Oracle Eloqua Export Integration

You can write Treasure Data job results directly to your Eloqua contact records and custom object. You use this integration and TD query jobs to add contacts to Eloqua Contact lists or add data to custom data objects (CDO).

This feature is in BETA. Contact your Customer Success Representative for more information.

- Prerequisites
- Requirements and Limitations
- Obtain the Custom Data Object Name from Oracle Eloqua Console
- Use the TD Console to Create Your Connection
    - Create a New Connection
- Define your Query
    - Integration Parameters for Eloqua
    - Example Query
- Optionally Schedule the Query Export Jobs
        - Custom cron... Details
    - Execute the Query
- Optionally Configure Export Results in Workflow
    - Example Workflow for Exporting Contacts
    - Example Workflow for Exporting Custom Object

Prerequisites

- Basic knowledge of Treasure Data, including the TD Toolbelt.
- An Eloqua account with API access enabled.

Requirements and Limitations

Your query requires that each type of resource has:

- specific columns with exact column names (case sensitive)
- data types

You must define the column mapping in the query.

Custom object fields, such as col_a, col_b, col_c, can be case insensitive.

If the custom object field contains a space, replace it with an underscore.

If your query contains any field name that does not exist in the custom object fields name, an error is returned, and the job fails.

Obtain the Custom Data Object Name from Oracle Eloqua Console

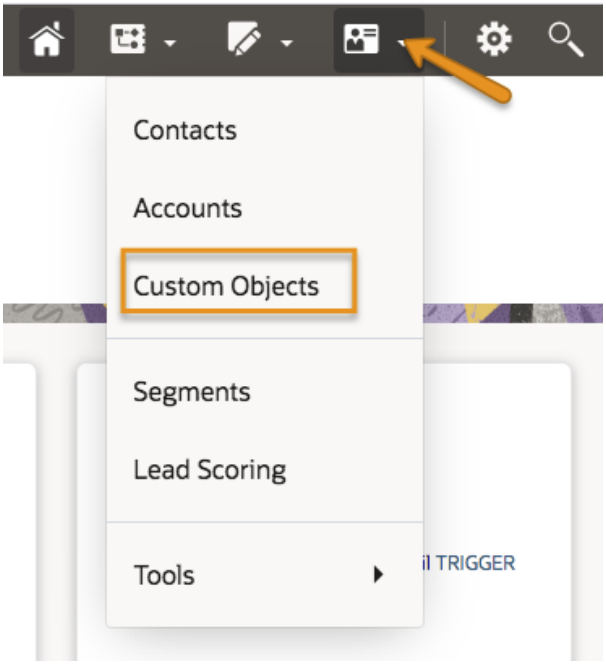
1. Login into Oracle Eloqua Console.

```
https://login.eloqua.com/
```

2. Enter the same company, user name, and password you entered when specifying the CSF authentication keys in Oracle JDeveloper and Oracle Enterprise Manager Fusion Middleware Control.

