University of the Philippines - Visayas

College of Arts and Sciences

Division of Physical Sciences and Mathematics

Computer Science 127 Database Systems

2nd Semester AY 2020-2021

Finfo: A database for Endemic Fish Species in the Philippines

Aleta, Allen Royce

Cabatu-an, John Gabriel

Castro, Karmela

De la Gente, Brethren Ace

Panizales, John Patrick

Piorque, Lester

Villanueva, Joshua

June 23, 2021

Table of Contents

List of Figures

List of Tables

I. INTRODUCTION

The Philippines is a tropical island that is surrounded by large bodies of water. It is also rich in natural resources. It also has different kinds of natural water forms, such as rivers, lakes, falls, gulfs, and straits. Being surrounded by the ocean, the Philippines relies on the waters as an important natural resource. Within these waters, there is an abundance of marine life and materials that are valuable to the nation's people and those across the world. [1]

This project aims to develop a database system organizing important information about endemic species from the Philippines, it will rely on to retrieve the needed information, depending on the circumstance. In addition, it can help researchers, students, and different sectors of the society, to easily access information about the endemic species in the Philippines.

The scope of the database is limited only to the entries of fishes. This database system would allow people all over the world to quickly pull out relevant data that they might find useful, such as the updated official number of species, location, order, family and scientific names of the species. Users are also planned to be able to customize their queries depending on certain categories such as searching for entries based on common name, scientific name, location, and/or description.

II. APPLICATION DOMAIN

The following tables/relations are included in the database.

1. ‘fish’ relation - this relation contains primary details about a specific species. Attributes include species, common_name, status, num_of_species, uses and description. The primary key for this relation is the fish_id.
2. ‘located_in’ relation - this is where fish_ids are associated with a particular location, as well as the depth of the marine environment under which a specific species is occurring. Attributes include fish_id, location_id and depth.
3. ‘location’ relation - this relation houses further details about where a certain species can be found. The location_ID serves as the relations’s primary key. The location_address attribute is a composite attribute which is further composed of the city and province at which the species is located or can be found.
4. The classified_as relation relates a fish_id to particular labels in terms of taxonomy, namely: class, order and family. Since the database’s focus is on fish species, only those taxonomic classifications which would vary between fish species are included.
5. The class relation houses information related to a class. Similarly, the order relation includes details about an order, as well as for the family relation -- also contains data about a particular family. Excluding the class relation, the mentioned taxonomic classification relations also includes a reference to the taxonomic label one level higher than the housing taxonomic label. This is to take into factor the hierarchical nature of taxonomic classifications, i.e., different orders under the same class.

