Uploading existing data

MERMAID

Uploading existing data

Authors: Ariel Greiner (University of Toronto), Avery Schwarz (University of Toronto), Amelia Wenger (WCS and University of Queensland), Amkieltiela (Yayasan PADMI Mandiri and MERMAID), Alexandra Kler Lago (MERMAID, WCS) and Emily Darling (WCS and University of Toronto).

Issued in August 2024, v.1.0

Table of Contents

Introduction

Who is this tutorial for?

This tutorial is designed for users with a significant amount of existing or 'legacy' data from coral reef surveys—such as fish belt, benthic PIT, benthic LIT, benthic photo quadrat, bleaching quadrat, and habitat complexity tape-and-chain methods—stored in .xlsx or .csv formats, who are seeking an efficient method to upload this data to MERMAID.

The workflow

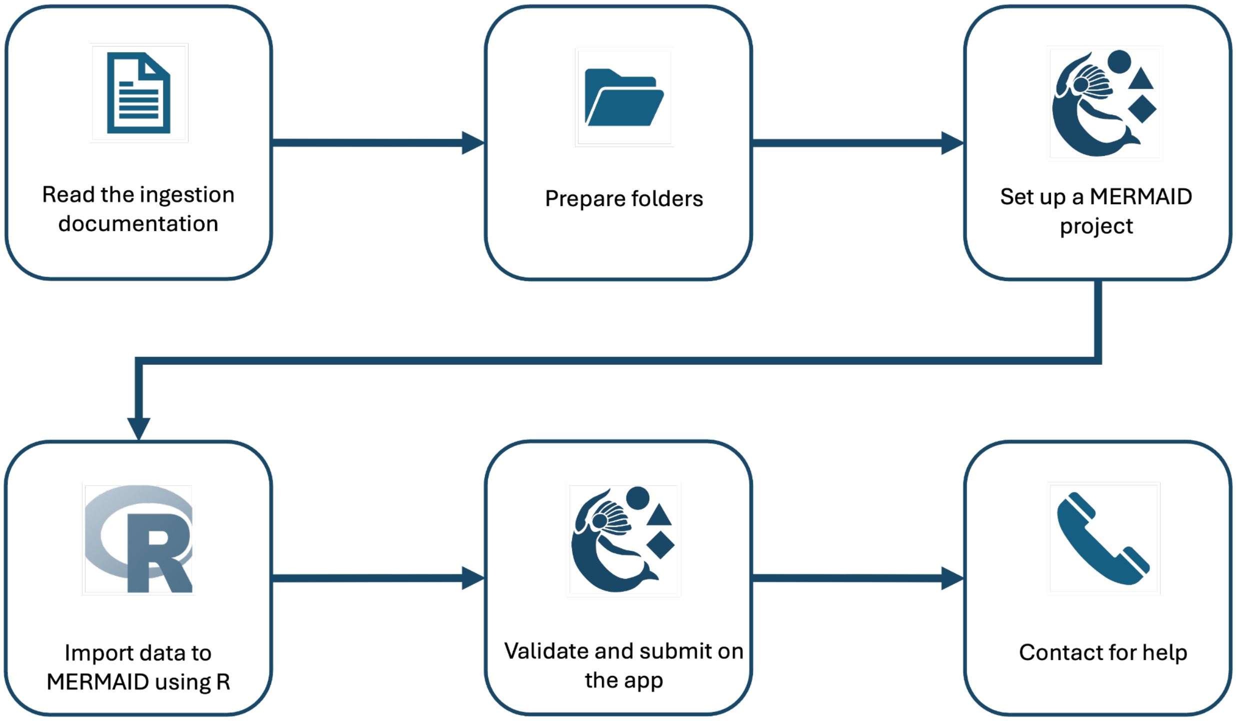

Read the ingestion documentation
Prepare folders
Set up a MERMAID project
Import data to MERMAID using R
Validate and submit on the app
Contact for help

Figure 1. Workflow for uploading existing data to MERMAID

The workflow to upload data from Excel to MERMAID involves several key steps (Figure 1):

1. **Read the documentation:** Review the documentation on datamermaid.org.
2. **Prepare folders:** Organize your data into distinct projects (folders on your computer).

3. **Set up a MERMAID project:** Create a new project in MERMAID, add organizations, add users and their permissions, set data sharing policies, and add sites and management regimes. More information can be found on the documentation.
4. **Prepare data:** using R and the R package **mermaidr**, download the .csv template for your survey type, load your data, adjust the format to match the template, and check all columns to ensure they match the accepted options.
5. **Upload data:** Follow the steps to upload your data to MERMAID using R.

You should only use this upload workflow for existing, or “legacy” data. For ongoing projects, it is easier and faster to enter observations directly in the MERMAID app.

Uploading existing data can be used for all methods supported by MERMAID, i.e., fishbelt, benthic PIT, benthic LIT, benthic photo quadrat, bleaching, and habitat complexity.

You can contact the MERMAID team (contact@datamermaid.org) for additional support.

Useful R packages for preparation of data upload to MERMAID

Package name	Description
here	install.packages(here) ○ Creates a path to your projects files, then ‘here()’ can be used as an easy replacement for the full ‘file.path()’ ○ eg. *read_csv(here("import-schemas", "benthic_pit_template.csv"))* ‘here’ replaces need for “MERMAID-Import’
tidyverse	install.packages(tidyverse) Tidyverse package containing many useful subpackages such as dplyr and tidyr, allow for easy manipulation of data
janitor	install.packages(janitor) ○ Helps format data frame column names ○ *clean_names()* function creates tidy, uniform column names by changing to lowercase letters with _ as separator, numbering duplicate names and more

mermaidr	remotes::install_github("data-mermaid/mermaidr") ○ Package that allows you to access data directly from MERMAID into R ○ mermaid_get_my_projects() ■ Returns information on all MERMAID projects you are currently part of ■ Can be used in combination with dplyr::filter to select one or more specific projects ○ mermaid_get_project_data() ■ Returns specific data from the project of interest, eg. fishbelt, bleaching ○ mermaid_get_reference() ■ Provides lists of either fish or benthic attributes that are accepted in MERMAID. Useful to check your fish names or benthic attributes against these to ensure exact match More info on installing and using the mermaidr package can be seen in this link: https://data-mermaid.github.io/mermaidr/
dplyr	install.packages(dplyr) The dplyr package makes data manipulation much easier - allows you to perform several useful operations: ○ Rows: ■ filter() chooses rows based on column values. ■ slice() chooses rows based on location. ■ arrange() changes the order of the rows. ○ Columns: ■ select() changes whether or not a column is included. ■ rename() changes the name of columns. ■ mutate() changes the values of columns and creates new columns. ■ relocate() changes the order of the columns. ○ Groups of rows: ■ summarise() collapses a group into a single row.

Uploading Excel spreadsheet to MERMAID

Prepare folders

To streamline your data upload to the MERMAID platform, it is essential to organize your data into distinct projects, each represented by a folder on your computer (Figure 2). MERMAID datasets are structured around projects, so this organization will facilitate a smoother transition. Here are some guidelines to help you organize your data files effectively:

1. Set up a dedicated folder for each project, organized by region and year(s). For example:
   - **Kubulau 2009-2011**
   - **Southern Bua 2011**
2. Within each project folder, create two subfolders to store data files:
   - **FishData:** For storing fish-related data.
   - **BenthicData:** For storing benthic-related data.
3. Ensure that the Excel spreadsheets in the **FishData** and **BenthicData** folders include the region name and the data type in their file names. For example:
   - For fish data: **Kubulau_Fish2011.xlsx**
   - For benthic data: **Kubulau_Benthic2011.xlsx**
4. It is recommended to copy the original Excel spreadsheets into the newly created project folders. Maintain a separate, unmodified copy of the original files in their initial location. This precaution allows you to make necessary modifications to the copies in the project folders without altering the original datasets.

By following these guidelines, you will ensure that your data is well-organized and ready for upload to MERMAID.

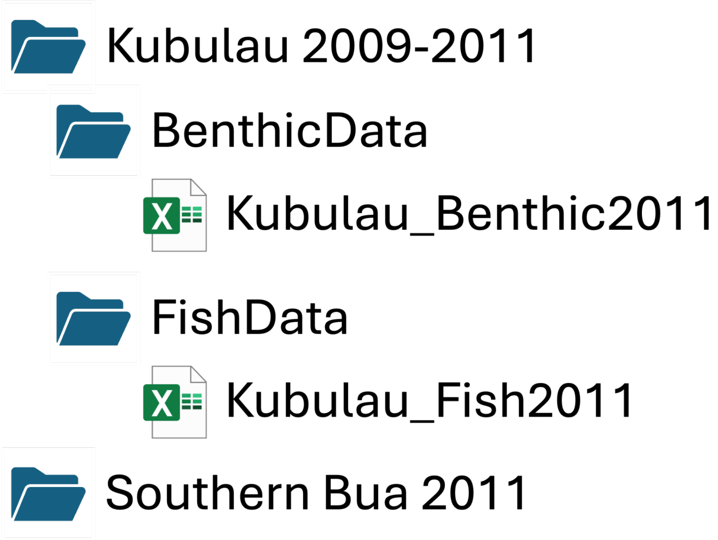


Figure 2. Example of how project files should be organized and named on your computer

