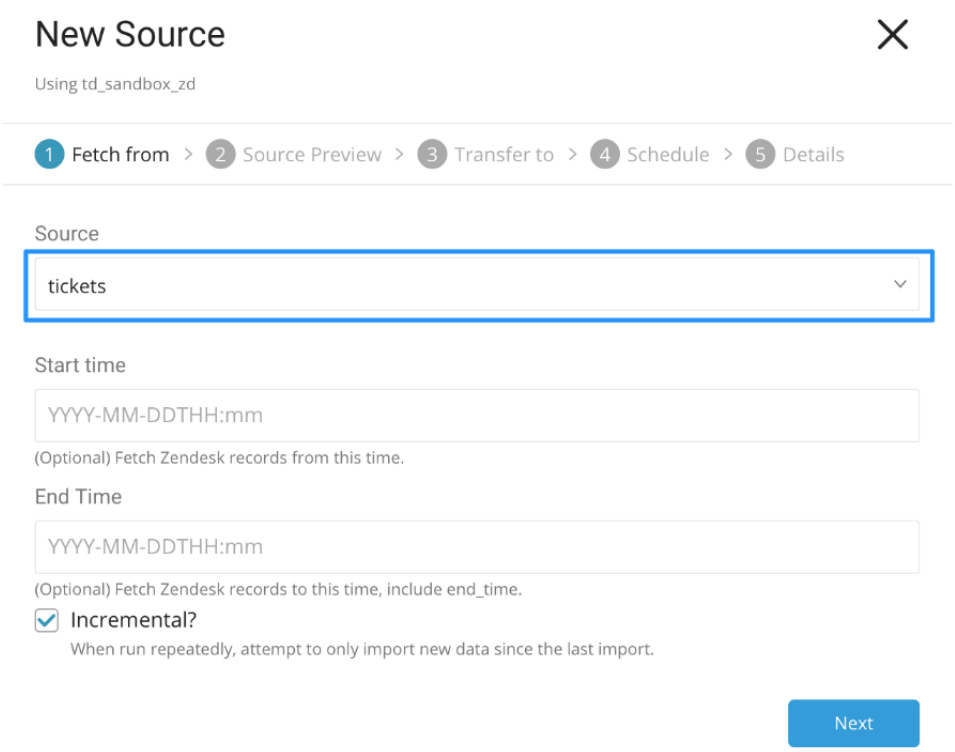
5. There are three options for **Auth method**: basic, token, oauth.
    a. To import chat data, enter in this url for Login Url for Zendesk: https://www.zopim.com. There is no support for token authentication with Chat.
6. Fill in all the required fields, then click **Continue**.
7. Name your new Zendesk connection. Click **Done**.



Transfer Your Data to Treasure Data

After creating the authenticated connection, you are automatically taken to the Authentications tab.

