Database Design Specification

Aqua-Zoo Management System

Version 4.3

November 16, 2016

Prepared by:

Akash Rajendra Ventekar
Aadil Ahmed Adam
Poornima Byre Gowda
Jeevan Chandrashekar

Prepared for:

Dr. Ramazan Aygun
CS 687 Database Systems
University of Alabama in Huntsville

Table of Contents

Revisions

Version	Authors	Description of Version	Date Completed
1.0	Akash, Poornima, Aadil, Jeevan	This was a draft of the database topic with a short informal description.	September 06, 2016
1.1	Akash, Poornima, Aadil, Jeevan	All members of the team agreed to submit this document as the initial topic submission.	September 07, 2016
2.0	Akash, Poornima, Aadil, Jeevan	This was a draft of the database description of the tables and a plan for completing the project.	September 12, 2016
2.1	Akash, Poornima, Aadil, Jeevan	All members of the team agreed to submit this document as the description of our database.	September 14, 2016
3.0	Akash, Poornima, Aadil, Jeevan	All members of the team met to work on the various entities and its attributes.	October 21, 2016
3.1	Akash, Poornima, Aadil, Jeevan	A draft of the formal description was developed. The team discussed the relationships between entitles, the cardinalities, and the constraints. A draft of the ER diagram and Schema diagram was developed.	October 22, 2016
3.2	Aadil, Jeevan	All members of the team agreed to submit the formal description consisting of the ER diagram and the Relational Schema Diagram.	October 24, 2016
3.3	Akash, Poornima, Aadil, Jeevan	Revised the formal descriptions after modifying the ER diagram. Finalized the ER diagram and schema diagram.	October 28, 2016
4.1	Akash, Poornima, Aadil, Jeevan	Began designing the web interface to login and access the database utilizing HTML and PHP programming.	November 01, 2016
4.2	Akash, Poornima	Worked on the implementation of the database in Cloud 9 using MySQL.	November 13, 2016
4.3	Akash, Poornima, Aadil, Jeevan	Completed the final report of our cloud-based implementation of our database and web-interface.	November 16, 2016

1. Introduction

1.1. Objective

Design a database that is focused towards the implementation of the basic functionalities of a database application. Here we depict this functionality in the form of Aqua-Zoo management responsibilities of system administrators.

1.2. Purpose

Managing an Aqua-Zoo is a complex and time-consuming responsibility for any zoo administrator. If the resources of the zoo are not managed efficiently and effectively, then the management may suffer from inconsistencies in the database of animals, the sponsorship funds associated with the animals, personnel data and the funds gathered by tourism and to the zoo as a whole. An Aqua-Zoo management system should be able to keep an updated database of the animals in the zoo, their health, description and trainer information if it exists, database of all the personnel's working for the zoo, business of the zoo with its restaurants, exhibitions to name a few. If it is not maintained properly then it would lead to conflict of interest and loss of crucial information of the zoo. Managing this data would reduce the risk of inconsistent data and better management of resources of the Aqua-Zoo.

1.3. Scope

Our solution is to develop a database that will be capable of storing the information of the various animals that exist in the environment, keep a record of trainers and sponsors of the zoo. Our initial approach will be to collect the data about the species of various animals located in the zoo and maintain a database for the same. Additionally, we will also keep track of all the sponsors and trainers available for some of the animals in the zoo. This technique will then allow for the design of a database that captures the information required for the administrators to manage the resources of the Aqua-Zoo and also give the users of this system a view of what they can expect when they visit the zoo.

1.4. Deliverable

The team will deliver a report that will further describe the requirements of an Aqua-Zoo Management system. We will also provide a plan as to how we plan to implement the database. This specification document will contain an in-depth design description of the database. Finally, the team will provide a final report of the Aqua-Zoo Management Database along with the implementation of the database.

1.5. Definitions and Acronyms

- Administrator – A person responsible for managing the Aqua-Zoo
- CS – abbreviation for: Computer Science.
- ER – abbreviation for Entity Relationship Diagram
- IDE – abbreviation for: Integrated Development Environment
- CSS – abbreviation for: Cascading Style Sheets
- Entity – a something that exists in the real world with an independent existence.
- Cloud9 – IDE utilized for developing front-end and back-end of our system.
- HTML5 – abbreviation for: HyperText Markup Language 5
- MySQL – A relational database management system.
- PHP – A server side scripting language designed for developing web applications.

- AID – abbreviation for: Animal Identification Number. Uniquely identifies an Animal entity.
- TID – abbreviation for: Trainer Identification Number. Uniquely identifies a Trainer entity.
- SID – abbreviation for: Species Identification Number. Uniquely identifies a Description entity.
- SPID – abbreviation for: Sponsor Identification Number. Uniquely identifies a Sponsor entity.