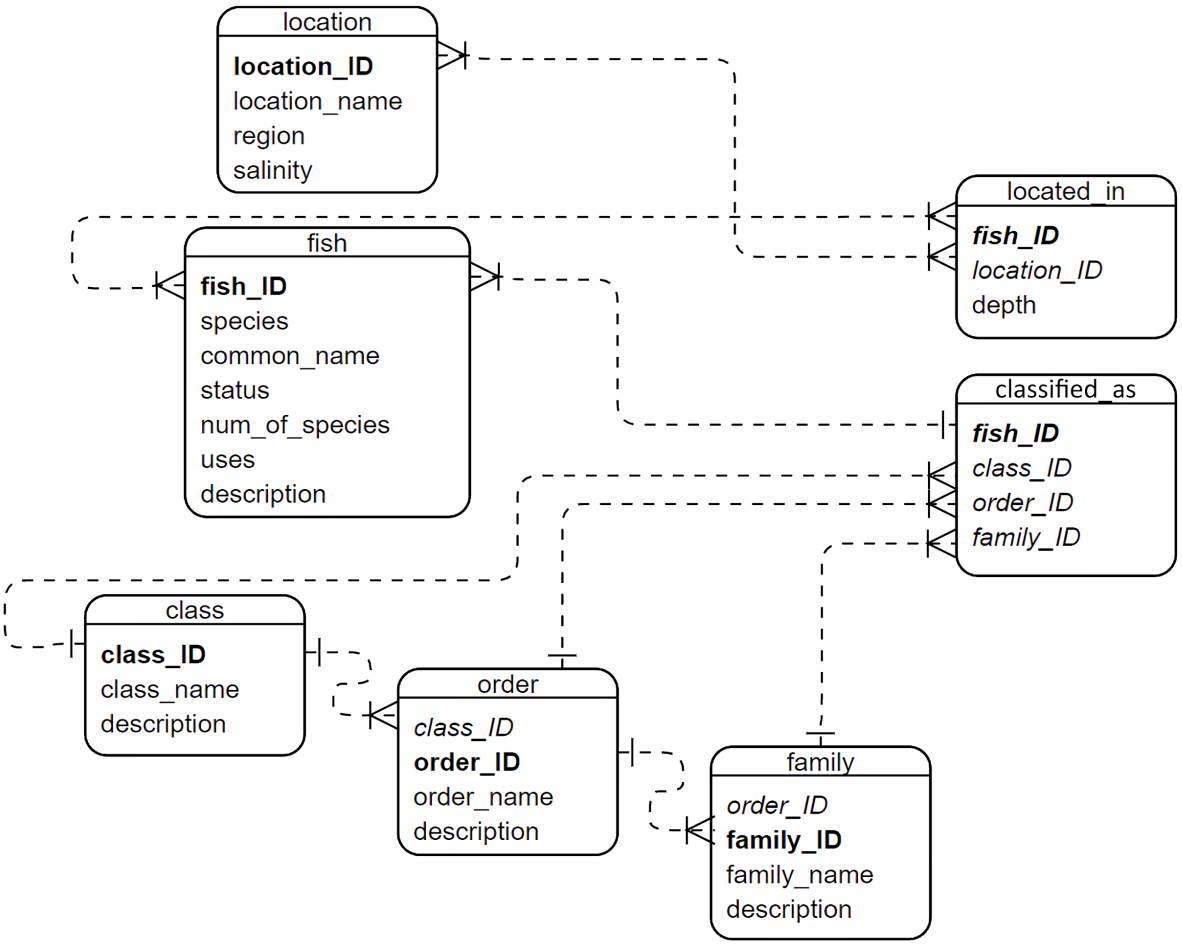

Fig. 1. Database Schema.