Setup a MERMAID project

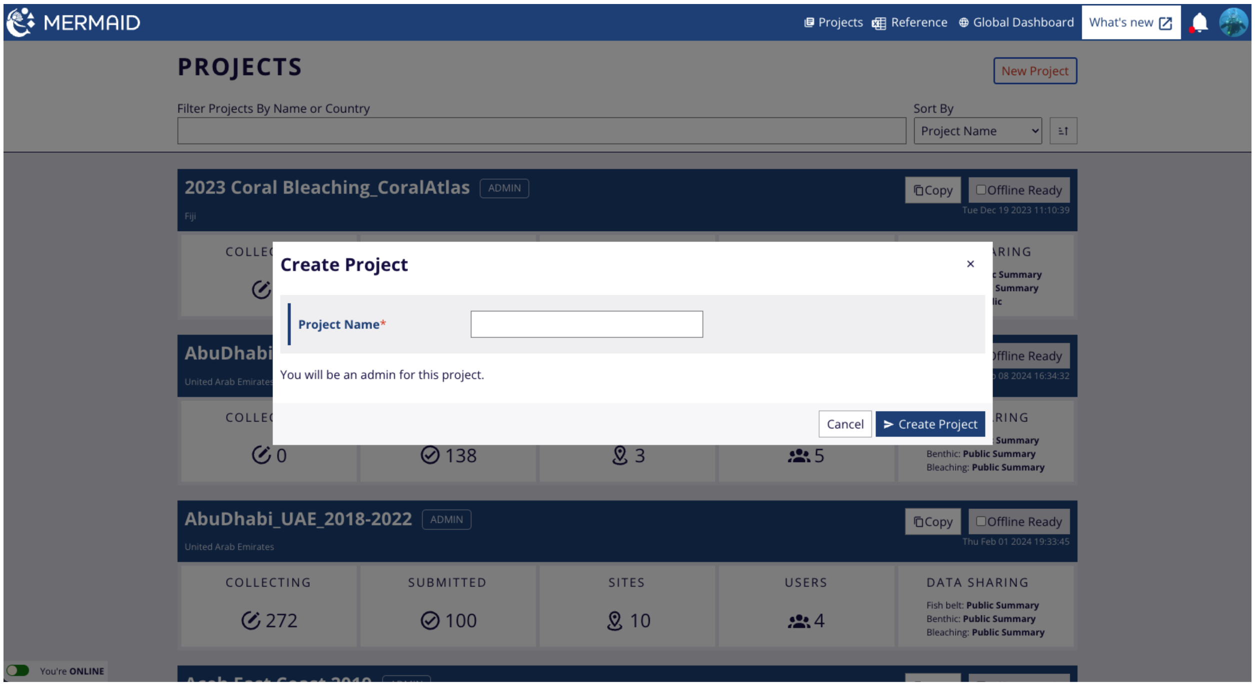
To set up a project in MERMAID corresponding to the project in your computer folder, follow the steps below. You can also refer to the documentation.

You can enter or update project details at any time. There is no need to have all information finalized when first creating the project. Focus on the essential details, such as adding organization(s), users, data sharing policies, sites, and management regimes, and refine them as needed.

1. Create your project

The first step to set up a MERMAID project is to create a new project (Figure 3).

a. Go to 'https://app.datamermaid.org'.
b. Click on the **New Project** button.
c. Name the project (e.g. MERMAID reef survey).



Create Project

Project Name*

You will be an admin for this project.

Cancel | Create Project

Figure 3. Creating a new project in MERMAID

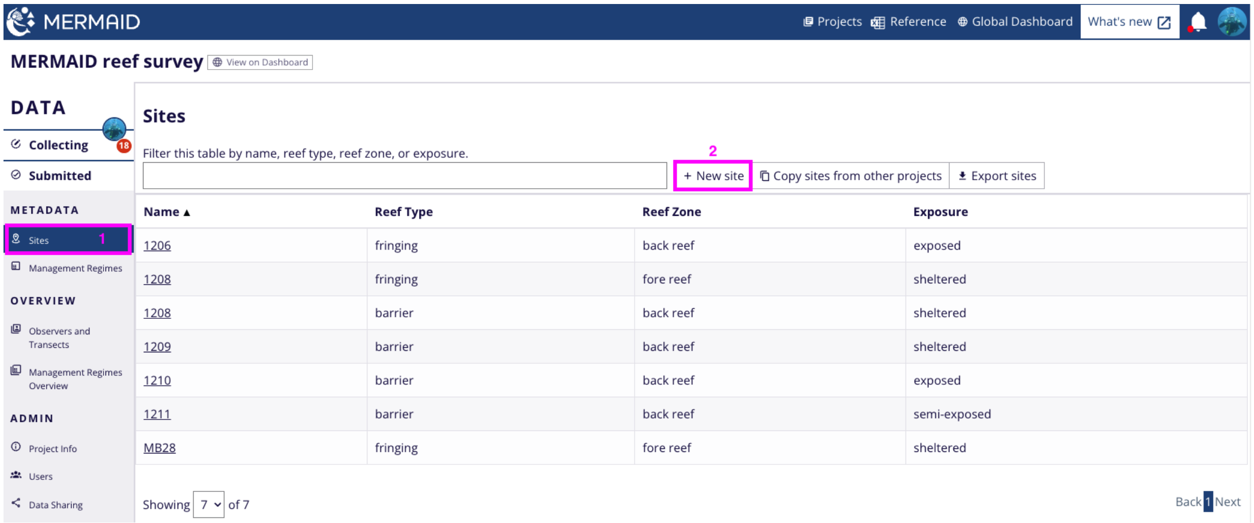
2. Add sites

Once you have created your project and added users, proceed by adding sites to the project. Follow these steps (Figure 7):

a. Click on the **Sites** menu option under **METADATA** in the left side menu

b. Click **+ New Site**. You can also add sites by copying from existing projects by clicking **Copy sites from other projects**.

Note: Each site needs to be entered only once, regardless of the number of surveys conducted at that site.



Filter this table by name, reef type, reef zone, or exposure.

Name ▲	Reef Type	Reef Zone	Exposure
1206	fringing	back reef	exposed
1208	fringing	fore reef	sheltered
1208	barrier	back reef	sheltered
1209	barrier	back reef	sheltered
1210	barrier	back reef	exposed
1211	barrier	back reef	semi-exposed
MB28	fringing	fore reef	sheltered

Showing 7 of 7

Figure 7. Adding new sites to a project in MERMAID

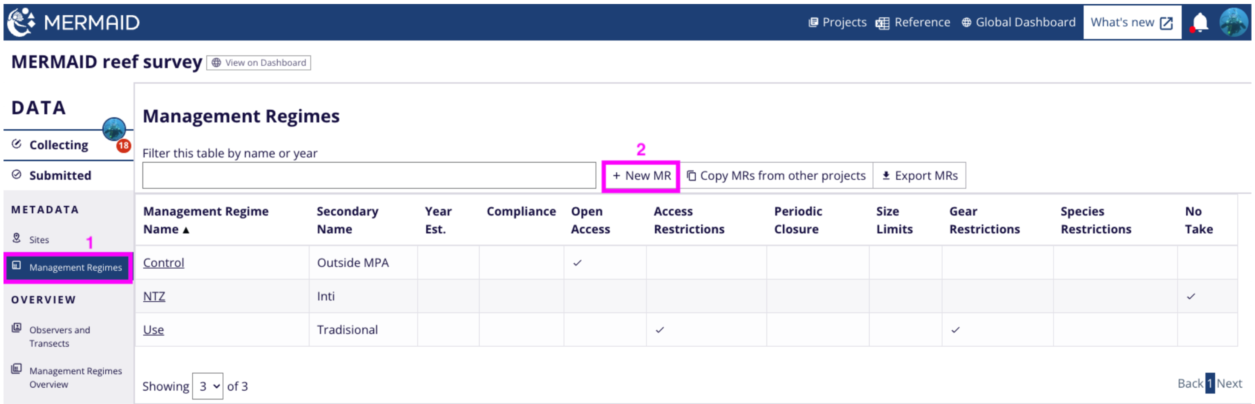
To add a new site, ensure you have the following information:

a. **Name**.
b. **Country**.
c. **Latitude** & **Longitude**.
d. **Exposure** (i.e., **very sheltered**, **sheltered**, **semi-exposed**, or **exposed**).
e. **Reef Type** (i.e., **atoll**, **barrier**, **fringing**, or **lagoon**).
f. **Reef Zone** (i.e., **back reef**, **bank**, **channel**, **crest**, **fore reef**, **pinnacle**).

3. Add management regimes

A management regime categorizes the regulations and restrictions placed on an area in which a survey site is located, on the data of the survey. To add management regimes:

a. Navigate to **Management Regimes** under **METADATA** in the left side menu in a project.
b. You can create new management regimes by clicking **+ New MR** (Figure 8) or copy from existing ones by clicking **Copy MRs from other projects**.
c. To add new management regimes, it is mandatory to fill out the **Management Regime Name** and **Rules** applied (Figure 9). Other fields, such as **Secondary Name**, **Year Established**, **Area**, **Parties**, **Compliance**, and **Notes** are optional.