3. Navigate to **Audience**

4. Select **Custom Objects**.



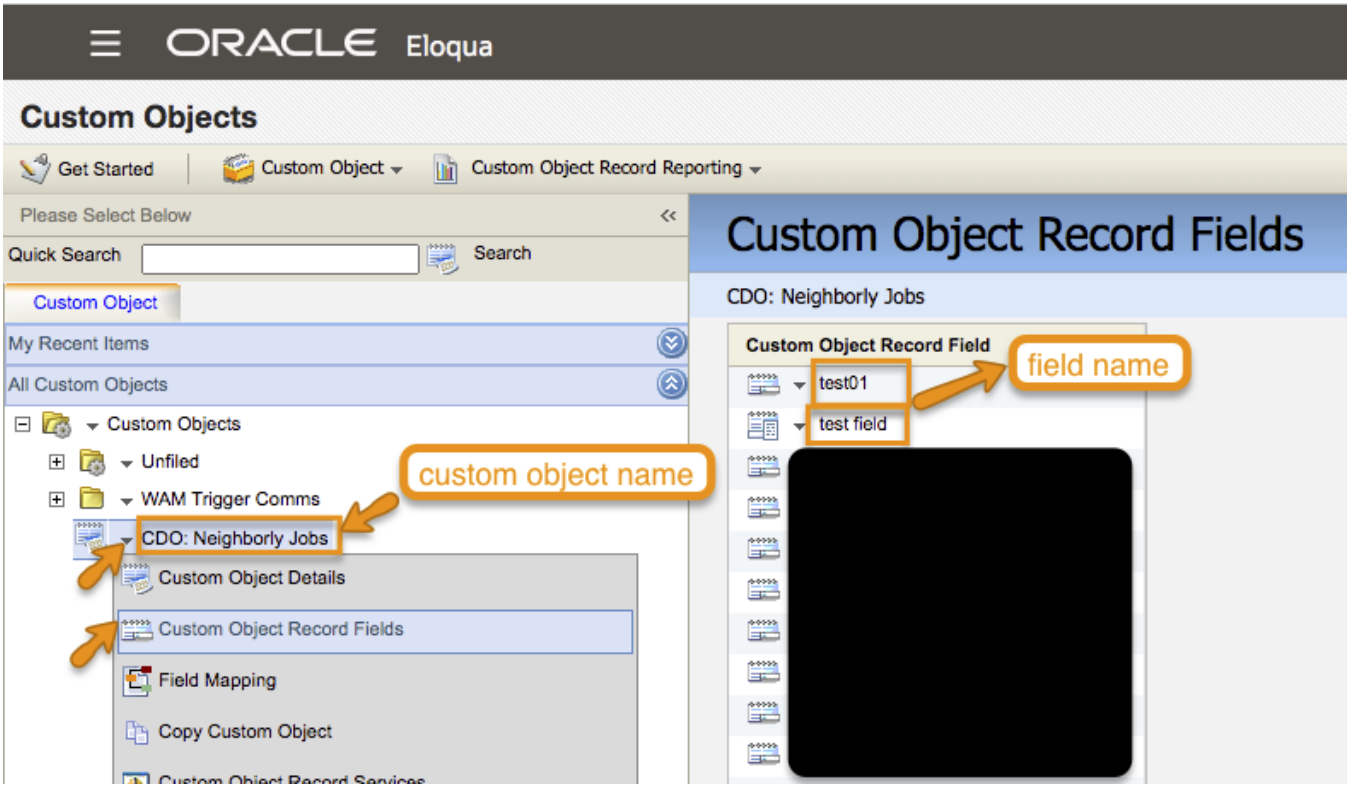
5. Optionally, navigate to **Custom Object Record Reporting** and select the desired report:

- **Custom Object Data**:
  - Use this report to view all the custom object records within the individual custom object.
  - You can print and export the report to Excel and copy, delete, or move the custom object records.
- **Custom Object and Contact Data**:
  - Use this report to view all the custom object records data within the individual custom object with the contact information they are mapped to. The report can be viewed in a specified time frame. The default is the past month, but you can select a different time range by changing the time range (the **Start Date** and **End Date**) or time span (for example, **Last week** or **Last month**) and select:

  - You can print and export the report to Excel.