2. Data Design

2.1. Database Description

- The Aqua-Zoo management system is organized into components. Each of these components has a unique identification number (UID) and a set of attributes which will describe the component in much finer detail. Each of these components will be utilized by the administrator of the management system in a different way and they may interact with each other to give a system which will be useful for the administrators and the users of the application.
- We will store the details concerning the animals, their description, medical records, trainers associated and the sponsors available for the animals in the Aqua-Zoo database. Most of these components will have a foreign key identifier linked to the primary key identifier of the description entity through which the other components are interconnected. The description entity will store the information about a particular species an animal belongs to and all the characteristics of that species such as its scientific name, lifespan, habitat, average height, average weight etc.
- We want to maintain the details of all the available trainers in the Aqua-Zoo for various animals. The trainers are assigned a unique identification number TID and we also store their details such as their names, salary, address and the AID of the animal they train. The TID is linked to the animal entity through the AID of the animal entity. The animal identification number (AID) here behaves as the foreign key linking the trainer entity with the animal entity.
- The trainers associated with a particular animal will also have the access to the database through which they will maintain the details about the medical history of that animal. Each entry in the medical history entity will have a unique record number which will differentiate itself from other entries. This entity will also be linked to the animal entity and trainer entity since it is the responsibility of the trainer to maintain the details about the animal they taking care of. The administrator records the weekly updates of the health of the animals based on the observations done by the Trainer. The attribute AID and TID will act as a foreign key linking the other two entities.

- We want to keep track of the sponsors that exist on some of the animals in the Aqua-Zoo. For each sponsor, we will maintain the sponsor identification number, their full name, address, amount of sponsorship, and associated animal identification number which they sponsor. One sponsor can sponsor multiple animals based on different AID for some defined period of time. Each sponsor will have SPID which is the unique identification number for each sponsor who is registered with the Aqua-Zoo administrator as a sponsor. This SPID will be linked to the animal entity through the AID of the animal entity.
- The database will maintain records for user information along with the physical hardware machines and operating systems the users are allowed to access.

2.2. Entity Types

- The Animal entity stores the details of different animals present in the zoo managed by the administrators. There can be many animals of the same species located in the Aqua-Zoo.
- The Description entity represents the biological and geographical details about each species of animals located in the Aqua-Zoo.
- The Trainer entity maintains a database of the available trainers in the Aqua-Zoo. These trainers will be associated with one or more animals in the Aqua-Zoo.
- The Sponsor entity represents the various sponsors who have been registered by the Aqua-Zoo management. This entity will contain the information about each and every sponsor enrolled with the Aqua-Zoo and the animals they sponsored.
- The Medical History entity represents the details of the medical details about the animals such the disease and their diet plan which is maintained by their particular trainers.

2.3. Entity Attributes

- The Animal entity contains a unique identification number called the AID for each animal present in the Aqua-Zoo. The animal will possess a name given to the animal by the zoo authorities, its sex and age, height, weight, location of the animal in the Aqua-Zoo.
- The Description entity contains the general information about the species a particular animal belongs to, along with its characteristics such as its common name, its image, scientific name, its average life span, average species height and weight, the status of conservation of the species, the habitat its normally located in, its type such as a carnivore or a omnivore, brief description about the animal and under which category is it classified such as mammals, reptiles, etc.
- The Trainer entity contains the unique trainer identification number, their first name , middle initial, last name, salary and the timings of their duties in the Aqua-Zoo.

- The Sponsor entity will possess a unique identification number, full name of the sponsor, their sponsoring amount, and address and email Id of the sponsor, the time period of their sponsorship.
- The Medical history entity will possess a record number of the entry about the disease, its date, diet plan of the concerned animal during the disease. The trainer will enter all these details with respect to the animals trained by them.