III. EER DIAGRAM

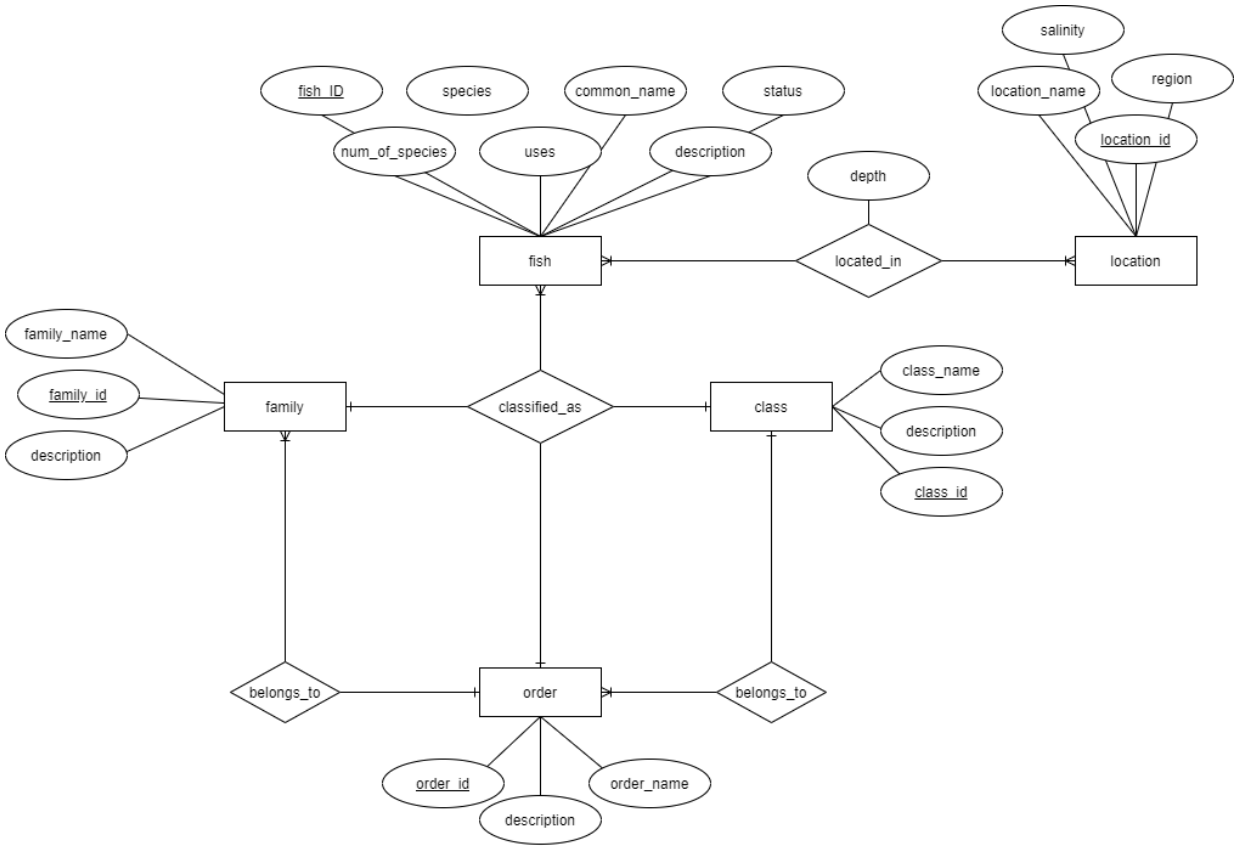

Fig. 2. EERD of Finfo database.

The Enhanced Entity Relationship Diagram was made using the Crow's Foot Notation.

A. Relationship Sentences

1. One or many fish may be located in one or many locations, and vice versa.
2. A fish must be classified to one and only one class, order, and family.
3. A class may have one or many fishes classified to it.
4. An order may have one or many fishes classified to it.
5. A family may have one or many fishes classified to it.
6. A class may have one or many orders. However, an order may belong to one and only one class.
7. A family may belong to one and only one order. However, an order may have one or many families.

IV. NORMALIZATION

A. 1NF

1) fish table

fish_ID	species	common_name	status	num_of_species	uses	description
fish-19820021	Callionymus acutirostris	Pointed dragonet			commercial	
fish-19817023	Callionymus guentheri	Günther's deepwater dragonet				
fish-19850023	Chaetodontoplus caeruleopunctatus	Bluespotted angelfish				Dorsal spines (total): 13; Dorsal soft rays (total): 17; Anal spines: 3; Anal soft rays: 17.

Fig. 3. 1NF of the Fish table.

Since all the tuples in the fish table are already in their atomic form aside from description as it is a text, no significant changes were made.

2) location table

location_ID	location_name	region	salinity
L-2001	Caban Island		marine
V-6031	Cebu		marine
L-2084	Calaguaguin Cove		marine
M-7037	Zamboanga		marine
L-0873	Balayan Bay, Batangas	IV-A	marine

location table

location_ID	salinity
L-2001	marine
V-6031	marine
L-2084	marine
M-7037	marine
L-0873	marine

location_address table

location_ID	barangay	municipality	province	region
L-2001	Caban Island			
V-6031			Cebu	
L-2084	Calaguaguin Cove		Zamboanga	
M-7037				
L-0873	Balayan Bay		Batangas	IV-A

Fig. 4. 1NF of the location table.

The location_name contains the address of the locality of fishes. Following the rules of 1NF, it can be further broken down into *barangay*, *municipality* and *province*. Thus, we made a separate table named *location_address* containing the said attributes and the *location_ID* and *region*. For the location table, we retained the *location_ID* and *salinity* attributes.

No significant changes were made on the classified_as, class, order, and family tables.

B. 2NF

1) classified_as table

fish_ID	class_ID	order_ID	family_ID
fish-19820021	cla-00930	ord-62624	fam-86050
fish-19817023	cla-00930	ord-62624	fam-86050
fish-19850023	cla-00930	ord-75450	fam-44045

Table. 1. 2NF of the classified_as table.

All the taxonomic identifiers are dependent on fish_ID. However, class_ID is also dependent on fish_ID. Similarly order_ID is also dependent on class_ID; and family_ID is also dependent on order_ID. This violates the 2nd normal form, since the non-prime attributes are not fully dependent on the primary_ID.

To fix this, the table is split as follows:

fish_ID	class_ID
fish-19820021	cla-00930
fish-19817023	cla-00930
fish-19850023	cla-00930

Table. 2. fish-class table.

class_ID	order_ID
cla-00930	ord-62624
cla-00930	ord-62624
cla-00930	ord-75450

Table. 3. class-order table.

order_ID	family_ID
ord-62624	fam-86050
ord-62624	fam-86050
ord-75450	fam-44045

Table. 4.order-family table.