MERMAID reef survey

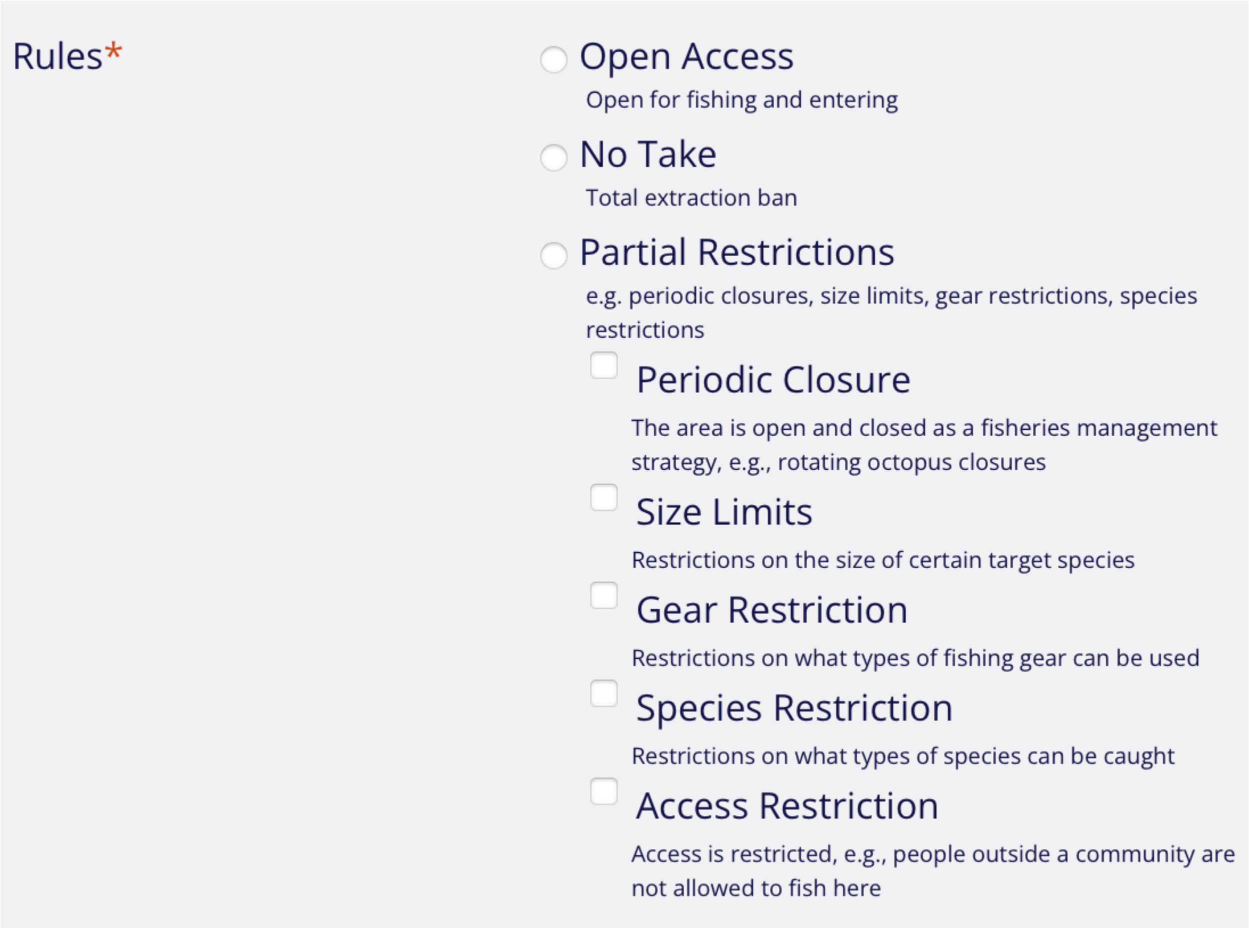
Management Regimes

Filter this table by name or year

Management Regime Name ▲	Secondary Name	Year Est.	Compliance	Open Access	Access Restrictions	Periodic Closure	Size Limits	Gear Restrictions	Species Restrictions	No Take
Control	Outside MPA			✓						
NTZ	Inti									✓
Use	Tradisional				✓			✓		

Showing 3 of 3

Figure 8. Adding new management regimes to a project in MERMAID



Rules*

○ Open Access

Open for fishing and entering

○ No Take

Total extraction ban

○ Partial Restrictions

e.g. periodic closures, size limits, gear restrictions, species restrictions

□ Periodic Closure

The area is open and closed as a fisheries management strategy, e.g., rotating octopus closures

□ Size Limits

Restrictions on the size of certain target species

□ Gear Restriction

Restrictions on what types of fishing gear can be used

□ Species Restriction

Restrictions on what types of species can be caught

□ Access Restriction

Access is restricted, e.g., people outside a community are not allowed to fish here

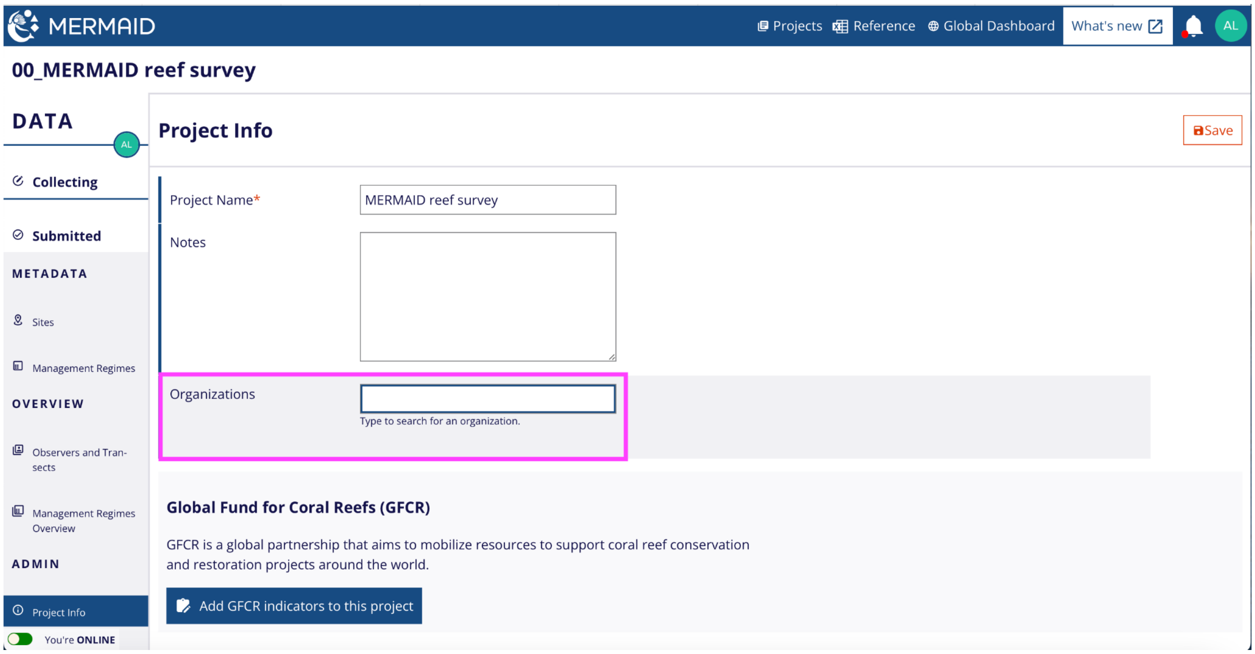
Figure 9. Acceptable options for rules of management regimes

Once all these steps have been completed, you can proceed with the upload process to upload your Excel spreadsheet to your project in MERMAID.

4. Add organizations

Including the organization(s) associated with your project is crucial for proper tagging in the MERMAID dashboard. By specifying these affiliations, you enable more efficient filtering and searching of project data by organization name. To add organizations (Figure 4):

a. Click on **Project Info** under **Admin** in the left side menu.
b. Add your organization(s) name in the corresponding field **Organizations**.



00_MERMAID reef survey

Project Info

Project Name* MERMAID reef survey

Notes

Organizations

Type to search for an organization.

Global Fund for Coral Reefs (GFCR)

GFCR is a global partnership that aims to mobilize resources to support coral reef conservation and restoration projects around the world.

Figure 4. Adding organizations to a project in MERMAID

5. Add users and roles

Add users to each project using their email address that is associated with a MERMAID account and select their respective roles with permissions. If adding users who do not yet have a MERMAID account, note that they will only be able to access the project after creating an account and verifying their email address. To add users (Figure 5):

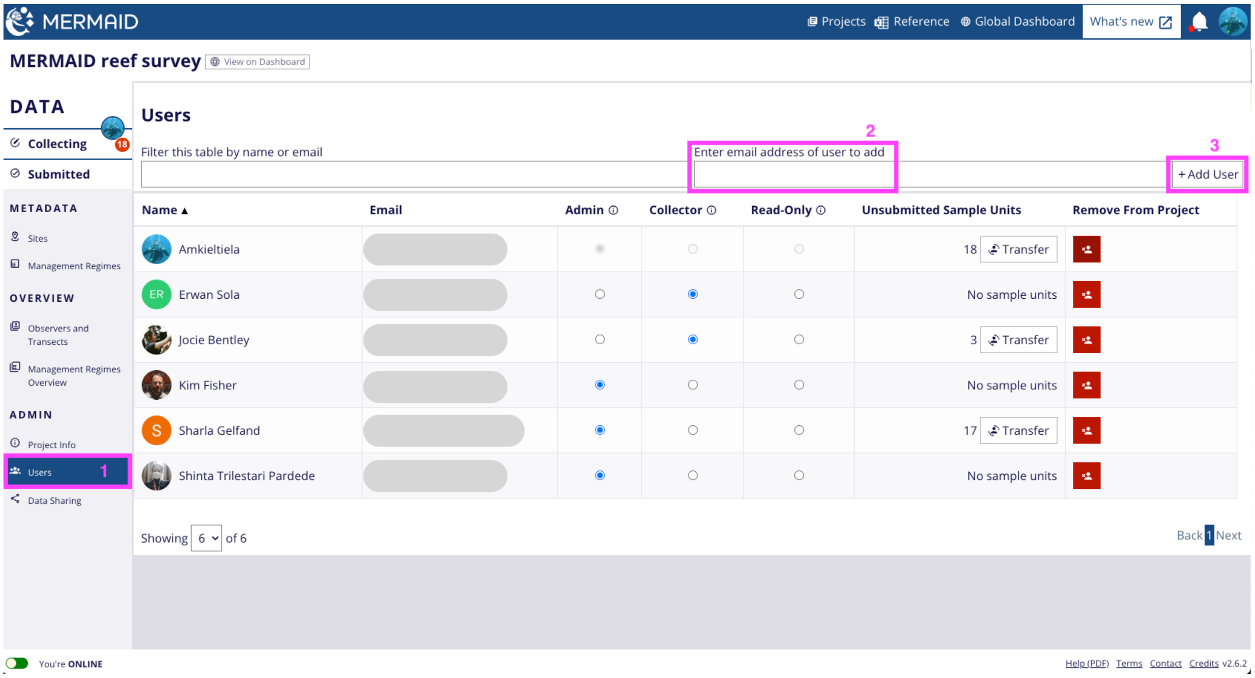
a. Click on **Users** under **Admin** in the left side menu.
b. Use the field **Enter email address of user to add** then click **+Add user**.
c. Choose each user's roles from the following options:

i. **Admin** has full control over the project. They can modify project information and data sharing policies, add or remove users and adjust their permissions, delete projects, transfer unsubmitted sample units between users, and un-submit sample units for further editing. Additionally, they have the capability to view, download, and analyze data, as well as collect new observations as sample units.

ii. **Collector** can view, download, and analyze data and collect new observations. Once a sample unit is submitted, a collector cannot edit or delete their observations.

iii. **Read-Only** is limited to viewing, downloading, and analyzing data. This user does not have the permissions to collect new observations.