2.4. Entity-Relationship Diagram of Aqua-Zoo Management Database

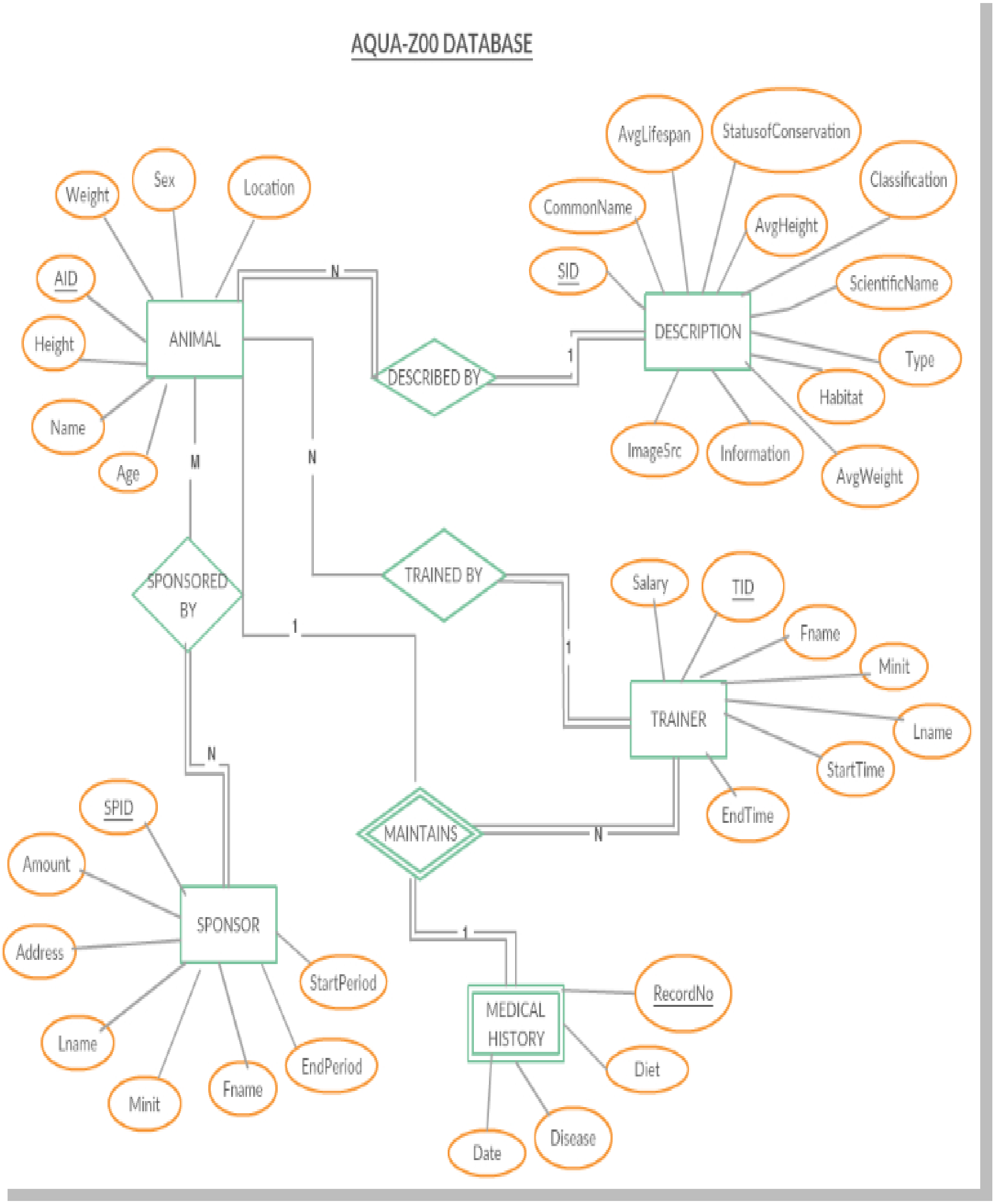
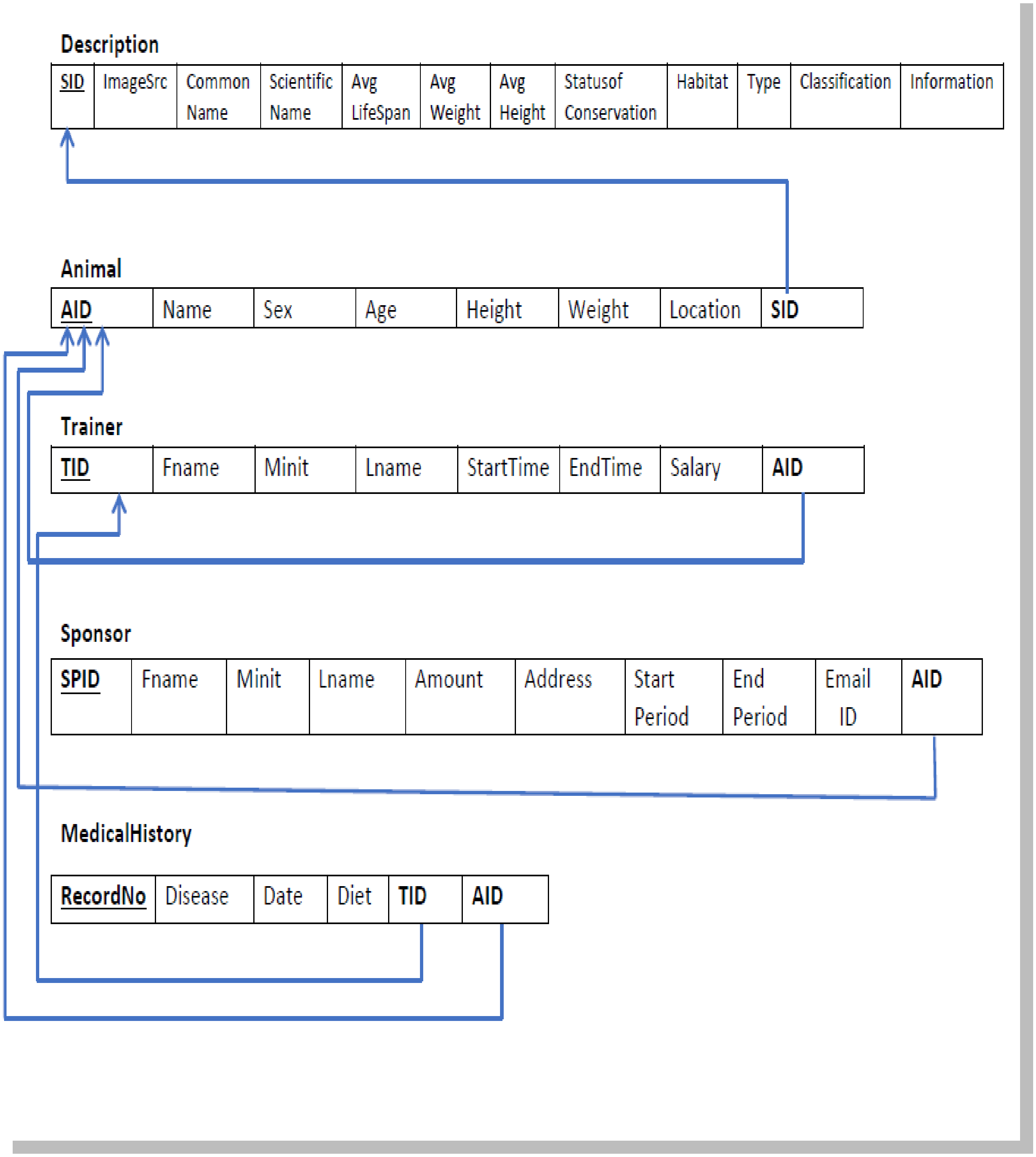