1. Search for the connection you created and click **New Source**.
2. Edit the appropriate fields.

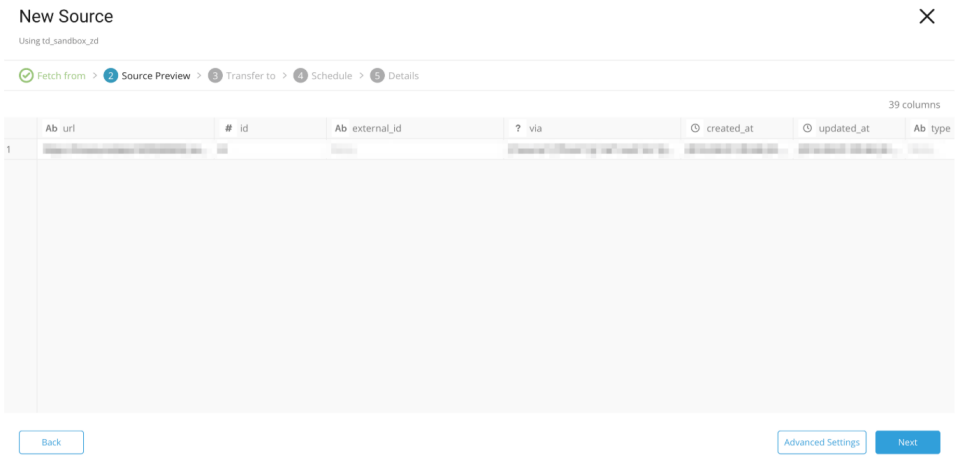


Source	Specify the kind of object that you want to transfer from Zendesk: tickets, ticket_fields, ticket_forms, ticket_events, ticket_metrics, users, organizations, scores, recipients, object_records, relationship_records, user_events, and chat
	• object_records and relationship_records provide information about Zendesk custom objects
	• scores and recipients provide information about Zendesk NPS
	• Chat Limitations:
	  • Only an administrator or owner has the permissions to retrieve chat data.
	  • No support for "Include Subresources" and "De-duplicate records" options

Incremental	Allows the connector to run in incremental mode, which enables **Start time** and **End time** can be used.
Start time	Enables you to select only objects, which have been updated since the 'start_time' • If **Start time** is not specified, all the objects are retrieved from the beginning.
End time	Enables you to select only objects, which have been updated up to the 'end_time'. • If **End time** is not specified, all the objects up to now are retrieved. • **Start time** and **End time** can combined to select only objects that have been updated within a specific period, from 'start_time' until 'end_time'

Source Preview

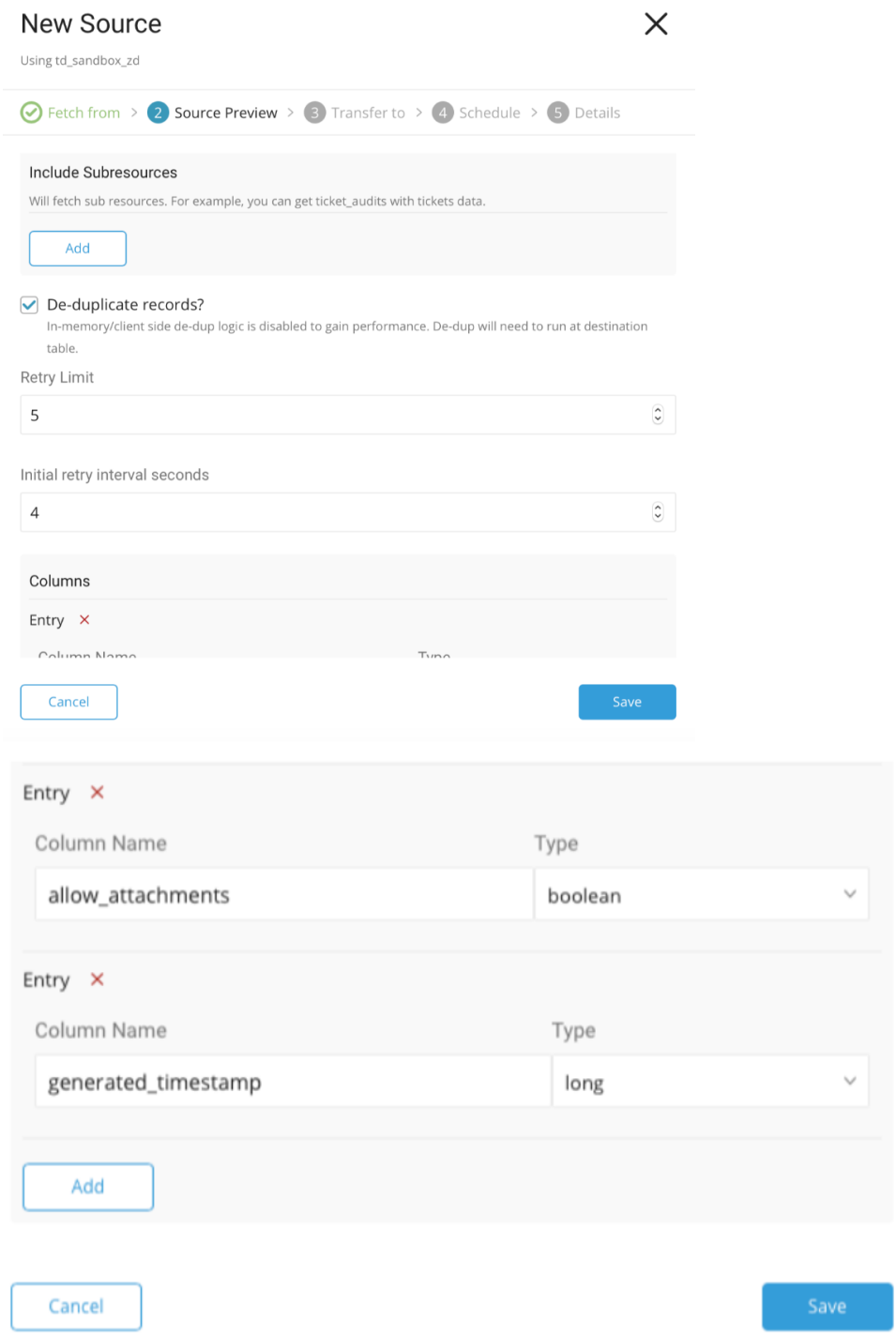
1. Preview your data. To make changes, click Advanced Settings.