6. Make note of the custom object name and the field names.

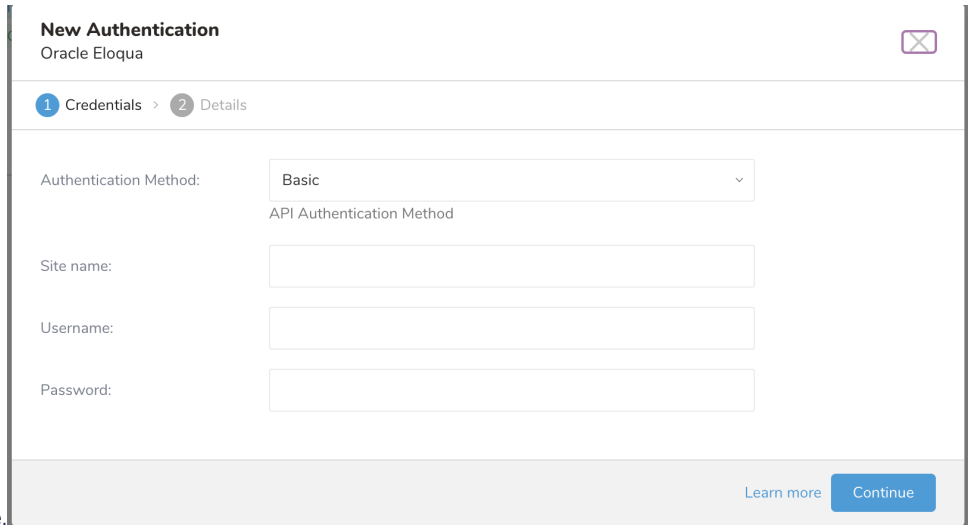


Use the TD Console to Create Your Connection

Create a New Connection

In Treasure Data, you must create and configure the data connection before running your query. As part of the data connection, you provide authentication to access the integration.

1. Open **TD Console**.
2. Navigate to **Integrations Hub** > **Catalog**.
3. Search for and select **Eloqua**.



4. Type the credentials to authenticate.
5. Type a name for your connection.
6. Select **Done.**

Define your Query

Your query requires each resource type to have specific columns and exact column names (case sensitive) and data types.

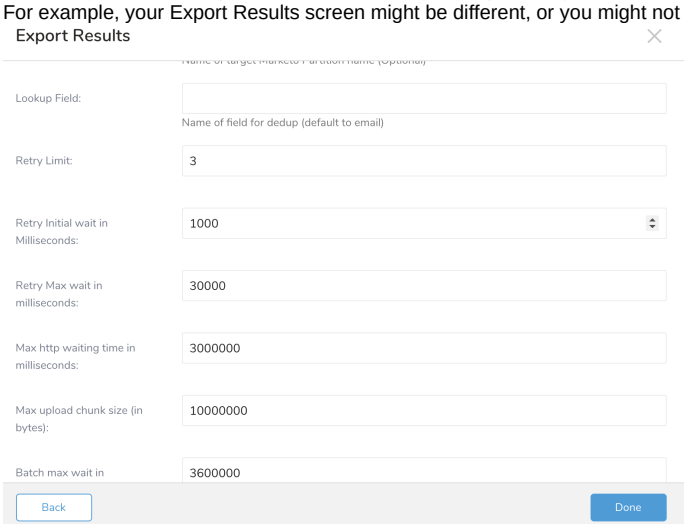
Custom object fields, such as col_a, col_b, col_c, can be case insensitive.

If the custom object field contains a space, replace it with an underscore.

If your query contains any field name that does not exist in the custom object fields name, an error is returned, and the job fails.