Figure 1. Entity-Relationship Diagram for Aqua-Zoo Management Database

2.5. Schema of Aqua-Zoo Management Database



Description

SID	ImageSrc	Common Name	Scientific Name	Avg LifeSpan	Avg Weight	Avg Height	Statusof Conservation	Habitat	Type	Classification	Information

Animal

AID	Name	Sex	Age	Height	Weight	Location	**SID**

Trainer

TID	Fname	Minit	Lname	StartTime	EndTime	Salary	**AID**

Sponsor

SPID	Fname	Minit	Lname	Amount	Address	Start Period	End Period	Email ID	**AID**

MedicalHistory

RecordNo	Disease	Date	Diet	**TID**	**AID**

Figure 2. Schema Diagram of Aqua-Zoo Management Database

2.6. Relationships, Cardinalities, and Partial and Total Participation Constraints

- DESCRIBED BY, an N: 1 (many-to-one) weak entity relationship between Animal and Description. An animal must possess its species description to which it belongs to in the description component. One or more animals can belong to the same species of animals but a species cannot have different type of animals. The description entity should describe at least one animal in the animal entity. An animal cannot exist without having its species description. All participations are determined to be total.
- TRAINED BY, a 1: N (one-to-many) weak entity relationship type between strong entity Animal and Trainer Component. An animal may or may not have a trainer. If a trainer exists for an animal then that animal can have no more than one trainer. A trainer, if he/she exists then they will be a trainer for an animal in the zoo else they will not be called a trainer. A trainer can train one or more animals but an animal would be trained by a single trainer. The trainer entity is determined to have total participation.
- SPONSORED BY, an M: N (many-to-many) relationship type between strong entity Animal and Sponsor entity. An animal could be sponsored by a sponsor who has been registered by the zoo administration. If a sponsor exists then he should be sponsoring one or more animals present in the zoo. An animal can have one or more sponsors and every sponsor can sponsor one or more animals. Both participations are total.
- MAINTAINS, a 1:1:N weak ternary relationship type between a strong entity Animal, Trainer and weak entity Medical History. An animal may or may not have a medical history unless it has been diagnosed with some disease. The trainer will be in charge for maintaining the medical history of the animal he/she is training. A medical history would belong to one among the animals of the zoo managed by the trainers of that particular animal. Since it is a weak entity, it is always involved in total participation with its identifying relation.

2.7. Weak Entity Types

- The Medical History component is a weak entity type of the Animal entity and the Trainer entity. Each Medical History component is dependent on the Animal for which the medical record holds correct and the Trainer entity who maintains the records of the animal he/she is training.

2.8. Specialization Constraints

- Animal components will only be maintained for eligible species description. An animal can exist only if it has been described in the description component. If there is no entry about the specified animal in the description component then the details about that animal cannot exist in the animal entity.

- The Aqua-Zoo Administrator user will have access to all trainer and sponsor data since they are registered with the aqua-zoo management and given access to view the operations.
- The Aqua-Zoo administrator will be responsible for maintaining the medical records of the animals in the Aqua-zoo weekly. The trainers will observe the medical conditions of their assigned animals; document these records and provide it to the administrator.
- The default value for the Habitat, Classification, StatusofConservation, Information and Type attribute of Description relation will be set to “NA- Not Available”.
- The default value for the sex attribute of Animal will be set to check for one of the options “M”- Male or “F”– Female.
- The default value for the Start Time and End Time of the Trainer relation will be set to “00:00:00”.
- The default value for the Start period and End Period of the Sponsor will be set to “1970-01-01”.
- The default value for the AvgHeight and AvgWeight of the Description relation will be set to “00”.
- The default value for the Height and Weight of the Animal relation will be set to “00”.
- The default value for the Address and Email of the Sponsor relation will be set to “NA”.
- The default value for the Date of the Medical history relation will be set to “1970-01-01”.
- The default value for the Disease and Date of the Medical history relation will be set to “NA”.