Note: The default mode for user permissions is **Collector.**



Name ▴	Email	Admin ⓘ	Collector ⓘ	Read-Only ⓘ	Unsubmitted Sample Units	Remove From Project
Amkieltiela	[redacted]	●	○	○	18 Transfer	
Erwan Sola	[redacted]	○	●	○	No sample units	
Jocie Bentley	[redacted]	○	●	○	3 Transfer	
Kim Fisher	[redacted]	●	○	○	No sample units	
Sharla Gelfand	[redacted]	●	○	○	17 Transfer	
Shinta Trilestari Pardede	[redacted]	●	○	○	No sample units	

Figure 5. Adding users and choosing their roles in a project in MERMAID

6. Set data sharing policies

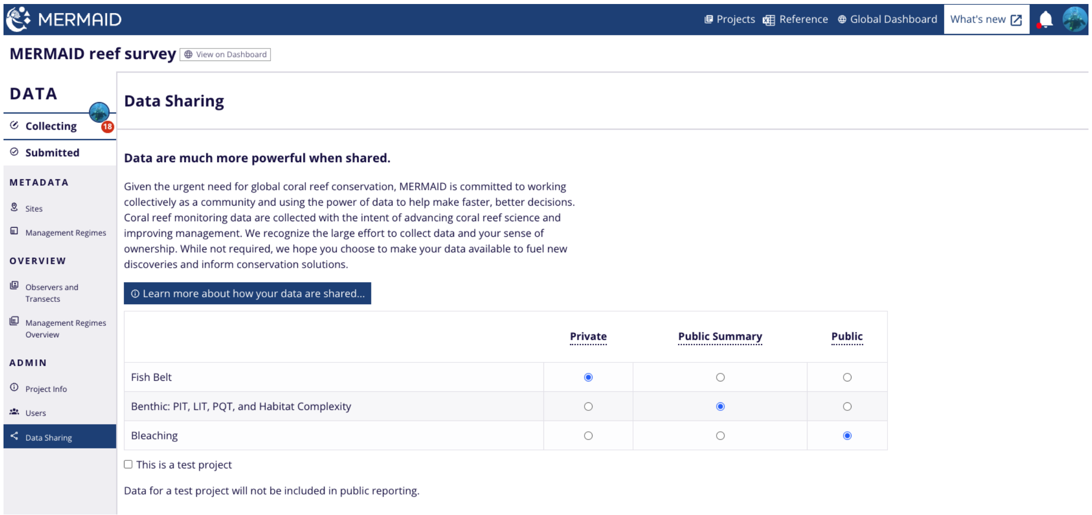
Discuss among the project admins to determine the appropriate **Data Sharing** options for your project. Select the settings that best align with your project's needs and data protection requirements.

If you are uploading data on behalf of a program or organization, confirm the appropriate privacy settings with the designated contact person before proceeding with the upload.

To set data sharing policies (Figure 6):

a. Click on **Data Sharing** under **Admin** in the left side menu
b. Choose data sharing policy for each survey method (**Fish Belt**, **Benthic**, **Bleaching**) from the following options:
    i. **Private:**
    ii. **Public Summary:**
    iii. **Public**

    Note: The default mode is **Public Summary.**



Data Sharing

Data are much more powerful when shared.

Given the urgent need for global coral reef conservation, MERMAID is committed to working collectively as a community and using the power of data to help make faster, better decisions. Coral reef monitoring data are collected with the intent of advancing coral reef science and improving management. We recognize the large effort to collect data and your sense of ownership. While not required, we hope you choose to make your data available to fuel new discoveries and inform conservation solutions.

Learn more about how your data are shared...

	Private	Public Summary	Public
Fish Belt	●	○	○
Benthic: PIT, LIT, PQT, and Habitat Complexity	○	●	○
Bleaching	○	○	●

☐ This is a test project

Data for a test project will not be included in public reporting.

Figure 6. Setting up data sharing for each survey method in a project in MERMAID

Upload data to MERMAID using R

After creating a project in MERMAID, you can start uploading your data from Excel spreadsheets to MERMAID using R and the R package mermaidr. Following the steps:

1. **Download the MERMAID template.**
2. **Load your data.**
3. **Adjust your data format to match the template.**
4. **Check that all columns match the accepted options.**
5. **Upload to MERMAID.**

1. Download the MERMAID template

Download your MERMAID template using the R package **mermaidr** after creating a new project in MERMAID.

a. Access your MERMAID project by using `mermaid_search_my_projects(“Project name”)`. For example:

```
library(mermaidr)
library(tidyverse)

reef_survey <- mermaid_search_my_projects("MERMAID reef survey")
```

The `“MERMAID reef survey”` is the name of the example project. Change it based on your project name.

b. Download the MERMAID template and options using `mermaid_import_get_template_and_options(project name, “method”, “name the template file.xlsx”)`. For example:

```
fish_template_and_options <- mermaid_import_get_template_and_options(
  reef_survey,
  "fishbelt",
  "fishbelt_mermaid_template.xlsx"
)
```

Change the method based on the data that you are uploading. The options are `fishbelt`, `benthicpit`, `benthiclit`, `benthicpqt`, `bleaching`, and `habitatcomplexity`.

2. Load your data

In this example, we are using a CSV file format. If your Excel file is in a different format, convert it to CSV by selecting the **Save As** option, choosing **.csv** from the file format options, and then clicking **Save**. To load your data into R, use the `read_csv` function.

3. Adjust your data format to match the template

To ensure your data is compatible with the MERMAID template, follow these steps:

a. Select relevant columns. Use the `select` function to include columns that align with those required by the MERMAID template. At this step, column names may not match exactly but should contain the correct information. Renaming columns to match the template will be done later.

b. Clean your data by renaming columns with complicated names to match the MERMAID template, renaming duplicate sites that are spelt differently, matching the data with the acceptable options from the MERMAID template, etc.

c. Rename columns to exactly match those in the MERMAID template. Ensure that all columns marked with an asterisk (*) are present and that all names match precisely. Refer to Annex 1 for the complete list of column names for each method.

d. Other potential issues and solutions:

i. General issues

- If required columns are absent, use the `add_column` function to add them
- If dates are in a single column, use the `separate` function to split them into year, month, and day columns.
- Create a site list to manually enter into your project in MERMAID.

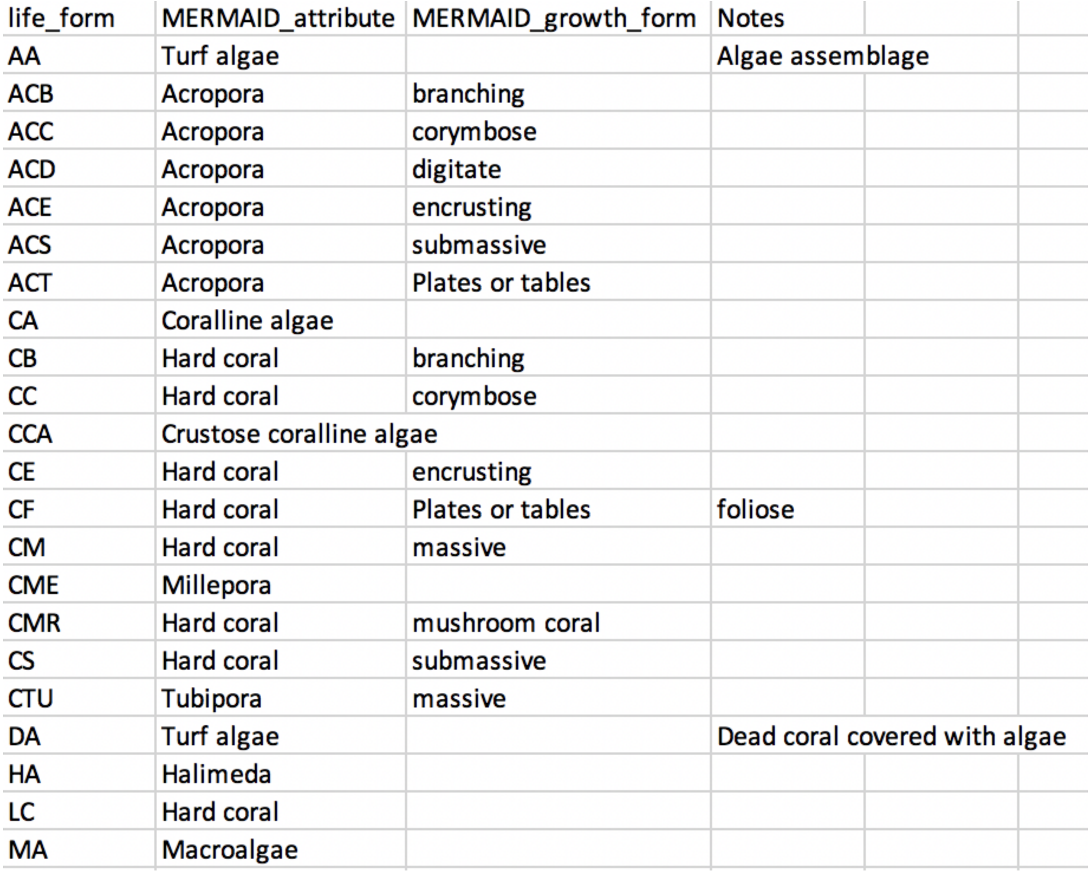
ii. Fish Belt issues

- If fish size bins are in multiple columns, use `pivot_longer` function to consolidate size bins into a single column with another column for corresponding values.
- If fish size is grouped into intervals like 3_10, 10_20, 20_30, etc., use the `mutate` function to convert these to midpoints (e.g. 5, 15, 25).

iii. Benthic PIT and LIT issues

- If benthic attributes are in multiple columns, use `pivot_longer` function to combine benthic attributes into a single column.
- For PIT, if benthic attribute counts are in a single column rather than separate rows for each point intercept, use the `uncount` function to ensure each point intercept is represented as a separate row.
- If you use different codes/names for benthic attributes to those in MERMAID, convert your codes/names to MERMAID's accepted benthic attributes and growth forms. Compile an Excel spreadsheet to map your codes to MERMAID's attributes/growth forms (Table 1). Growth forms accepted in MERMAID are arborescent, branching, columnar, corymbose, digitate, encrusting, foliose, massive, mushroom coral. plates or tables, submassive.