2. Select Next.

Advanced Settings

1. After selecting Advanced Settings, the following dialog opens.



2. Edit the parameters. Select Save and Next.

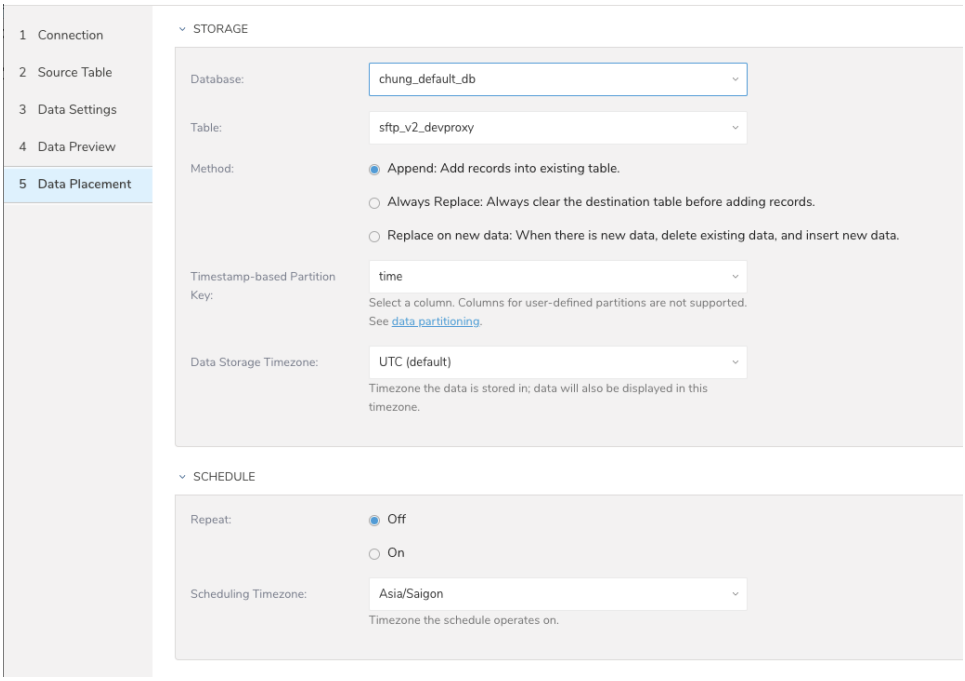
Parameter	Description
Include Subresources	Enables you to fetch sub-resource along with the main object. Click **Add** to add more sub resource by name and **Add** a correspondence column as well. The sub resource is considered as an JSON object, presented in a column, with the same name . • In Zendesk, this endpoint is supported: GET /api/v2/users/{user_id}/organizations.json That means we can consider organizations as a sub-resource of users. We can get all the information of organizations that a users belong to. • To configure it, you must add 'organizations' as a sub-resource and also add one more column with the same name. The data type should be JSON. • Include Subresources is not supported for Chat.

