THE UNIVERSITY OF QUEENSLAND
AUSTRALIA

INFS 7901 Database Principle

Project Proposal

Turtle Conservation and Research Database System

Picture from: https://www.adoptananimalkits.com/animal_encyclopedia

Name: Jie Zhang

Student Number: s4755838

Project description

The Turtle Conservation and Research Database System has been meticulously designed to facilitate the conservation and study of turtles. This comprehensive platform serves as a centralized repository, aiding researchers and wildlife organizations in collecting, managing, and analyzing data concerning individual turtles, their species, habitats, and conservation efforts.

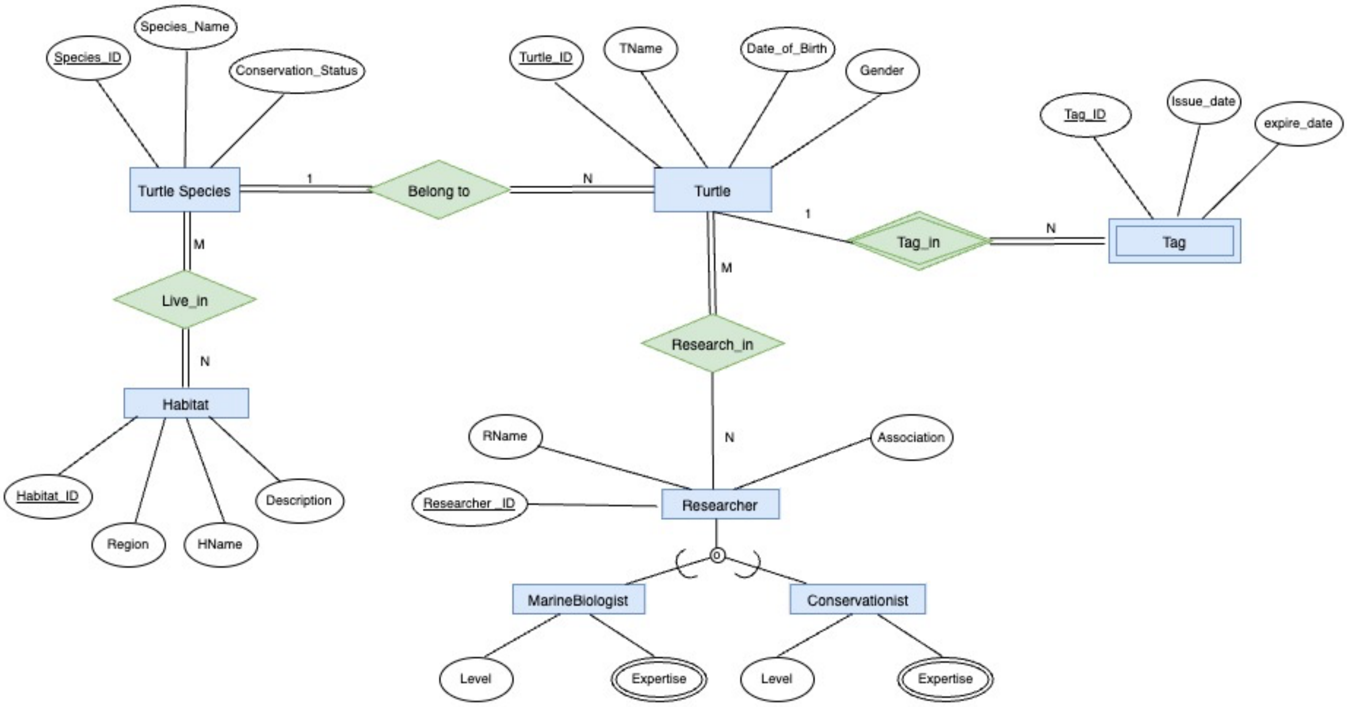
The database encompasses seven distinct domains, namely Turtle, Turtle species, Habitat, Tag, Researcher, Marine Biologist, and Conservationist. The attribute domains primarily consist of character, numeric, date, time, and Boolean values, each of which is extensively defined in the accompanying text.

This application is equipped with a wide range of functionalities that enable researchers to manage individual turtles, species, habitats, observations, conservation projects, and researcher profiles. Additionally, the platform supports data analysis, visualization, and collaboration among researchers, ensuring secure access control and user management. The system provides three main functionalities, which are as follows:

1. Researchers can register and log in using their unique Researcher_ID.
2. All entities' information can be accessed through a simple search function.
3. Users can add, delete, and update data related to all entities.

The final project will be developed using MySQL and Python, providing a robust and reliable database management system for researchers and conservationists.