Table 1. An example of how WCS Fiji data codes align with MERMAID attributes (life_form = Fiji's benthic codes)



life_form	MERMAID_attribute	MERMAID_growth_form	Notes
AA	Turf algae		Algae assemblage
ACB	Acropora	branching	
ACC	Acropora	corymbose	
ACD	Acropora	digitate	
ACE	Acropora	encrusting	
ACS	Acropora	submassive	
ACT	Acropora	Plates or tables	
CA	Coralline algae		
CB	Hard coral	branching	
CC	Hard coral	corymbose	
CCA	Crustose coralline algae		
CE	Hard coral	encrusting	
CF	Hard coral	Plates or tables	foliose
CM	Hard coral	massive	
CME	Millepora		
CMR	Hard coral	mushroom coral	
CS	Hard coral	submassive	
CTU	Tubipora	massive	
DA	Turf algae		Dead coral covered with algae
HA	Halimeda		
LC	Hard coral		
MA	Macroalgae		

iv. Bleaching issues

- Ensure quadrat numbers are unique (avoiding duplicates), especially when combining data from multiple surveys with both same time and site
- If some % cover columns are missing data (e.g., only hard coral and no soft coral or macroalgae), remove rows with incomplete data. % cover columns cannot be left blank.

v. Benthic PQT issues

- Add missing transect and quadrat numbers, as these are common omissions in data formats. Ensure these numbers are included before uploading data to MERMAID.
- Sum duplicated benthic attributes within the same transect and quadrat number. For example, if the same benthic attribute is recorded at multiple points, consolidate these into a single entry with the total count. Benthic attributes need to be unique in the same transect number and quadrat number (e.g., you have

Acropora branching in point 1 and 5 of 10 points per quadrat, then sum Acropora branching to a total of 2).

4. Check that all columns match the accepted options

Compare all column names and their options against the list of MERMAID's accepted options from the template. Each column needs to be checked individually using the `mermaid_import_check_options()` function. Note that each column name must exactly match the names specified in the template. You can check the template column names in using the following syntax:

```
names(fishbelt_template_and_options[[“Template”]])
```

After all column names match the columns in the MERMAID template, verify each column in your dataset using the `mermaid_import_check_options(your data, MERMAID template, “column name”)`. For example, to check the `Site` column, use:

```
mermaid_import_check_options(fishbelt_data,
fishbelt_template_and_options, “Site”)
```

5. Upload to MERMAID

After ensuring all column names match those in the MERMAID template, and the data in each column is correct, you can proceed to upload your data to MERMAID using the R package **mermaidr**. Ensure you have admin rights for the project before importing data. The uploaded data will appear on the **Collecting** page of your project in MERMAID and you will need to validate and submit each sample unit.

Follow these steps to upload your data to MERMAID:

1. Conduct a dry run using `mermaid_import_project_data()` function and set `dryrun` to `TRUE` to check for potential issues without actually uploading the data. For example:

   ```
   mermaid_import_project_data(
   fishbelt_data,
   reef_survey,
   method = “fishbelt”,
   dryrun = TRUE)
   ```

Note: `fishbelt_data` is the data you've prepared to be uploaded to MERMAID, `reef_survey` is the project where your data will be uploaded to (see the preparing data section), and `method` is the method for how your data was collected (choose between `fishbelt`, `benthicpit`, `benthiclit`, `benthicpqt`, `bleaching`, or `habitatcomplexity`).

2. After receiving the message, `Records successfully checked!`, change the `dryrun` option to `FALSE` to start uploading your data to MERMAID. For example:

```
mermaid_import_project_data(
Fishbelt_data,
Reef_survey,
method = “fishbelt”,
dryrun = FALSE)
```

After receiving the message, `Records successfully imported!`, go to the **Collecting** page in your project in the MERMAID app to validate and submit each sample unit.

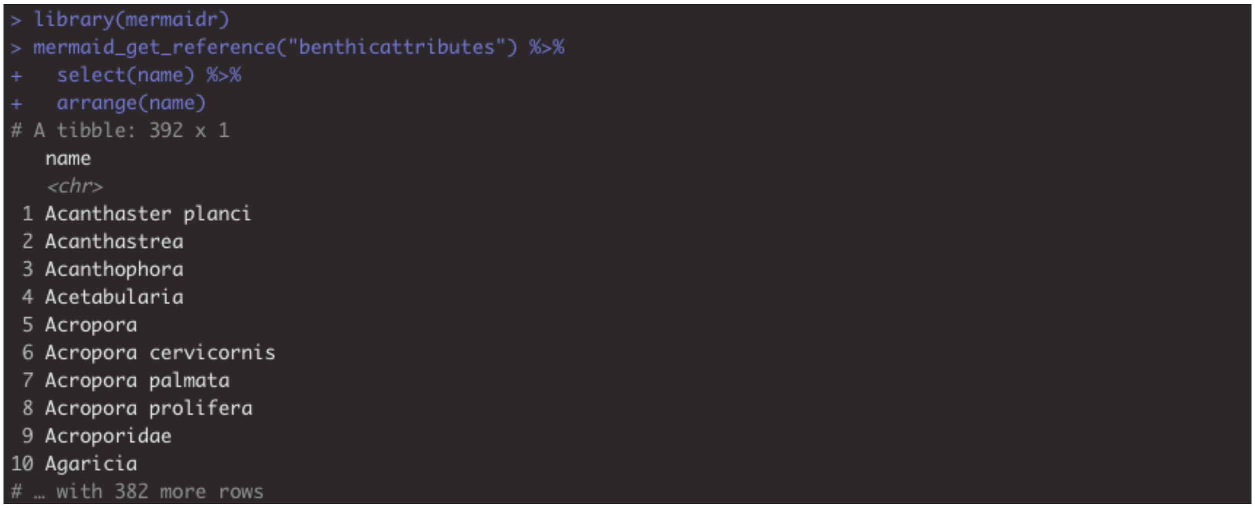
Note: If you do not see your data in the **Collecting** page, refresh the page, log out and log in again, or contact our support at contact@datamermaid.org.

Troubleshooting and data cleaning your Excel files

1. Re-coding data in R

a. If your data includes all the necessary columns listed above (for either the fish belt, benthic PIT, benthic LIT, benthic PQT, bleaching, or habitat complexity) in the required format with MERMAID acceptable options, proceed to the 'Troubleshooting' section below to ensure your datasheet is ready for upload.

To find the list of 'acceptable MERMAID options' use the `mermaid_get_reference()` function from the **mermaidr** package. This function gives you the most up-to-date list of attributes from the MERMAID API.



```
> library(mermaidr)
> mermaid_get_reference("benthicattributes") %>%
+   select(name) %>%
+   arrange(name)
# A tibble: 392 x 1
   name
   <chr>
 1 Acanthaster planci
 2 Acanthastrea
 3 Acanthophora
 4 Acetabularia
 5 Acropora
 6 Acropora cervicornis
 7 Acropora palmata
 8 Acropora prolifera
 9 Acroporidae
10 Agaricia
# … with 382 more rows
```

a. If your data is not in the correct format (excluding the date and time columns, which must be correctly formatted to be read into R), you can easily and consistently reformat it in R. See the section below for common data cleaning steps.

2. Troubleshooting in R

a. Benthic PIT: ensuring correct number of entries per transect
   - Recall from above: Transect length, Interval size, Interval start, Observation Interval need to be consistent - e.g., if transect length is 20 m, the interval size is 0.5 m and interval start is 0.5 m, then there should be 40 data points per transect, with observation intervals column such as 0.5, 1, 1.5, ...,19.5, 20.
   - To check for any deviations in the expected number of observations per transect:

```
final %>%
  count(site,
    management,
    sample_date_year,
    sample_date_month,
    sample_date_day,
    sample_time,
    depth,
    tide,
    transect_number,
    relative_depth,
    observer_emails) %>%
  filter(n != 100) %>%
  view()
```

b. Ensuring no missing or duplicate transects per site
    - Verify that the expected number of transects is present for each site/year/relative depth

```
final %>%
  group_by(site, sample_date_year, relative_depth, sample_time, transect_number) %>%
  summarize(n())
#could also add observer_emails and/or day/month inside the “group_by” syntax if necessary
```

c. Verifying the number of Collect records against those uploaded to MERMAID)
    - Verify how many transects should be expected based on how MERMAID separates transects vs. how many are actually uploaded

```
#find n=575 collect records - yup
final %>%
  distinct(site,
           management,
           depth,
           sample_date_year,
           sample_date_month,
           sample_date_day,
           depth,
           transect_number,
           observer_emails) %>%
  nrow()
```

3. Additional notes and troubleshooting tips

It is crucial to review all the columns in your dataset to confirm that they are properly formatted and do not contain any unexpected values. This is particularly important for critical fields such as `site`, `management regime`, and `observer emails`. These should align with the entries in MERMAID and be consistent with the project specifications.

To efficiently check the contents of a column, use the `tabyl` function from the dplyr package, which provides a summary table for the column:

```
fishbelt_data %>%
  tabyl(`site *`)
```

Occasionally, you may need to recode values in a column to align with MERMAID's accepted formats. For instance, the `tides` column might contain various terms that need to be standardized. In the dataset, `full` might need to be recorded as `high` according to MERMAID's requirements. To ensure all variations are captured, it is best to convert all entries to lowercase, as MERMAID is case-sensitive. Ensures that the `fct_recode` command catches all the `full` entries in the dataset, even those that might have had a different capitalization initially.

```
fishbelt_data <- fishbelt_data %>%
  mutate(Tide = str_to_lower(tide),
         Tide = fct_recode(tide,
                           "high" = "full"))
```

What if months were written out and not written as numbers?