De-duplicated Records	Enables you to avoid duplicated records when running in incremental mode because the Zendesk API doesn't prevent duplication. • Deduplication is not supported for Chat.
Retry Limit	Indicates how many times the job should retry when error occurs.
Initial retry interval seconds	Indicates the first waiting time before a retry. Measured in seconds.

Data Transfer

For data placement, select the target database and table where you want your data placed and indicate how often the import should run.

1. Select **Next.** Under Storage you will create a new or select an existing database and create a new or select an existing table for where you want to place the imported data.



2. Select a **Database** > **Select an existing** or **Create New Database**.
3. Optionally, type a database name.
4. Select a **Table**> **Select an existing** or **Create New Table**.
5. Optionally, type a table name.
6. Choose the method for importing the data.
    - **Append** (default)-Data import results are appended to the table.
      If the table does not exist, it will be created.
    - **Always Replace**-Replaces the entire content of an existing table with the result output of the query. If the table does not exist, a new table is created.
    - **Replace on New Data**-Only replace the entire content of an existing table with the result output when there is new data.
7. Select the **Timestamp-based Partition Key** column.
   If you want to set a different partition key seed than the default key, you can specify the long or timestamp column as the partitioning time. As a default time column, it uses upload_time with the add_time filter.
8. Select the **Timezone** for your data storage.
9. Under **Schedule**, you can choose when and how often you want to run this query.
    - Run once:
        a. Select **Off**.
        b. Select **Scheduling Timezone**.
        c. Select **Create & Run Now**.
    - Repeat the query:
        a. Select **On**.
        b. Select the **Schedule**. The UI provides these four options: *@hourly*, *@daily* and *@monthly* or custom *cron*.
        c. You can also select **Delay Transfer** and add a delay of execution time.
        d. Select **Scheduling Timezone**.
        e. Select **Create & Run Now**.

After your transfer has run, you can see the results of your transfer in **Data Workbench** > **Databases**.

Use Command Line

Install ‘td’ Command

Install the newest TD Toolbelt.

Create Seed Config File (seed.yml)

Prepare `seed.yml` as shown in the following example, with your `login_url`, `username` (email), `token` and, `target`. In this example, you use “append” mode:

```
in:
  type: zendesk
  login_url: https://<YOUR_DOMAIN_NAME>.zendesk.com
  auth_method: token
  username: <YOUR_EMAIL_ADDRESS>
  token: <YOUR_API_TOKEN>
  target: tickets
  start_time: "2007-01-01 00:00:00+0000"
out:
  mode: append
```

`token` can be created by going to `Admin Home` –> `CHANNELS` –> `API` –> `"add new token"` (`https://<YOUR_DOMAIN_NAME>.zendesk.com /agent/admin/api`).

`target` specifies the type of object that you want to dump from Zendesk. tickets, ticket_events, ticket_forms, ticket_fields, users, organizations, scores, recipients, object_records, relationship_records and user_events are supported.

For more details on available `out` modes, see Appendix.