ER Diagram



ER-diagram: Turtle Conservation and Research Database

Relational cardinality

(Non-weak) Binary 1: N Relationship	Binary M: N Relationship
Turtle (N)- Turtle Species (1)	Turtle Species (M) – Habitat (N) Turtle (M) – Researcher (N)

Participation cardinality

Relationship	Participation cardinality
Belong_to	Turtle (T)-Turtle Species (T)
Live_in	Turtle Species (T)- Habitat (T)
Research_in	Researcher (P) – Turtle (T)
Tag_in	Turtle (P) – Tag (T)

Schema

Mapping ER-diagram to Schema processing

Step1: Strong entities mapping: Turtle, Turtle Species, Habitat, Researcher

Choose one key attribute as primary key (using underscore) for each relationship.

Turtle (**Turtle_ID**, TName, Date_of_Birth, Gender, Species_ID (FK))

Turtle Species (**Species_ID**, Species_Name, Conservation_Status)

Habitat (**Habitat_ID**, Region, HName, Description)

Researcher (**Researcher _ID**, RName, Association)

Step 2: Weak Entity Mapping: There is only one weak entity Tag.

Combine the primary key of weak entity and the owner's primary key which is Turtle_ID as Tag_in. Also add owner's primary key as the foreign key of Tag_in.

Tag_in (**Turtle_ID**, **Tag_ID**, Issue_date, expire_date)

Foreign Key: Tag_in.Turtle_ID -> Turtle.Turtle_ID

Step 3:**(Non-weak) Binary 1: N Relationship:** Turtle (N)- Turtle Species (1)

Add foreign key to Turtle (N) side, as a result

Foreign Key: Turtle. Species_ID-> Turtle Species. Species_ID

Step 4: Binary M: N Relationship:

Turtle Species (M) – Habitat (N)

Turtle (M) – Researcher (N)

To create a new relation to represent it and combine the two entities' primary key as the new relationship's primary key, then add the foreign key as follows:

Live_in (**Species_ID, Habitat_ID**)

Research_in (**Turtle_ID, Researcher _ID**)

Foreign Key:

Live_in. Species_ID-> Turtle Species. Species_ID

Live_in. Habitat_ID-> Habitat. Habitat_ID

Research_in. Turtle_ID-> Turtle. Turtle_ID

Research_in. Researcher _ID-> Researcher. Researcher _ID

Step 5: Multivalued Attributes and sub-class: MarineBiologist, Conservationist

Create a new relation for each multivalued attribute, and choose this method to deal with sub-class, create a relational table for the superclass and create a relational table for each subclass. The primary key of each of the subclass is the primary key of the superclass.

MarineBiologist (**Researcher _ID**, Level, Expertise)

Conservationist (**Researcher _ID**, Level, Expertise)

MarineBiologist_ Expertise (**Researcher _ID**, Expertise)

Conservationist_ Expertise (**Researcher _ID**, Expertise)

Foreign Key:

MarineBiologist_ Expertise. Researcher _ID-> MarineBiologist. Researcher _ID

Conservationist_ Expertise. Researcher _ID-> Conservationist. Researcher _ID

Relations

Turtle (**Turtle_ID**, TName, Date_of_Birth, Gender, Species_ID (FK))

Turtle Species (**Species_ID**, Species_Name, Conservation_Status)

Habitat (**Habitat_ID**, Region, HName, Description)

Researcher (**Researcher _ID**, RName, Association)

MarineBiologist (**Researcher _ID**, Level, Expertise)

Conservationist (**Researcher _ID**, Level, Expertise)

Tag_in (**Turtle_ID**, **Tag_ID**, Issue_date, expire_date)

Live_in (**Species_ID, Habitat_ID**)

Research_in (**Turtle_ID, Researcher _ID**)

MarineBiologist_ Expertise (**Researcher _ID**, Expertise)

Conservationist_ Expertise (**Researcher _ID**, Expertise)

Foreign Keys

Tag_in.Turtle_ID -> Turtle.Turtle_ID

Turtle. Species_ID-> Turtle Species. Species_ID

Live_in. Species_ID-> Turtle Species. Species_ID

Live_in. Habitat_ID-> Habitat. Habitat_ID

Research_in. Turtle_ID-> Turtle. Turtle_ID

Research_in. Researcher _ID-> Researcher. Researcher _ID

MarineBiologist_ Expertise. Researcher _ID-> MarineBiologist. Researcher _ID

Conservationist_ Expertise. Researcher _ID-> Conservationist. Researcher _ID

Functional Dependencies

Turtle

Turtle_ID→{TName, Date_of_Birth, Gender, Species_ID}

Turtle Species

Species_ID→{Species_Name, Conservation_Status}

Habitat

Habitat_ID→ {Region, HName, Description}

Researcher

Researcher _ID, Association→ {RName}

- MarineBiologist_ Expertise:

  Researcher _ID→ {Level, Expertise}

- Conservationist_ Expertise:

  Researcher _ID→ {Level, Expertise}

Tag

Turtle_ID, Tag_ID→ {Issue_date, expire_date}

Research_in

Turtle_ID→ Researcher _ID

Normalized Schema (include domain constrains)

Table 1 - Turtle

Turtle_ID *(PK)*	TName	Date_of_Birth	Gender	Species_ID*(FK)*
varchar or char data type, depending on the length of the ID.	varchar data type, with a maximum length of characters allowed.	date data type, used to store date values in a specific format.	varchar data type	Numeric data type, used to store the unique identifier of each species in the table.