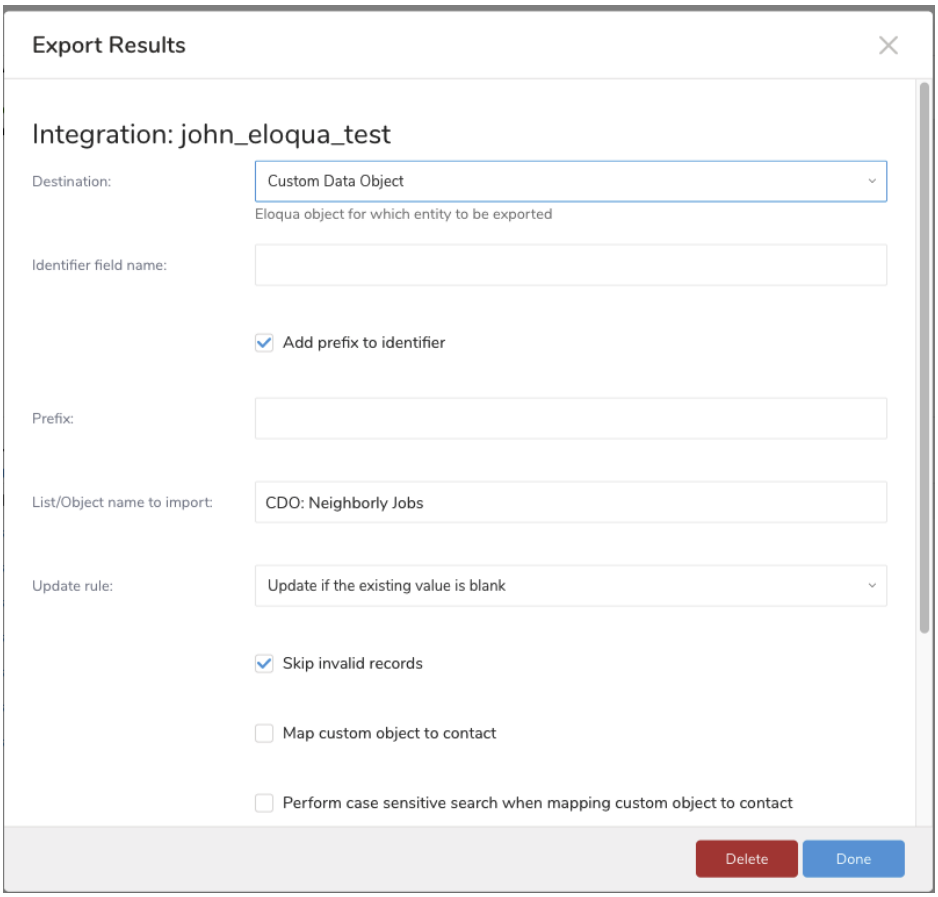
1. Complete the instructions in Creating a Destination Integration.
2. Navigate to **Data Workbench > Queries**.
3. Select a query for which you would like to export data.
4. Run the query to validate the result set.
5. Select **Export Results**.
6. Select an existing integration authentication.

7. Define any additional Export Results details. In your export integration content review the integration parameters. For example, your Export Results screen might be different, or you might not have additional details to fill out:



8. Select **Done**.
9. Run your query.
10. Validate that your data moved to the destination you specified.

Integration Parameters for Eloqua



Contact and Custom Object Data Parameters	**Value**	**Description**
Destination	Contact	The Eloqua object to export data (contact list or custom object)
	Custom Object Data	The Eloqua object to export data (contact list or custom object)
Identifier Field Name	- Contact's email address [c_emailaddress] - Contact's external ID [c_externalid] - Contact's ID [c_id]	The field name is used to identify the contact entity. If you chose a Contact's email address, you must have the c_emailaddress field in your query with the Contact's external ID as c_externalid field, and the Contact's ID is a c_id field.
	One of the field names from your custom object data query. For example, if your query is: `SELECT col_a, col_b, col_b FROM your_table;` then the value should be one col_a, col_b, or col_b.	The field name is used to identify the custom object row.
Add prefix to the identifier		Check if you want to modify the identifier value by adding a prefix value to the identifier column. This option is supported for an identifier column that has a data type of string type.

Prefix	string value	prefix value will be added to identify column value
List/Object name to import	Custom object names can be case insensitive.	The contact list name or object name for the exported data. The custom object name that we need to export data into.
Update Rule	- Always update - Update if the new value is not blank - Update if the existing value is blank - Use the rule defined at the field level	The rule is used when doing updates on existing data. For more information about this parameter, see the Oracle Eloqua.
Skip invalid records		If you checked when Eloqua synced data and you receive a data rows error, the export job is still a success. Otherwise, the job failed.
Map custom object to contact		If you checked, the job tries to map a custom object to a contact.
Perform case sensitive search when mapping custom object to contact		Perform a case-sensitive search when mapping a custom object to a contact.
CDO source field	Column name of the field	Input the column name of the field to be mapped, as used in the export schema. Required when Map custom object to contact is enabled.
Contact map field	Column name of the field	Input the contact field for mapping. Required when Map custom object to contact is enabled.

Example Query

Your query requires each resource type to have specific columns and exact column names (case sensitive) and data types.

Custom object fields, such as col_a, col_b, col_c, can be case insensitive.

If the custom object field contains a space, replace it with an underscore.

If your query contains any field name that does not exist in the custom object fields name, an error is returned, and the job fails.

custom object query

```
SELECT col_a, col_b, col_b
FROM your_table;
```

For example, the following query to export contact data does not match any real users and is for demonstration purposes only.