If the dataset uses month names instead of numbers, use `case_when()` to convert them. This function is useful as it retains the column as a non-factor variable, which can be beneficial for certain analyses:

```
#need to change from 'name of month' -> month #
fishbelt_data <- fishbelt_data %>%
  mutate(`Sampel date: Month *` = case_when(
    (`Sampel date: Month *` == "January") ~ 1,
    (`Sampel date: Month *` == "February") ~ 2,
    (`Sampel date: Month *` == "March") ~ 3,
    (`Sampel date: Month *` == "April") ~ 4,
    (`Sampel date: Month *` == "May") ~ 5,
    (`Sampel date: Month *` == "June") ~ 6,
    (`Sampel date: Month *` == "July") ~ 7,
    (`Sampel date: Month *` == "August") ~ 8,
    (`Sampel date: Month *` == "September") ~ 9,
    (`Sampel date: Month *` == "October") ~ 10,
    (`Sampel date: Month *` == "November") ~ 11,
    (`Sampel date: Month *` == "December") ~ 12))
```

Tips for formatting specific columns:

Time column:

When importing data from Excel, R might include an unintended date component in time-formatted columns (e.g., "1899-12-31 02:15:00"). This must be removed for proper data entry:

```
#converting the 'time' column into a hh:mm:ss format
fishbelt_data <- fishbelt_data %>%
    mutate(`Sample time` = format.POSIXct(time,format = "%H:%M:%S"))
```

Additionally, MERMAID interprets any `<NA>` values in the time column as midnight, which is not typically when fish are surveyed. It is advisable to update these and any other night-time entries to their appropriate daytime equivalents to ensure accurate data representation.

```
#change 2:22:00 and 2:15:00 to their during the day counterparts as no one is surveying fish in the middle of the night
fishbelt_data <- fishbelt_data %>%
  mutate(`Sample time` = case_when(
    is.na(`Sample time`) ~ "12:00:00",
    TRUE ~ `Sample time`)) %>%
  mutate(`Sample time` = fct_recode(`Sample time`,
                          "14:22:00" = "02:22:00",
                          "14:15:00" = "02:15:00")) %>%
  mutate(`Sample time` = as.character(`Sample time`))
```

Fish sizes:

In some datasets, like the Fiji data, the largest size class is '>40', while MERMAID requires a '>50' classification. To successfully upload the data to MERMAID, these size classes need to be converted to match MERMAID's accepted options. Adjust the column name based on the name of the columns in your data to use the code.

```
#size has to be the mid-size of the bin in the fiji data unless it is the '>40' bin in which case, if it is between 40-45 -> 42.5, 45-50 -> 47.5, 45 -> 42.5 and if it is >50 => actual size
#check what size_classes there are
kubulau_fish_template %>%
  tabyl(size_class)
#some seem to have been turned into nonsense - e.g. "42309","42492","42649" (i think they were turned into dates by excel), but there are no NAs [if there are, need to add sizes]
kubulau_fish_template %>%
  filter(size_class == 42649) %>% #checked each one of these independently
  tabyl(size_fiji)
#42309 - all are 13cm, 42492 - 3cm,3.5cm, 42649 - 8cm

#making an appropriate size column
#size_fiji <- turned the original 'size' column into 'size_fiji' - now making a new 'size' column to be imported
kubulau_fish_template <- kubulau_fish_template %>%
  mutate(size = case_when(
    (size_class == "11-15cm" | size_class == 42309) ~ 12.5,
    (size_class == "16 - 20" | size_class == "16-20" | size_class == "16-20cm") ~ 17.5,
    (size_class == "2-5cm" | size_class == 42492) ~ 2.5,
    (size_class == "6-10cm" | size_class == 42649) ~ 7.5,
    (size_class == "21 - 25" | size_class == "21-25" | size_class == "21-25cm") ~ 22.5,
    (size_class == "26 - 30" | size_class == "26-30" | size_class == "26-30cm") ~ 27.5,
    (size_class == "31 - 35" | size_class == "31-35" | size_class == "31-35cm") ~ 32.5,
    (size_class == "36 - 40" | size_class == "36-40" | size_class == "36-40cm") ~ 37.5,
    ((size_class == ">40" | size_class == ">40cm") & size_fiji == 40) ~ 37.5, #underestimate!
    ((size_class == ">40" | size_class == ">40cm") & size_fiji == 50) ~ 47.5, #underestimate!
    ((size_class == ">40" | size_class == ">40cm") & size_fiji > 50) ~ size_fiji,
    ((size_class == ">40" | size_class == ">40cm") & size_fiji > 40 & size_fiji < 45) ~ 42.5,
    ((size_class == ">40" | size_class == ">40cm") & size_fiji == 45) ~ 42.5,
    ((size_class == ">40" | size_class == ">40cm") & size_fiji > 45 & size_fiji <= 50) ~ 47.5,
    TRUE ~ -10)) #this last case should never happen

kubulau_fish_template %>%
  tabyl(size)

#double checking the >40s make sense - yes they're fine
kubulau_fish_template %>%
  #filter(size_class == ">40") %>%
  filter(size_class == ">40cm") %>%
  group_by(size,size_fiji) %>%
  summarize(n())
```

Fish names:

MERMAID accepts fish names at the species, genus, and family levels. For instance, names like 'Genus sp.' are not accepted. Instead, MERMAID requires genus-level fish names to be provided without the 'sp.' suffix.

```
#list of fish attributes that are valid in MERMAID
Fishbelt_template_and_options[["Fish name *"]]

#remove the sp.'s
#str_remove(colname, 'sp\\.$') OR str_replace(colname, 'sp\\.$', '')
fishbelt_data <- fishbelt_data %>%
  mutate(`fish name *` = str_replace(`fish name *`," sp\\.$", ""))

fishbelt_data %>%
  tabyl(`fish name *`)


#check if all fish names match the MERMAID database. To run this code, make sure that your data has exactly the same column names as the MERMAID template.
```

```
mermaid_import_check_options(fishbelt_data, fishbelt_template_and_options, “Fish
name *”)

# Recode the names that don't match, to match the MERMAID database
fishbelt_data <- fishbelt_data %>%
  mutate(`Fish name *` = case_when (
 `Fish name *` == “Parupeneus multifaskiatus” ~ “Parupeneus      multifasciatus”,
 `Fish name *` == “Parupeneus indikus” ~ “Parupeneus indicus”,
 `Fish name *` == “Lutjanus sp.” ~ “Lutjanus”,
 TRUE ~ `Fish name *`))
```

Benthic attributes, growth forms:

MERMAID accepts benthic attributes data in species, genus, family, and top-level benthic categories (i.e. hard coral, soft coral, etc.). To learn more about MERMAID taxonomy, go to the documentation. Ensuring your benthic attribute data matches the MERMAID database requires only two simple steps, which are:

a. Checking if all benthic attributes match those in the MERMAID database

```
#Check for unmatch benthic attributes
mermaid_import_check_options(benthicpit_data, benthicpit_template_and_options,
“Benthic attribute *”)
```

b. Renaming unmatched benthic attributes to match those in the MERMAID database

```
# Recode to match MERMAID database
benthicpit_data <- benthicpit_data %>%
     mutate(`Benthic attribute *`  = case_when(
   `Benthic attribute *` == “zooanthids” ~ “Zoanthid”,
   `Benthic attribute *` == “gorgonian fan” ~ “Gorgonian”,
   `Benthic attribute *` == “Caulastrea” ~ “Caulastraea”,
   TRUE ~ `Benthic attribute *`))
```

Annex

Annex 1. Columns in the MERMAID template for each method

Mandatory Columns	Optional Columns
All methods	
Site * - Matching exactly to a site name entered in the MERMAID project	Sample Time - HH:MM:SS, 24hr format
Management * - Matching exactly to a management regime name added to the MERMAID project	Transect label - Can be numbers or characters. An optional character field to add information like ‘deep’ or ‘shallow’ to your transects
Sample date: Year * - Needs to be numeric and must be four numbers (e.g., 2024)	Reef slope - Accepted options are flat, slope, wall, and crest
Sample date: Month * - Needs to be numeric and must be between 1-12	Visibility - Accepted options are <1m - bad; 1-5m - poor; 5-10m - fair; and >10m - excellent
Sample date: Day * - Needs to be numeric and must be between 1-31	Current - Accepted options are low/none, moderate, and high
Depth * - Needs to be in ‘m’, but the column should not include measurement unit (i.e., 7.8 not 7.8 m)	Relative depth - Accepted options are shallow and deep
Transect number * - Only numbers in this column	Tide - Accepted options are slack, low, rising, high, and falling
Transect length surveyed * - Needs to be in ‘m’, but the column should not include measurement unit	Sample unit notes - Notes recorded by observer for specific transect
Observer emails * - Add the email(s) of the observer(s). They must match the email associated with their MERMAID account, and the user must be listed on the MERMAID project	

Fishbelt specific columns	
Width * - In meters. Accepted options are: ● 1m ● 2m ● 5m ● 10m ● 20m ● Mixed: <10cm @ 2m, >= 10cm @ 5m ● Mixed: <20cm @ 2m, >= 20cm @ 4m ● Mixed: <20cm @ 4m, >= 20cm @ 8m ● Mixed: <=10 cm & <35 cm @ 5 m, >= 35 cm @ 20 m	
Fish size bin * - In centimeters. Accepted options are: ● 1 ● 5 ● 10 ● AGGRA ● WCS India	
Fish name * - has to be one of the accepted options in MERMAID. Can be species, genus or family level	
Size * - Total length of fish observed in the previous column. Use the midpoint of each size bin up until 50 cm in size. For example, enter 7.5 for fish with size 5-10 cm. For fish observations larger than 50cm, enter the size to the nearest cm (i.e., 55cm, 85 cm).	
Count * - Abundance of the fish observed with that size class	
Benthic PIT specific columns	
Interval size * - Spacing of data points (e.g., if values are recorded at 0.5 m,1 m,1.5 m, etc., interval size would be 0.5)	Growth form - Growth form of the corresponding attribute. i.e., Acropora (attribute) and branching (growth form), or Hard coral (attribute) and massive (growth form)