V. RELATIONAL MODEL

A. Domain Definition

1) varchar(n)

A variable width character string can contain letters, numbers, and special characters. The maximum size is up to 8,000 characters. [2]

2) int(n)

An int is a medium integer that allows whole numbers between -2,147,483,648 and 2,147,483,647 [2].

3) text

A text can hold a string with a maximum length of 2GB of text data [2].

4) decimal(p, s)

A decimal has a fixed precision and scale numbers. The supported range is from -10^38 +1 to 10^38 –1 [2].

B. Schema Diagram

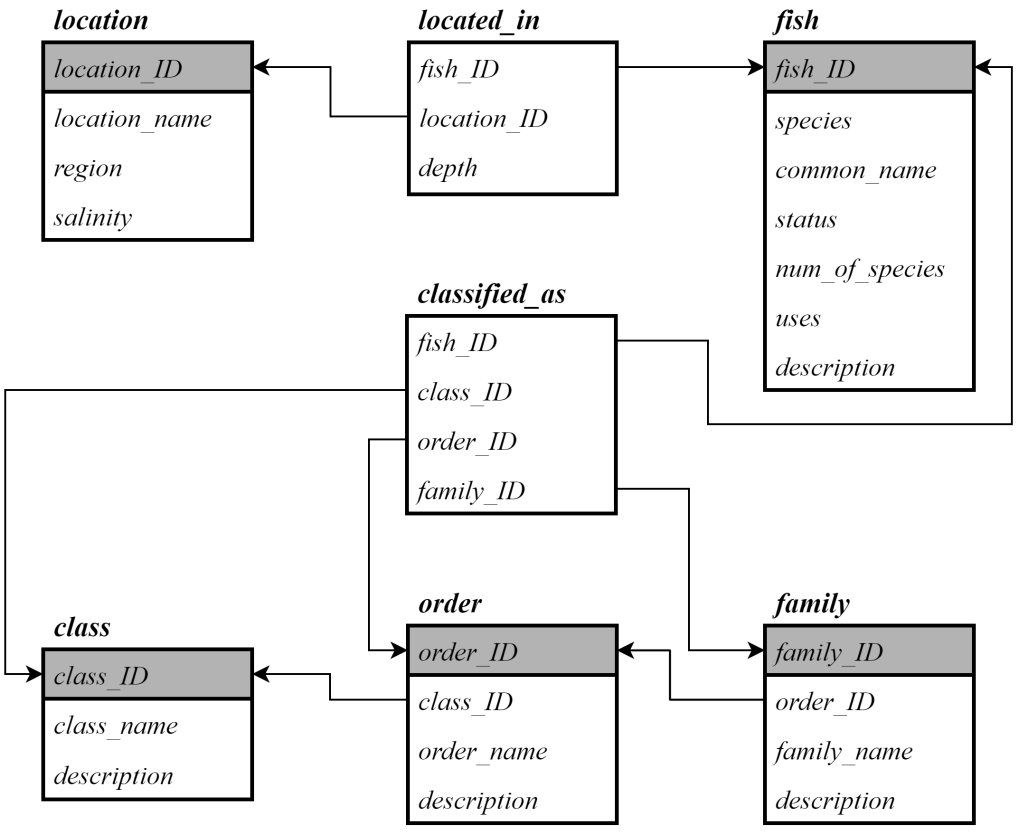

Fig. 5. Schema diagram of the Finfo database.

VI. SQL DDL

This section includes the (1) scripts to create the database based on the relational model, and (2) scripts to create each table of the database based on the relational model.