Contact

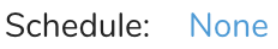
```
SELECT c_emailaddress, c_firstname, c_lastname
FROM contact;
```

```
SELECT identifier_type, identifier
FROM table my_table
```

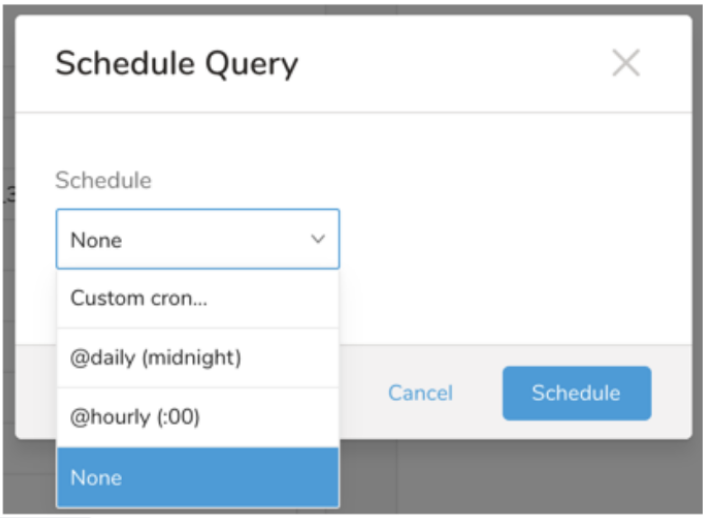
Optionally Schedule the Query Export Jobs

You can use Scheduled Jobs with Result Export to periodically write the output result to a target destination that you specify.

1. Navigate to **Data Workbench > Queries**.
2. Create a new query or select an existing query.
3. Next to **Schedule**, select None.

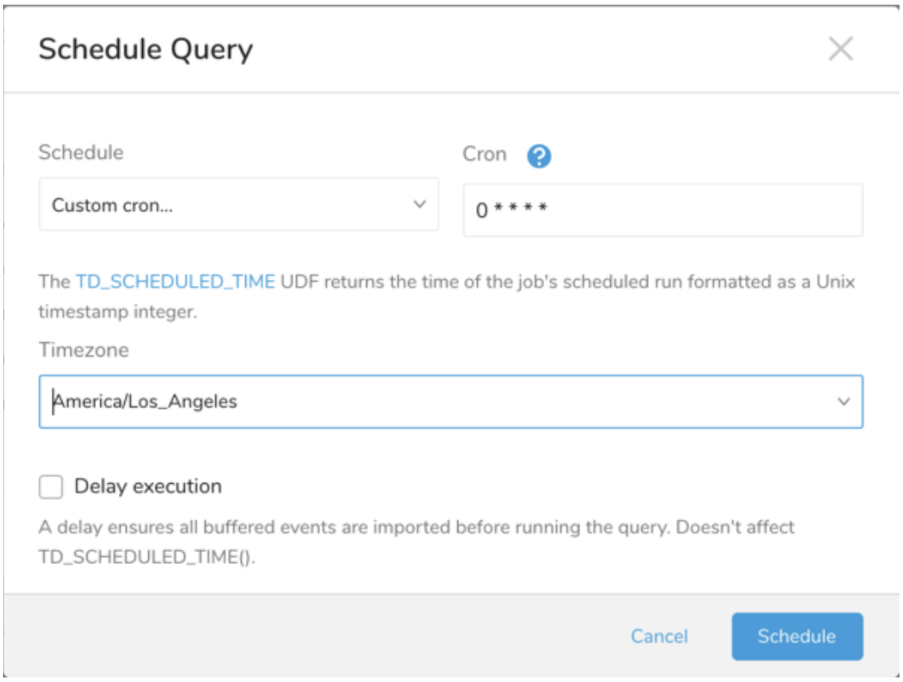


4. In the drop-down, select one of the following schedule options.



Drop-down Value	Description
Custom cron...	Review Custom cron... details.
@daily (midnight)	Run once a day at midnight (00:00 am) in the specified time zone.
@hourly (:00)	Run every hour at 00 minutes.
None	No schedule.