Interval start * - Distance of the first data point (e.g., if recorded values begin at 0.5 m, this would be 0.5)	
Observation interval * - Transect point where the corresponding benthic attribute, detailed in the next column, was observed.	
Benthic attribute * - Observed organism. MERMAID will only accept options listed in the downloaded MERMAID template	
Benthic LIT specific columns	
Benthic attribute * - Observed organism. MERMAID will only accept the options listed in the downloaded MERMAID template	Growth form - Growth form of the attribute. i.e., Acropora (attribute) and branching (growth form), or Hard coral (attribute) and massive (growth form)
Observation length * - length of a benthic attribute (/growth form) observed in cm (e.g., 14)	
Benthic PQT specific columns	
Number of quadrats * - total number of quadrats per transect as an integer (e.g., 10)	First quadrat number - Number of the first quadrat/photo along the transect as an integer (e.g., 1).
Quadrat size * - Quadrat size used per transect in square meters (e.g., 1)	Growth form - Growth form of the attribute. i.e., Acropora (attribute) and branching (growth form), or Hard coral (attribute) and massive (growth form)
Number of points per quadrat * - Total number of points per quadrat used in a transect, as an integer (e.g., 100)	
Quadrat * - Number of quadrat/photo in transect as an integer (e.g., 1)	
Benthic attribute * - Observed organism. MERMAID will only accept the options listed in the downloaded MERMAID template	

Number of points * - Number of points with unique benthic attribute (/growth form) for the quadrat	
Bleaching specific columns	
Quadrat size * - Quadrat size used in square meters (e.g. 1).	Benthic attribute - Observed organism. MERMAID will only accept the options listed in the downloaded MERMAID template
	Growth form - Growth form of the attribute. i.e., Acropora (attribute) and branching (growth form), or Hard coral (attribute) and massive (growth form)
	Number of colonies normal - Number of normal colonies as integer (e.g., 3)
	Number of colonies pale - Number of pale colonies as integer (e.g, 3)
	Number of colonies bleached 0-20% - Number of 0-20% bleached colonies as integer (e.g., 3)
	Number of colonies bleached 20-50% - Number of 20-50% bleached colonies as integer (e.g., 3)
	Number of colonies bleached 50-80% - Number of 50-80% bleached colonies as integer (e.g., 3)
	Number of colonies bleached 80-100% - Number of 80-100% bleached colonies as integer (e.g., 3)
	Number of colonies recently dead - Number of recently dead colonies as integer (e.g., 3)
	Quadrat number - Number of quadrats in sample unit collection as integer (e.g., 1).
	Hard coral % cover - Hard coral cover as

	decimal percentage of quadrat total area (e.g., 33.3)
	Soft coral % cover - Soft coral cover as decimal percentage of quadrat total area (e.g., 33.3)
	Macroalgae % cover - Macroalgae cover as decimal percentage of quadrat total area (e.g., 33.3)
Habitat complexity specific columns	
Interval size * - Distance between observations on a transect, in meters. May include decimal (e.g., 0.5)	
Observation interval * - Interval of observation along the transect in meters as a multiple of interval size. May include decimal (e.g., 3.5).	
Habitat complexity score * - Benthic complexity score (0 - 5) for transect interval as integer (e.g., 3). The categories are 0 no vertical reef, flat or rubbly areas; 1 low (<30 cm high) and sparse relief; 2 low but widespread relief; 3 widespread moderately complex (30-60 cm high) relief; 4 widespread very complex (60 -100 cm high) relief with numerous fissures and caves; 5 exceptionally complex (>1 m high) relief with numerous caves and overhangs). See relevant tab on ingestion template for choices.	

Annex 2. Example of uploading data from Excel spreadsheet to MERMAID

Here we provide a step-by-step example of how to upload your existing fish data (using the fish belt method) to MERMAID using the R package **mermaidr**.

Before beginning, ensure your project is set up in the MERMAID app; instructions for project setup can be found here. In this example, we have a project named “MERMAID reef survey”. Additionally, a walkthrough video is available to guide you through this example. You can also download the fishbelt and sites data for this example.

Steps for uploading pre-existing fish belt data:

1. **Download the fish belt MERMAID template**
2. **Reformat your data to match the template**
3. **Address errors and warnings**
4. **Upload your data to MERMAID**

1. Download the fish belt MERMAID template

Before downloading the MERMAID template, ensure the **mermaidr** package is installed from Github:

```
remotes::install_github("data-mermaid/mermaidr")
```

Load the necessary packages and authenticate your MERMAID account to gain access to your projects in MERMAID using the `mermaid_search_my_projects()` function with the project name:

```
library(mermaidr)
library(tidyverse)

reef_survey <- mermaid_search_my_projects("MERMAID reef survey")
```

A browser window will open for you to authenticate by logging into the MERMAID app. Once you’ve logged in, you can close the browser and come back to R. Your login credentials will last for a day and after they expire you will need to log in again.

Download the fish belt MERMAID template and options using `mermaid_import_get_template_and_options()` and save it to a file called fishbelt_mermaid_template.xlsx:

```
fish_template_and_options <- mermaid_import_get_template_and_options(
  reef_survey,
  "fishbelt",
  "fishbelt_mermaid_template.xlsx"
)
```

The XLSX file consists of the fish belt MERMAID template and the options that are tailored based on your project. For example, the site options that are accepted are all the sites that you have added to your project in MERMAID.

You can preview the template in R:

```
fish_template_and_options[["Template"]]
```

```
## # A tibble: 0 × 22
## # … with 22 variables: Site * <chr>, Management * <chr>, Sample date: Year * <chr>,
## #   Sample date: Month * <chr>, Sample date: Day * <chr>, Sample time <chr>,
## #   Depth * <chr>, Transect number * <chr>, Transect label <chr>,
## #   Transect length surveyed * <chr>, Width * <chr>, Fish size bin * <chr>,
## #   Reef slope <chr>, Visibility <chr>, Current <chr>, Relative depth <chr>, Tide <chr>,
## #   Sample unit notes <chr>, Observer emails * <chr>, Fish name * <chr>, Size * <chr>,
## #   Count * <chr>
```

Or if you want to investigate the column names in R:

```
names(fish_template_and_options)
```

```
##  [1] "Template"                   "Site *"                     "Management *"
##  [4] "Sample date: Year *"        "Sample date: Month *"       "Sample date: Day *"
##  [7] "Sample time"                "Depth *"                    "Transect number *"
## [10] "Transect label"             "Transect length surveyed *" "Width *"
## [13] "Fish size bin *"            "Reef slope"                 "Visibility"
## [16] "Current"                    "Relative depth"             "Tide"
## [19] "Sample unit notes"          "Observer emails *"          "Fish name *"
## [22] "Size *"                     "Count *"
```

All column names with an asterisk are mandatory. Columns without an asterisk are optional and can be left blank, if data is not available.

Investigate all the available options for each column before adjusting the format. For example, to look at the options for site in R:

```
fish_template_and_options[["Site *"]]
```

```
## $required
## [1] TRUE
##
## $help_text
## [1] "A unique name of a site where data was collected. Every site must be defined before inges
tion and set up in the project in the web app."
##
## $choices
## # A tibble: 5 × 1
##   value
##   <chr>
## 1 1206
## 2 1208
## 3 1209
## 4 1210
## 5 1211
```

2. Reformat your data to match the template

Prepare your fish belt data according to the MERMAID template. In this example, we are using a fish belt width of 5 m for fish size 10-34 cm, and 20 m for fish size bigger than 34 cm. We also identify the fish size up to the closest cm.

You should prepare your data as two separate CSV files (fish belt data stored in **fishbelt.csv** file and sampling event data stored in **sites.csv** file) and stored it in your project directory.

Read your data files into R and save the two data files as two different objects:

```
fishbelt_data <- read_csv("fishbelt.csv")
```

```
sites_data <- read_csv("sites.csv")
```

The MERMAID upload template requires the sample event data and the observation data to be combined. Look at the available columns in the `site` and `fishbelt` datasets:

```
sites_data
```

```
## # A tibble: 2 × 8
##   SiteID Zone     Year Month   Day Reef_slope visibility current
##    <dbl> <chr>   <dbl> <dbl> <dbl> <chr>           <dbl> <chr>
## 1   1208 Control  2021     3    15 slope              15 low
## 2   1206 Control  2021     3    15 slope              15 low
```

```
fishbelt_data
```

```
## # A tibble: 10 × 8
##    SiteID Depth Transect_length Width Transect_number `Fish species`      Size_cm Abundance
##     <dbl> <dbl>           <dbl> <dbl>           <dbl> <chr>                <dbl>     <dbl>
##  1   1208     5               5     5               1 Parupeneus multifa…      15         5
##  2   1208     5               5    20               1 Acanthurus bariene       37         1
##  3   1208     5               5    20               1 Platax teira             37         1
##  4   1208     5               5    20               1 Bolbometopon muric…     120         1
##  5   1208     5               5    20               1 Parupeneus indikus       36         1
##  6   1208     5               5     5               1 Acanthurus auranti…      17         4
##  7   1208     5               5    20               1 Plectorhinchus cha…      36         4
##  8   1208     5               5     5               1 Zanclus cornutus         10         2
##  9   1208     5               5    20               1 Variola albimargin…      36         1
## 10   1208     5               5    20               1 Lutjanus sp.             36         3
```

Combine site data with fishbelt data using the SiteID identifier:

```
fishbelt_data <- fishbelt_data %>%
  left_join(sites_data, by = "SiteID")
```

Add missing mandatory fields and adjust existing ones. In this example, there are two mandatory fields that have not yet been added (Fish size bin and Observer emails) and one mandatory field that needs to be adjusted (Fish Belt Width). You need to look at the options available for these columns:

```
fish_template_and_options[["Width *"]]
```

```
## $required
## [1] TRUE
##
## $help_text
## [1] "Width of fish belt transect, in meters. See relevant tab on ingestion template for choice
s."
##
## $choices
## # A tibble: 6 × 1
##   value
##   <chr>
## 1 10m
## 2 20m
## 3 2m
## 4 5m
## 5 Mixed: < 10cm @ 2m, >= 10cm @ 5m
## 6 Mixed: >10 cm & <35 cm @ 5 m, >=35 cm @ 20 m
```

```
fish_template_and_options[["Fish size bin *"]]
```

```
## $required
## [1] TRUE
##
## $help_text
## [1] "Name of bin scheme used to estimate fish size for the transect. See relevant tab on inges
tion template for choices. Choose 1 cm if the fish size recorded does not use bins."
##
## $choices
## # A tibble: 5 × 1
##   value
##   <chr>
## 1 1
## 2 10
## 3 5
## 4 AGRRA
## 5 WCS India
```

```
fish_template_and_options[["Observer emails *"]]
```

```
## $required
## [1] TRUE
##
## $help_text
## [1] "Comma-separated list of emails of sample unit observers (e.g. 'me@example.com,you@exampl
e.com')."
##
## $choices
## # A tibble: 1 × 1
##   value
##   <chr>
## 1 email@mermaid.org
```