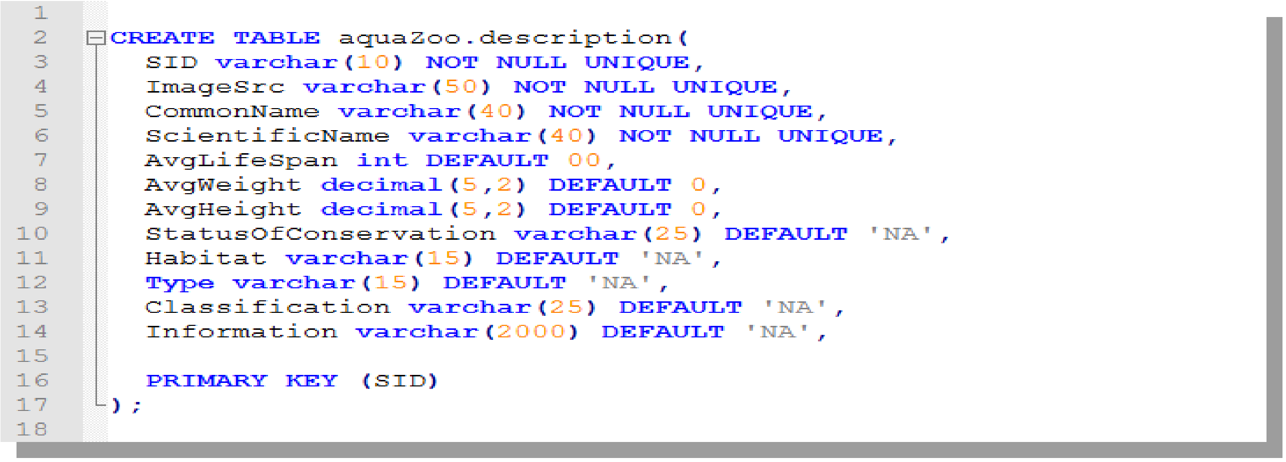
2.9. Mapping of Entity-Relationship Diagram to Relational Schema

- We created the relations Animal, Description, Trainer and Sponsor to correspond to the regular entity types Animal, Description, Trainer and Sponsor, respectively.
- We created the relation Medical History to correspond to the weak entity type Medical History. We included the primary key AID of the Animal relation as a foreign key attribute of Medical History. We included the primary key TID of the Trainer relation as a foreign key attribute of Medical History. The primary key of the Medical History is the combination of AID, TID and the partial key Record Number.

- SPONSORED BY is a weak relationship type of M: N cardinality between the strong entity Animal and sponsor and hence we need to create a relation. Due to the increasing redundancy in creating this relation, the team determined to neglect this relation and include the attributes of the proposed Sponsored By relation into the relation Sponsor.

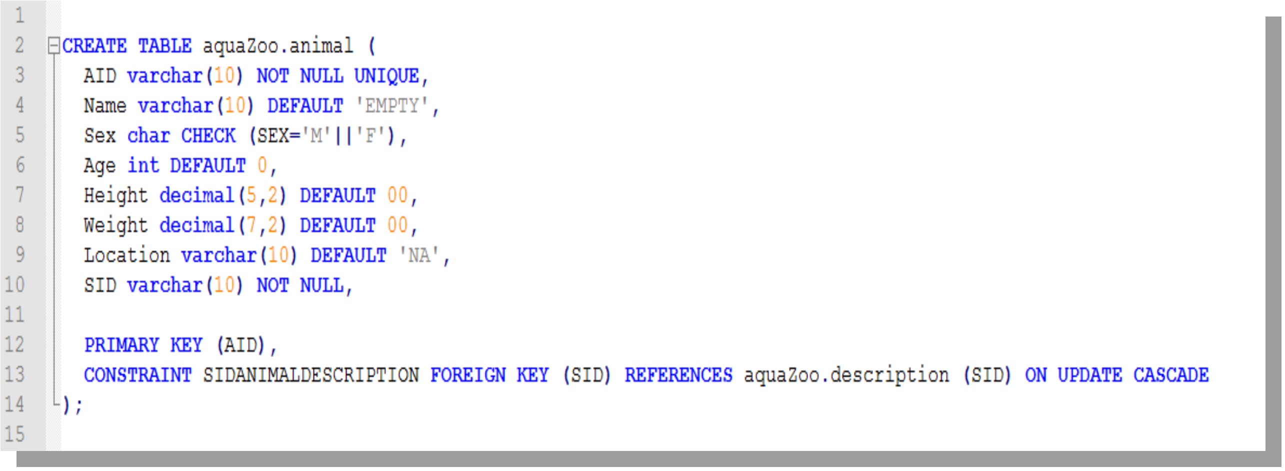
2.10. Mapping Relational Schema to Table Creation