Custom cron... Details



Cron Value	Description
`0 * * * *`	Run once an hour
`0 0 * * *`	Run once a day at midnight
`0 0 1 * *`	Run once a month at midnight on the morning of the first day of the month
""	Create a job that has no scheduled run time.

```
 *    *    *    *    *
 -    -    -    -    -
 |    |    |    |    |
 |    |    |    |    +----- day of week (0 - 6) (Sunday=0)
 |    |    |    +---------- month (1 - 12)
 |    |    +--------------- day of month (1 - 31)
 |    +-------------------- hour (0 - 23)
 +------------------------- min (0 - 59)
```

The following named entries can be used:

- Day of Week: sun, mon, tue, wed, thu, fri, sat
- Month: jan, feb, mar, apr, may, jun, jul, aug, sep, oct, nov, dec

A single space is required between each field. The values for each field can be composed of:

Field Value	Example	Example Description
a single value, within the limits displayed above for each field.		
a wildcard `'*'` to indicate no restriction based on the field.	`'0 0 1 * *'`	configures the schedule to run at midnight (00:00) on the first day of each month.
a range `'2-5'`, indicating the range of accepted values for the field.	`'0 0 1-10 * *'`	configures the schedule to run at midnight (00:00) on the first 10 days of each month.
a list of comma-separated values `'2,3,4,5'`, indicating the list of accepted values for the field.	`0 0 1,11,21 * *'`	configures the schedule to run at midnight (00:00) every 1st, 11th, and 21st day of each month.

a periodicity indicator `'*/5'` to express how often based on the field's valid range of values a schedule is allowed to run.	`'30 */2 1 * *'`	configures the schedule to run on the 1st of every month, every 2 hours starting at 00:30. `'0 0 */5 * *'` configures the schedule to run at midnight (00:00) every 5 days starting on the 5th of each month.
a comma-separated list of any of the above except the `'*'` wildcard is also supported `'2,*/5,8-10'`.	`'0 0 5,*/10,25 * *'`	configures the schedule to run at midnight (00:00) every 5th, 10th, 20th, and 25th day of each month.

5. (Optional) If you enabled the Delay execution, you can delay the start time of a query.

Execute the Query

Save the query with a name and run, or just run the query. Upon successful completion of the query, the query result is automatically imported to the specified container destination.

> Scheduled jobs that continuously fail due to configuration errors may be disabled on the system side after several notifications.

Optionally Configure Export Results in Workflow

Within Treasure Workflow, you can specify the use of a data connector to export data.

Learn more at Using Workflows to Export Data with the TD Toolbelt.

Example Workflow for Exporting Contacts

The identifier_field_name accepts the following values:

- C_EmailAddress
- C_ExternalID
- C_ID

The update_rule accepts the following values:

- always
- ifNewIsNotNull
- ifExistingIsNull
- useFieldRule

```
_export:
  td:
    database: eloqua_db

+eloqua_custom_object_export_task:
  td>: export_contact.sql
  database: ${td.database}
  result_connection: new_created_eloqua_auth
  result_settings:
    type: eloqua
    target: contact
    identifier_field_name: C_EmailAddress

    list_name: td_shared_list2
    update_rule: always
    skipInvalidRecords: true
```

Example Workflow for Exporting Custom Object

The custom_object_identifier_field is one of the fields named in the export_contact.sql query.

The update_rule accepts the following values:

- always
- ifNewIsNotNull
- ifExistingIsNull
- useFieldRule

Obtain the custom_object_name from the Oracle Eloqua Console.

```
_export:
  td:
    database: eloqua_db

+eloqua_custom_object_export_task:
  td>: export_contact.sql
  database: ${td.database}
  result_connection: new_created_eloqua_auth
  result_settings:
    type: eloqua
    target: custom_object
    custom_object_identifier_field: {xxxxx}
    list_name: {custom_object_name}
    update_rule: always
    skipInvalidRecords: true
    mapCustomObjectToContact: true,
    performCaseSensitiveSearch: false,
    cdoSourceField: Email_Address,
    contactMapField: C_EmailAddress
```