A. Database

```
CREATE DATABASE finfo_db;
```

B. fish Relation

```
CREATE TABLE fish (
    fish_ID             varchar(12) UNIQUE NOT NULL AUTO_INCREMENT,
    species             varchar(255),
    common_name         text,
    status              varchar(25),
    num_of_species      int(30),
    uses                varchar(255),
    description         varchar(255),
    PRIMARY KEY (fish_ID)
)
```

C. location Relation

```
CREATE TABLE location (
    location_ID         varchar(12) UNIQUE NOT NULL AUTO_INCREMENT,
    salinity            decimal(30, 4),
    PRIMARY KEY (location_ID)
)
```

D. location_address Relation

```
CREATE TABLE location_address (
    location_ID         varchar(12) NOT NULL,
    barangay            varchar(255),
    municipality        varchar(255),
    province            varchar(255),
    region              varchar(255),
    FOREIGN KEY (location_ID)
)
```

E. located_in Relation

```
CREATE TABLE located_in (
        fish_ID             varchar(12) NOT NULL,
        location_ID         varchar(12) NOT NULL,
        depth               decimal(30, 4),
        FOREIGN KEY (fish_ID) REFERENCES fish (fish_ID),
        FOREIGN KEY (location_ID) REFERENCES location (location_ID)
)
```

F. class Relation

```
CREATE TABLE class (
        class_ID            varchar(12) UNIQUE NOT NULL AUTO_INCREMENT,
        class_name          varchar(255),
        description         text,
        PRIMARY KEY (class_ID)
)
```

G. order Relation

```
CREATE TABLE order (
        class_ID            varchar(12) NOT NULL,
        order_ID            varchar(12) UNIQUE NOT NULL AUTO_INCREMENT,
        order_name          varchar(255),
        description         text,
        PRIMARY KEY (order_ID),
        FOREIGN KEY (class_ID) REFERENCES class (class_ID)
)
```

H. family Relation

```
CREATE TABLE family (
        order_ID            varchar(12) NOT NULL,
        family_ID           varchar(12) UNIQUE NOT NULL AUTO_INCREMENT,
        family_name         varchar(255),
        description         text,
        PRIMARY KEY (family_ID),
        FOREIGN KEY (order_ID) REFERENCES order (order_ID),
)
```

I. classified_as Relation

```
CREATE TABLE classified_as (
        fish_ID             varchar(12) NOT NULL,
        class_ID            varchar(12) NOT NULL,
        order_ID            varchar(12) NOT NULL,
        family_ID           varchar(12) NOT NULL,
        FOREIGN KEY (fish_ID) REFERENCES fish (fish_ID),
        FOREIGN KEY (class_ID) REFERENCES class (class_ID),
        FOREIGN KEY (order_ID) REFERENCES order (order_ID),
        FOREIGN KEY (family_ID) REFERENCES family (family_ID)
)
```

VII. SQL DML

This section includes the scripts to insert, update, and delete records in the Finfo database.

A. INSERT

```
INSERT INTO fish (fish_ID, species, common_name, status, num_of_species, uses, description)
VALUES (‘fish-19820021’, 'Callionymus acutirostris', 'Pointed dragonet’, NULL, NULL, NULL, NULL);

INSERT INTO location_address (location_ID, barangay, municipality, province, region)
VALUES (‘L-2001’, ‘Caban Island’, NULL, NULL, ‘Luzon’);

INSERT INTO location (location_ID, salinity)
VALUES (‘L-2001’, ‘marine’);

INSERT INTO class (class_ID, class_name, description)
VALUES (‘cla-00930’, ‘Actinopterygii’, NULL);

INSERT INTO order (class_ID, order_ID, order_name, description)
VALUES (‘cla-00930’, ‘ord-62624’, ‘Callionymiformes’, NULL);

INSERT INTO order (class_ID, order_ID, order_name, description)
VALUES (‘cla-00930’, ‘ord-75450’, ‘Acanthuriformes’, NULL);

INSERT INTO family (order_ID, family_ID, family_name, description)
VALUES (‘ord-62624’, ‘fam-86050’, ‘Callionymidae’, NULL);

INSERT INTO family (order_ID, family_ID, family_name, description)
VALUES (‘ord-75450’, ‘fam-44045, ‘Pomacanthidae’, NULL);
```

B. UPDATE

```
UPDATE fish
SET common_name = “dragonet”
WHERE fish_ID = ‘fish-19910023’;

UPDATE location
SET salinity = “marine”
WHERE location_ID = ‘V-0034’;
```

C. DELETE

```
DELETE FROM fish WHERE fish_ID = ‘fish-19910023’;

DELETE FROM class WHERE class_name = ‘Actinopterygii’;

DELETE FROM location WHERE location_ID = ‘L-2001’;
```

VIII. SAMPLE DATABASE QUERIES

The following are the illustrations of basic and complex queries that are applicable to the Finfo database.