Guess Fields (Generate load.yml)

Use `connector:guess`. This command automatically reads the target data, and intelligently guesses the data format.

```
$ td connector:guess seed.yml -o load.yml
```

If you open the `load.yml` file, you see guessed file format definitions including, in some cases, file formats, encodings, column names, and types.

```
---in: type: zendesk login_url: https://<YOUR_DOMAIN_NAME>.zendesk.com auth_method: token username:
<YOUR_EMAIL_ADDRESS> token: <YOUR_API_TOKEN> target: tickets start_time: '2019-05-15T00:00:00+00:00' columns: -
{name: url, type: string} - {name: id, type: long} - {name: external_id, type: string} - {name: via, type:
json} - {name: created_at, type: timestamp, format: "%Y-%m-%dT%H:%M:%S%z"} - {name: updated_at, type:
timestamp, format: "%Y-%m-%dT%H:%M:%S%z"} - {name: type, type: string} - {name: subject, type: string} - {name:
raw_subject, type: string} - {name: description, type: string} - {name: priority, type: string} - {name:
status, type: string} - {name: recipient, type: string} - {name: requester_id, type: string} - {name:
submitter_id, type: string} - {name: assignee_id, type: string} - {name: organization_id, type: string} -
{name: group_id, type: string} - {name: collaborator_ids, type: json} - {name: follower_ids, type: json} -
{name: email_cc_ids, type: json} - {name: forum_topic_id, type: string} - {name: problem_id, type: string} -
{name: has_incidents, type: boolean} - {name: is_public, type: boolean} - {name: due_at, type: string} - {name:
tags, type: json} - {name: custom_fields, type: json} - {name: satisfaction_rating, type: json} - {name:
sharing_agreement_ids, type: json} - {name: fields, type: json} - {name: followup_ids, type: json} - {name:
ticket_form_id, type: string} - {name: brand_id, type: string} - {name: satisfaction_probability, type: string}
- {name: allow_channelback, type: boolean} - {name: allow_attachments, type: boolean} - {name:
generated_timestamp, type: long}out: {mode: append}exec: {}filters:- from_value: {mode: upload_time}
to_column: {name: time}  type: add_time
```

Then you can preview how the system parses the file by using `preview` command.

```
$ td connector:preview load.yml
```

If the system detects your column name or type unexpectedly, modify the `load.yml` directly and preview again.

The Data Connector supports parsing of "boolean", "long", "double", "string", and "timestamp" types.

Execute Load Job

Submit the load job. It may take a couple of hours depending on the data size. Users need to specify the database and table where their data is stored.

```
$ td connector:issue load.yml --database td_sample_db --table td_sample_table
```

The preceding command assumes that you have already created *database(td_sample_db)* and *table(td_sample_table)*. If the database or the table do not exist in TD this command will not succeed, so create the database and table manually or use `--auto-create-table` option with `td connector:issue` command to automatically create the database and table:

```
$ td connector:issue load.yml --database td_sample_db --table td_sample_table --time-column created_at --auto-create-table
```

⚠ You can assign Time Format column to the "Partitioning Key" by "--time-column" option.

Incremental Load

You can load records incrementally from Zendesk by using the `incremental` flag. If `False`, the `start_time` and `end_time` in next.yml is not updated. The connector will always fetch all the data from Zendesk with static conditions. If `True`, the `start_time` and `end_time` is updated in next.yml. The default is `True`.

```
in:
  type: zendesk
  login_url: https://<YOUR_DOMAIN_NAME>.zendesk.com
  auth_method: token
  username: <YOUR_EMAIL_ADDRESS>
  token: <YOUR_API_TOKEN>
  target: tickets
  start_time: "2007-01-01 00:00:00+0000"  end_time: "2008-01-01 00:00:00+0000"
  incremental: true
out:
  mode: append
```

Scheduled Execution

You can schedule periodic data connector execution for periodic Zendesk import. We carefully configure our scheduler to ensure high availability. By using this feature, you no longer need a `cron` daemon on your local data center.