2.10.1. CREATE TABLE Statements



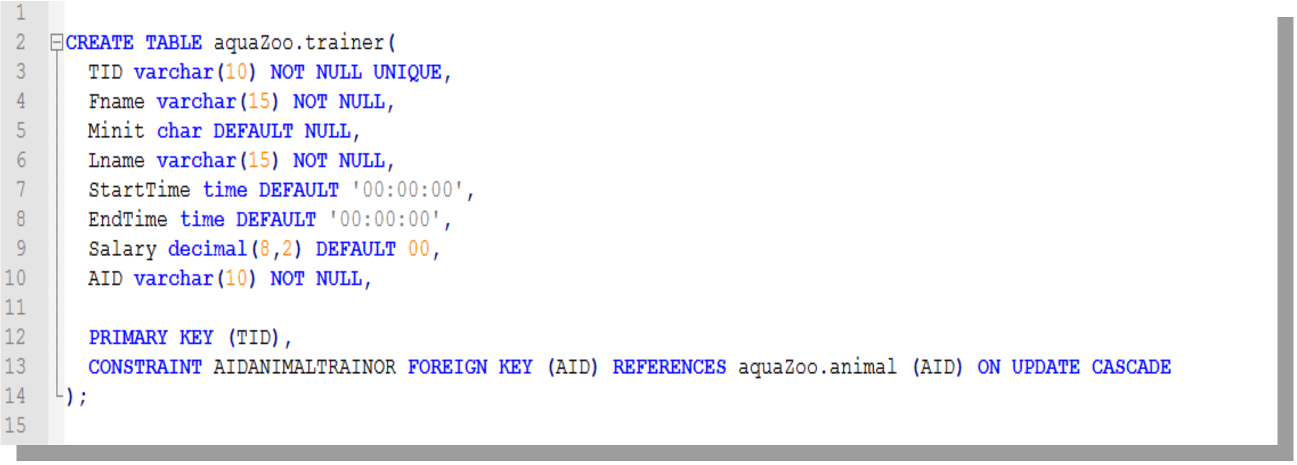
```
CREATE TABLE aquaZoo.description(
   SID varchar(10) NOT NULL UNIQUE,
   ImageSrc varchar(50) NOT NULL UNIQUE,
   CommonName varchar(40) NOT NULL UNIQUE,
   ScientificName varchar(40) NOT NULL UNIQUE,
   AvgLifeSpan int DEFAULT 00,
   AvgWeight decimal(5,2) DEFAULT 0,
   AvgHeight decimal(5,2) DEFAULT 0,
   StatusOfConservation varchar(25) DEFAULT 'NA',
   Habitat varchar(15) DEFAULT 'NA',
   Type varchar(15) DEFAULT 'NA',
   Classification varchar(25) DEFAULT 'NA',
   Information varchar(2000) DEFAULT 'NA',

   PRIMARY KEY (SID)
);
```

Figure 3. Description Table with Primary Key SID.



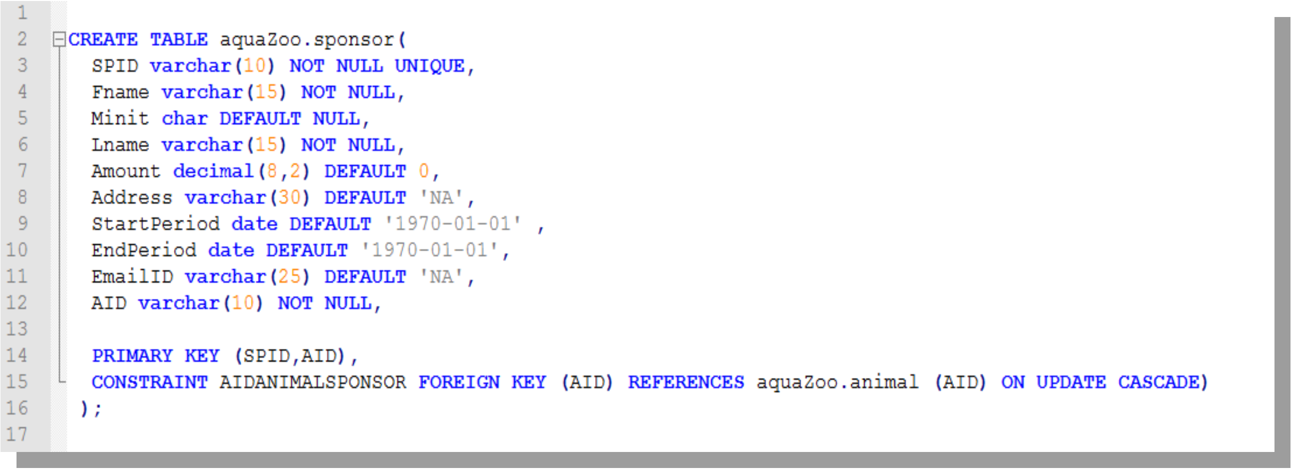
```
CREATE TABLE aquaZoo.animal (
  AID varchar(10) NOT NULL UNIQUE,
  Name varchar(10) DEFAULT 'EMPTY',
  Sex char CHECK (SEX='M'||'F'),
  Age int DEFAULT 0,
  Height decimal(5,2) DEFAULT 00,
  Weight decimal(7,2) DEFAULT 00,
  Location varchar(10) DEFAULT 'NA',
  SID varchar(10) NOT NULL,

  PRIMARY KEY (AID),
  CONSTRAINT SIDANIMALDESCRIPTION FOREIGN KEY (SID) REFERENCES aquaZoo.description (SID) ON UPDATE CASCADE
);
```