1. List the species and common name with the family name of “Cyprinidae”.

```
SELECT species, common_name
FROM fish
WHERE fish_ID IN
        (SELECT c.fish_ID
        FROM classified_as AS c, family AS f
        WHERE c.family_ID = f.family_ID AND f.family = “Cyprinidae”);
```

2. List all the fishes according to their order.

```
SELECT *
FROM fish
WHERE fish_ID IN
        (SELECT fish_ID
        FROM classified_as
        GROUP BY order_ID);
```

3. Show which of the two is larger in terms of species for each region based on salinity (freshwater or saltwater)

```
SELECT region, MAX(fish_count), salinity
FROM
        (SELECT region, salinity, COUNT(fish_ID) as fish_count
        FROM fish
        GROUP BY salinity, region
        ORDER BY COUNT(fish_ID)
GROUP BY region, salinity
ORDER BY MAX(fish_count);
```

4. List all the fishes alphabetically, found in the Luzon region.

```
SELECT species, common_name
FROM fish
WHERE region = ‘luzon’
ORDER BY species ASC;
```

5. Add the order_name = “Gobiiformes” into the order table.

```
INSERT INTO order (order_name, description)
VALUES (‘Gobiiformes’, NULL);
```

6. Show count in terms of status (endangered, or not and the like) from each region and display the region which has the most count of endangered species

```
SELECT COUNT(fish_ID), status
FROM fish
WHERE status = ‘endangered’ AND COUNT(fish_id) > ALL
                                        (SELECT COUNT(fish_id)
                                        FROM fish
                                        WHERE status = ‘endangered’);
```

7. Find all the families of fishes.

```
SELECT DISTINCT family_name
FROM family;
```

8. Show the number of species found in each region.

```
SELECT COUNT(fish_id), region
FROM fish
GROUP BY region;
```

9. Display the class, order and family of species and the level of depth at which it occur and select only those at less than or equal to 60 ft or 18 meters

```
SELECT species, class, order, family
FROM fish JOIN classified_as USING(fish_ID)
WHERE depth >= 18;
```

10. List the order name, family name, species, status, common name of all fishes.

```
SELECT o.order_name, fam.family_name, f.species, f.common_name, f.status
FROM fish AS f, classified_as AS ca, order AS o, family as fam
WHERE ca.fish_ID = f.fish_ID
AND ca.order_ID = o.order_ID
AND ca.family_ID = fam.family_ID;
```

References

[1] L. Morgan, "List of Natural Resources in the Philippines", *Travel Tips - USA Today*. [Online]. Available: https://traveltips.usatoday.com/list-natural-resources-philippines-54929.html.

[2] "SQL Data Types for MySQL, SQL Server, and MS Access", *W3schools.com*. [Online]. Available: https://www.w3schools.com/sql/sql_datatypes.asp.

[3] R. Fricke, L.A. Jawad, and J.M. Al-Mamry, "Callionymus guentheri summary page", *FishBase,* 2014. [Online]. Available: http://fishbase.org/Country/CountrySpeciesSummary.php?c_code=608&id=59277.

[4] R. Fricke, "Callionymus acutirostris summary page", *FishBase,* 1981. [Online]. Available: http://fishbase.org/Country/CountrySpeciesSummary.php?c_code=608&id=59275.

[5] G. R. Allen, "Chaetodontoplus caeruleopunctatus summary page", *FishBase,* 1985. [Online]. Available: https://www.fishbase.org/Country/CountrySpeciesSummary.php?c_code=608&id=11116.

[6] J. S. Nelson, "FAMILY Details for Callionymidae - Dragonets", *FishBase*, 1994. [Online]. Available: https://www.fishbase.se/summary/FamilySummary.php?ID=435.

[7] J.S. Nelson, "FAMILY Details for Pomacanthidae - Angelfishes", *FishBase*, 1994. [Online]. Available: https://www.fishbase.se/summary/FamilySummary.php?ID=460.

[8] B. S. Dyer and B. Chernoff, "FAMILY Details for Atherinidae - Silversides", *FishBase*, 1996. [Online]. Available: https://www.fishbase.se/summary/FamilySummary.php?ID=218.

[9] J.S. Nelson, "FAMILY Details for Gobiidae - Gobies", *FishBase*, 1994. [Online]. Available: https://www.fishbase.in/summary/FamilySummary.php?ID=405.