It's already in BCNF, because Turtle_ID→{TName, Date_of_Birth, Gender, Species_ID} and Turtle_ID is a superkey.

Table 2 -Turtle Species

Species_ID *(PK)*	Species_Name	Conservation_Status
Numeric data type, used to store the unique identifier of each species in the table.	varchar data type	varchar data type, used to store the conservation status of each species, such as "Endangered", "Threatened", "Vulnerable" etc.

It’s already in BCNF, because Species_ID→{Species_Name, Conservation_Status} and Species_ID is a superkey.

Table 3 -Habitat

Habitat_ID *(PK)*	Region	HName	Description
Numeric data type, used to store the unique identifier of each habitat in the table.	Varchar data type, used to store the region where the habitat is located.	Varchar, used to store the name of the habitat.	Varchar, used to store a description of the habitat.

It’s already in BCNF, because Habitat_ID→ {Region, HName, Description} and Habitat_ID is a superkey.

Not normalised Table-Researcher

Researcher _ID *(PK)*	Association *(PK)*	RName

Researcher _ID, Association→ {RName}

Because RName is only dependent on Researcher _ID, it’s partial dependent, it’s not in 2NF, so split it to two tables, first one contains Researcher _ID, Name, Association_ID as a foreign key referring to another new table with Association_ID as primary key. Table 4 and Table 5 is in 3NF and also satisfy BCNF.

Table 4 - Researcher

Researcher _ID *(PK)*	RName	Association_ID *(FK)*
varchar or char data type, depending on the length of the ID.	Varchar, used to store the name of the Researcher	Numeric data type, used to store the unique identifier of each association in the table.

Table 5 - Association

Association_ID *(PK)*	Association
Numeric data type, used to store the unique identifier of each association in the table.	Varchar data type, used to store the name of the association.

Table 6 - MarineBiologist

Researcher _ID *(PK)*	Level
varchar or char data type, depending on the length of the ID.	VARCHAR data type, used to store the level of the MarineBiologist

Researcher _ID→ {Level}, so Table6 is already in BCNF, because Researcher _ID is a superkey.

Table 7 - Conservationist

Researcher _ID *(PK)*	Level
varchar or char data type, depending on the length of the ID.	VARCHAR data type, used to store the level of the Conservationist

Same explanation as Table6

Table 8 - MarineBiologist_ Expertise

Researcher _ID *(PK) (FK)*	Expertise *(PK)*
varchar or char data type, depending on the length of the ID.	used to store the area of expertise of MarineBiologist

As MarineBiologist_ Expertise is mapped from multivalued attributes, the primary kye is combined with MarineBiologist's primary key (Researcher_ID) and Expertise together. And

Researcher_ID is a foreign key as well. There is no non-trivial relationship, so Table 8 is already in BCNF.

Table 9 - Conservationist_ Expertise

Researcher _ID *(PK) (FK)*	Expertise *(PK)*
varchar or char data type, depending on the length of the ID.	used to store the area of expertise of Conservationist

Same situation as Table 8, Table 9 is already in BCNF.

Table 10 - Research_in

Turtle_ID *(PK) (FK)*	Researcher _ID *(FK)*
varchar or char data type, depending on the length of the ID.	varchar or char data type, depending on the length of the ID.

Turtle_ID→ Researcher _ID, and Turtle_ID is a superkey, so Table 10 is in BCNF.

Table 11 - Live_in

Species_ID *(PK) (FK)*	Habitat_ID*(PK) (FK)*
Numeric data type, used to store the unique identifier of each species in the table.	Numeric data type, used to store the unique identifier of each habitat in the table.

There is no non-trivial relationship, so Table 11 is already in BCNF.

Table12 - Tag_in

Turtle_ID(PK) *(FK)*	Tag_ID *(PK)*	Issue_date	expire_date
varchar or char data type, depending on the length of the ID.	Numeric data type, used to store the unique identifier of each tag in the table.	Date data type, used to store the date when the tag was issued.	Date data type, used to store the date when the tag will expire.

Because Tag is a weak entity, Tag_in is a new created relationship between owner Turtle and tag, in Table12, Turtle_ID and Tag_ID are combined to represent primary key. And Turtle_ID is also a foreign key referring to Table1- Turtle. And table10 is already in BCNF, because Turtle_ID, Tag_ID→ {Issue_date, expire_date}, where Turtle_ID plus Tag_ID is a superkey.

PK = Primary Key; FK = Foreign Key

Table1 – Table12 (blue colour) are all normalised schema.