Figure 4. Animal Table with Primary Key AID and Foreign Key SID.



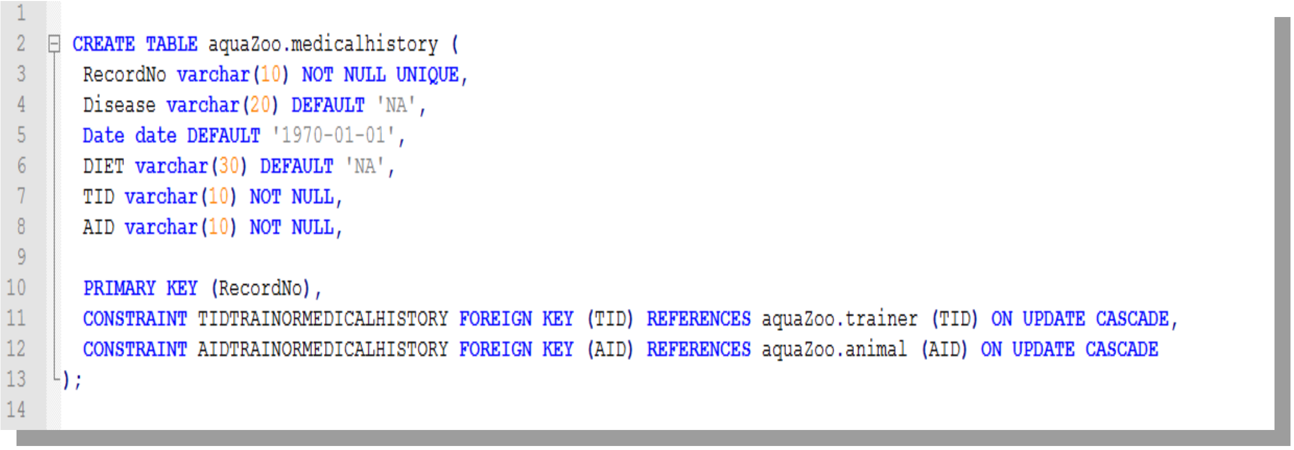
```
CREATE TABLE aquaZoo.trainer(
  TID varchar(10) NOT NULL UNIQUE,
  Fname varchar(15) NOT NULL,
  Minit char DEFAULT NULL,
  Lname varchar(15) NOT NULL,
  StartTime time DEFAULT '00:00:00',
  EndTime time DEFAULT '00:00:00',
  Salary decimal(8,2) DEFAULT 00,
  AID varchar(10) NOT NULL,

  PRIMARY KEY (TID),
  CONSTRAINT AIDANIMALTRAINOR FOREIGN KEY (AID) REFERENCES aquaZoo.animal (AID) ON UPDATE CASCADE
);
```

Figure 3. Trainer Table with Primary Key TID.



```
CREATE TABLE aquaZoo.sponsor(
  SPID varchar(10) NOT NULL UNIQUE,
  Fname varchar(15) NOT NULL,
  Minit char DEFAULT NULL,
  Lname varchar(15) NOT NULL,
  Amount decimal(8,2) DEFAULT 0,
  Address varchar(30) DEFAULT 'NA',
  StartPeriod date DEFAULT '1970-01-01' ,
  EndPeriod date DEFAULT '1970-01-01',
  EmailID varchar(25) DEFAULT 'NA',
  AID varchar(10) NOT NULL,

  PRIMARY KEY (SPID,AID),
  CONSTRAINT AIDANIMALSPONSOR FOREIGN KEY (AID) REFERENCES aquaZoo.animal (AID) ON UPDATE CASCADE)
 );
```