Create the Schedule

A new schedule can be created by using the `td connector:create` command. The name of the schedule, cron-style schedule, the database and table where their data is stored, and the data connector configuration file are required.

```
$ td connector:create \
    daily_zendesk_import \
    "10 0 * * *" \
    td_sample_db \
    td_sample_table \
    load.yml
```

The `cron` parameter also accepts three special options: `@hourly`, `@daily` and `@monthly`.

By default, schedule is setup in UTC timezone. You can set the schedule in a timezone using -t or --timezone option. The `--timezone` option only supports extended timezone formats like 'Asia/Tokyo', 'America/Los_Angeles' etc. Timezone abbreviations like PST, CST are *not* supported and may lead to unexpected schedules.

List the Schedules

You can see the list of scheduled entries by `td connector:list`.

```
$ td connector:list
+----------------------+--------------+----------+-------+--------------+-----------------
+--------------------------+
| Name                 | Cron         | Timezone | Delay | Database     | Table           |
Config                   |
+----------------------+--------------+----------+-------+--------------+-----------------
+--------------------------+
| daily_zendesk_import | 10 0 * * *   | UTC      | 0     | td_sample_db | td_sample_table | {"type"=>"zendes",
... } |
+----------------------+--------------+----------+-------+--------------+-----------------
+--------------------------+
```

Show the Setting and History of Schedules

`td connector:show` shows the execution setting of a schedule entry.

```
% td connector:show daily_zendesk_import
Name     : daily_zendesk_import
Cron     : 10 0 * * *
Timezone : UTC
Delay    : 0
Database : td_sample_db
Table    : td_sample_table
Config
---
// Displayed load.yml configuration.
```

`td connector:history` shows the execution history of a schedule entry. To investigate the results of each individual execution, use `td job <jobid>`.

```
% td connector:history daily_zendesk_import
+--------+---------+---------+--------------+-----------------+----------+--------------------------+----------
+
| JobID  | Status  | Records | Database     | Table           | Priority | Started                  | Duration
|
+--------+---------+---------+--------------+-----------------+----------+--------------------------+----------
+
| 578066 | success | 10000   | td_sample_db | td_sample_table | 0        | 2015-04-18 00:10:05 +0000 | 160
|
| 577968 | success | 10000   | td_sample_db | td_sample_table | 0        | 2015-04-17 00:10:07 +0000 | 161
|
| 577914 | success | 10000   | td_sample_db | td_sample_table | 0        | 2015-04-16 00:10:03 +0000 | 152
|
| 577872 | success | 10000   | td_sample_db | td_sample_table | 0        | 2015-04-15 00:10:04 +0000 | 163
|
| 577810 | success | 10000   | td_sample_db | td_sample_table | 0        | 2015-04-14 00:10:04 +0000 | 164
|
| 577766 | success | 10000   | td_sample_db | td_sample_table | 0        | 2015-04-13 00:10:04 +0000 | 155
|
| 577710 | success | 10000   | td_sample_db | td_sample_table | 0        | 2015-04-12 00:10:05 +0000 | 156
|
| 577610 | success | 10000   | td_sample_db | td_sample_table | 0        | 2015-04-11 00:10:04 +0000 | 157
|
+--------+---------+---------+--------------+-----------------+----------+--------------------------+----------
```

```
+
8 rows in set
```

Delete the Schedule

`td connector:delete` removes the schedule.

```
$ td connector:delete daily_zendesk_import
```

Appendix

Modes for Out Plugin

You can specify file import mode in `out` section of seed.yml.

append (default)

This is the default mode and records are appended to the target table.

```
in:
  ...
out:
  mode: append
```

replace

This mode replaces data in the target table. Any manual schema changes made to the target table remain intact with this mode.

```
in:
  ...
out:
  mode: replace
```

Include Sub Resources

You can specify `includes` option to get related objects.

For example, if you want to get tickets data with comments, use a configuration as follows:.

```
in:
  type: zendesk
  target: tickets
  includes:
    - comments
  ...
out:
  mode: replace
```

Further Information

- List of Options for Zendesk Data Connector