Now that you know the available options, you must manually add them:

```r
fishbelt_data <- fishbelt_data %>%
  mutate(
    `Width *` = "Mixed: >10 cm & <35 cm @ 5 m, >=35 cm @ 20 m",
    `Fish size bin *` = 1,
    `Observer emails *` = "email@mermaid.org"
  )
```

Visibility data are not the same as the options accepted by MERMAID:

```r
fishbelt_data %>%
  distinct(visibility)
```

```
## # A tibble: 1 × 1
##   visibility
##        <dbl>
## 1         15
```

Check how data is formatted in the template and implement the options into your data:

```r
fish_template_and_options[["Visibility"]][["choices"]]
```

```
## # A tibble: 4 × 1
##   value
##   <chr>
## 1 <1m - bad
## 2 1-5m - poor
## 3 5-10m - fair
## 4 >10m - excellent
```

```r
fishbelt_data <- fishbelt_data %>%
  mutate(visibility = case_when(
    visibility == 1 ~ "<1m - bad",
    visibility == 5 ~ "1-5m - poor",
    visibility > 5 & visibility <= 10 ~ "5-10m",
    visibility >= 10 ~ ">10m - excellent"
  ))
```

You may have noticed that the column names in your dataset differ from those in the template. It is crucial that the column names in your dataset exactly match those specified in the template, even though the order of the columns can vary. To ensure compatibility, follow these steps to rename and reorder the columns:

```
names(fish_template_and_options[["Template"]])
```

```
##  [1] "Site *"                     "Management *"               "Sample date: Year *"
##  [4] "Sample date: Month *"       "Sample date: Day *"         "Sample time"
##  [7] "Depth *"                    "Transect number *"          "Transect label"
## [10] "Transect length surveyed *" "Width *"                    "Fish size bin *"
## [13] "Reef slope"                 "Visibility"                 "Current"
## [16] "Relative depth"             "Tide"                       "Sample unit notes"
## [19] "Observer emails *"          "Fish name *"                "Size *"
## [22] "Count *"
```

```
names(fishbelt_data)
```

```
##  [1] "SiteID"            "Depth"             "Transect_length"   "Width"
##  [5] "Transect_number"   "Fish species"      "Size_cm"           "Abundance"
##  [9] "Zone"              "Year"              "Month"             "Day"
## [13] "Reef_slope"        "visibility"        "current"           "Width *"
## [17] "Fish size bin *"   "Observer emails *"
```

You are not required to provide information for the fields **Sample time**, **Transect label**, **Relative depth**, **Tide**, and **Sample unit notes**, as these are optional in MERMAID. Omitting these fields will not impede the data upload process. However, if this information is available, we strongly recommend including it for comprehensive data analysis.

Proceed to rename and reorder the columns in your dataset to match the format specified in the template.

```
fishbelt_data <- fishbelt_data %>%
  select(
    `Site *` = SiteID,
    `Management *` = Zone,
    `Sample date: Year *` = Year,
    `Sample date: Month *` = Month,
    `Sample date: Day *` = Day,
    `Depth *` = Depth,
    `Transect number *` = Transect_number,
    `Transect length surveyed *` = Transect_length,
    `Width *`,
    `Fish size bin *`,
    `Reef slope` = Reef_slope,
    `Visibility` = visibility,
    `Current` = current,
    `Observer emails *`,
    `Fish name *` = `Fish species`,
    `Size *` = Size_cm,
    `Count *` = Abundance
  )

fishbelt_data
```

3. Address errors and warnings

After reformatting your data, you will clean it using `mermaid_import_check_options()`.

This involves verifying each column to ensure that the data conforms to the requirements of the MERMAID template.

The following code demonstrates how to check if your reformatted data aligns with the fish belt template for MERMAID. If the data is correct, a check mark will appear:

```
mermaid_import_check_options(fishbelt_data, fish_template_and_options, "Site *")
```

```
## ✔ All values of `Site *` match
```

```
## # A tibble: 1 × 3
##   data_value closest_choice match
##   <chr>      <chr>          <lgl>
## 1 1208       1208           TRUE
```

```
mermaid_import_check_options(fishbelt_data, fish_template_and_options, "Management *")
```

```
## ✔ All values of `Management *` match
```

```
mermaid_import_check_options(fishbelt_data, fish_template_and_options, "Sample date: Year *")
```

```
## ✔ Any value is allowed for `Sample date: Year *` – no checking to be done
```

```
mermaid_import_check_options(fishbelt_data, fish_template_and_options, "Sample date: Month *")
```

```
## ✔ Any value is allowed for `Sample date: Month *` – no checking to be done
```

```
mermaid_import_check_options(fishbelt_data, fish_template_and_options, "Sample date: Day *")
```

```
## ✔ Any value is allowed for `Sample date: Day *` – no checking to be done
```

```
mermaid_import_check_options(fishbelt_data, fish_template_and_options, "Depth *")
```

```
## ✔ Any value is allowed for `Depth *` – no checking to be done
```

```
mermaid_import_check_options(fishbelt_data, fish_template_and_options, "Transect number *")
```

```
## ✔ Any value is allowed for `Transect number *` – no checking to be done
```

```
mermaid_import_check_options(fishbelt_data, fish_template_and_options, "Transect length surveyed
*")
```

```
## ✔ Any value is allowed for `Transect length surveyed *` – no checking to be done
```

```
mermaid_import_check_options(fishbelt_data, fish_template_and_options, "Width *")
```

```
## ✔ All values of `Width *` match
```

```
mermaid_import_check_options(fishbelt_data, fish_template_and_options, "Fish size bin *")
```

```
## ✔ All values of `Fish size bin *` match
```

```
mermaid_import_check_options(fishbelt_data, fish_template_and_options, "Reef slope")
```

```
## ✔ All values of `Reef slope` match
```

```
mermaid_import_check_options(fishbelt_data, fish_template_and_options, "Visibility")
```

```
## ✔ All values of `Visibility` match
```

```
mermaid_import_check_options(fishbelt_data, fish_template_and_options, "Current")
```

```
## ✔ All values of `Current` match
```

```
mermaid_import_check_options(fishbelt_data, fish_template_and_options, "Observer emails *")
```

```
## ✔ All values of `Observer emails *` match
```

If data is incorrect, MERMAID will give you the closest option.

```
mermaid_import_check_options(fishbelt_data, fish_template_and_options, "Fish name *")
```

```
## • Some errors in values of `Fish name *` - please check table below
```

```
## # A tibble: 10 × 3
##    data_value                     closest_choice                 match
##    <chr>                          <chr>                          <lgl>
##  1 Parupeneus multifaskiatus      Parupeneus multifasciatus      FALSE
##  2 Parupeneus indikus             Parupeneus indicus             FALSE
##  3 Lutjanus sp.                   Lutjanus                       FALSE
##  4 Acanthurus bariene             Acanthurus bariene             TRUE
##  5 Platax teira                   Platax teira                   TRUE
##  6 Bolbometopon muricatum         Bolbometopon muricatum         TRUE
##  7 Acanthurus auranticavus        Acanthurus auranticavus        TRUE
##  8 Plectorhinchus chaetodonoides Plectorhinchus chaetodonoides TRUE
##  9 Zanclus cornutus               Zanclus cornutus               TRUE
## 10 Variola albimarginata          Variola albimarginata          TRUE
```

There are discrepancies in Fish name that need to be addressed, as indicated by the `FALSE` notes under the `match` column. MERMAID suggests the closest possible matches to assist with data cleaning. These issues must be corrected before proceeding with the data upload.

```
fishbelt_data <- fishbelt_data %>%
  mutate(`Fish name *` = case_when(
    `Fish name *` == "Parupeneus multifaskiatus" ~ "Parupeneus multifasciatus",
    `Fish name *` == "Parupeneus indikus" ~ "Parupeneus indicus",
    `Fish name *` == "Lutjanus sp." ~ "Lutjanus",
    TRUE ~ `Fish name *`
  ))
```

After making these corrections, recheck the values to ensure that they meet the required reference and that all issues have been resolved and a check mark appears.

```
mermaid_import_check_options(fishbelt_data, fish_template_and_options, "Fish name *")
```

```
## ✔ All values of `Fish name *` match
```

Finish checking all the columns.

```
mermaid_import_check_options(fishbelt_data, fish_template_and_options, "Size *")
```

```
## ✔ Any value is allowed for `Size *` - no checking to be done
```

```
mermaid_import_check_options(fishbelt_data, fish_template_and_options, "Count *")
```

```
## ✔ Any value is allowed for `Count *` - no checking to be done
```

Once you get all the check marks, you have the clean version of your data that is ready to be uploaded. Save the clean fishbelt data.

```
write_csv(fishbelt_data, "fishbelt_clean.csv")
```

4. Upload data to MERMAID

Once you have clean data formatted for MERMAID, the next step is to upload the data to MERMAID. You will do one "dry run" before the upload to check your data one last time:

```
mermaid_import_project_data(
  fishbelt_data,
  reef_survey,
  method = "fishbelt",
  dryrun = TRUE
)
```

```
## Records successfully checked! To import, please run the function again with `dryrun = FALSE`.
```

Once you receive this message, `Records successfully checked!`, change the `dryrun` option to `FALSE` to start uploading your data to MERMAID. Once you receive the message, your `Collect records were successfully uploaded!`, navigate to the **Collecting Page** in your project in the MERMAID app to start validating and submitting each sample unit.

```
mermaid_import_project_data(
  fishbelt_data,
  reef_survey,
  method = "fishbelt",
  dryrun = FALSE
)
```

```
## Records successfully imported! Please review in Collect.
```

You're done with uploading your data to MERMAID. Now, you just have to validate and submit your sample units in the **Collecting page** in the MERMAID app as a final step.

MERMAID