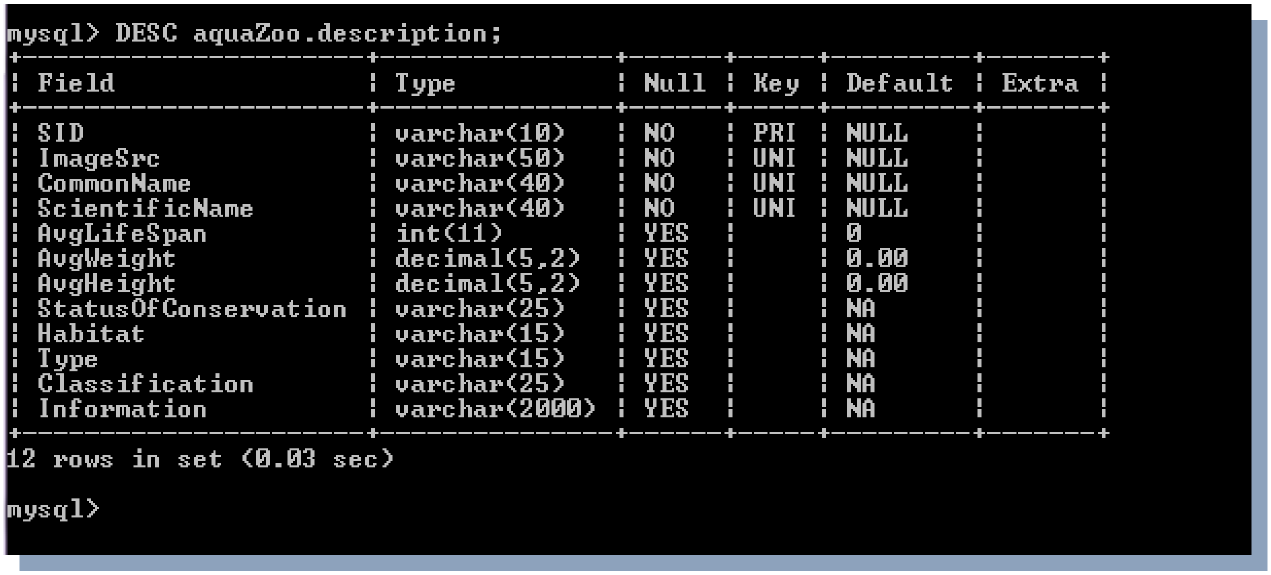
Figure 4. Sponsor Table with Primary SPID, AID and Foreign Key AID.



```
 CREATE TABLE aquaZoo.medicalhistory (
  RecordNo varchar(10) NOT NULL UNIQUE,
  Disease varchar(20) DEFAULT 'NA',
  Date date DEFAULT '1970-01-01',
  DIET varchar(30) DEFAULT 'NA',
  TID varchar(10) NOT NULL,
  AID varchar(10) NOT NULL,

  PRIMARY KEY (RecordNo),
  CONSTRAINT TIDTRAINORMEDICALHISTORY FOREIGN KEY (TID) REFERENCES aquaZoo.trainer (TID) ON UPDATE CASCADE,
  CONSTRAINT AIDTRAINORMEDICALHISTORY FOREIGN KEY (AID) REFERENCES aquaZoo.animal (AID) ON UPDATE CASCADE
);
```

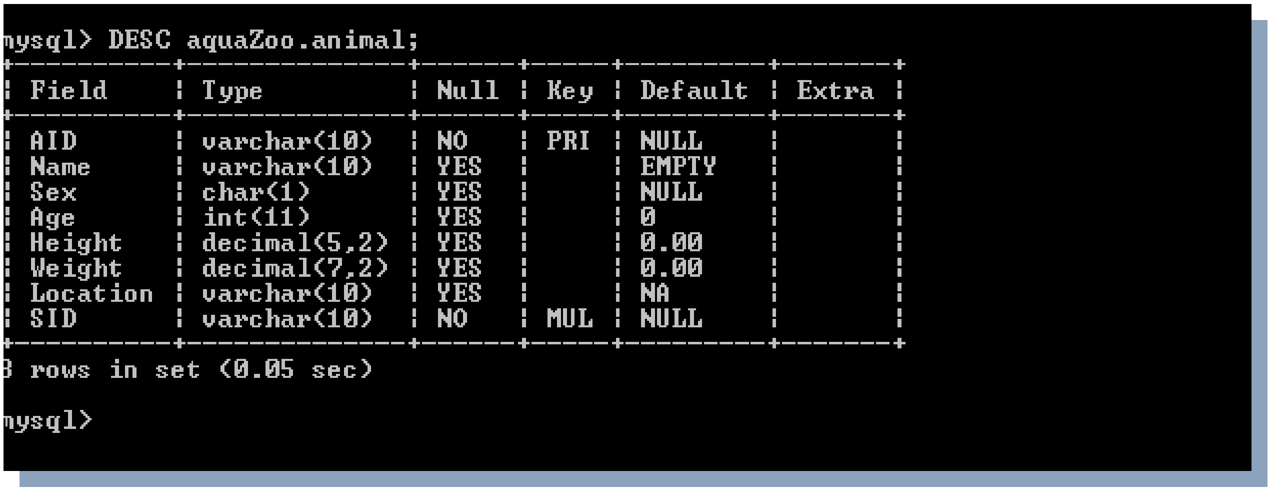
Figure 5. Medical History Table with Primary Key RecordNo and Foreign Keys TID and AID.

2.10.2. TABLE Relation Information



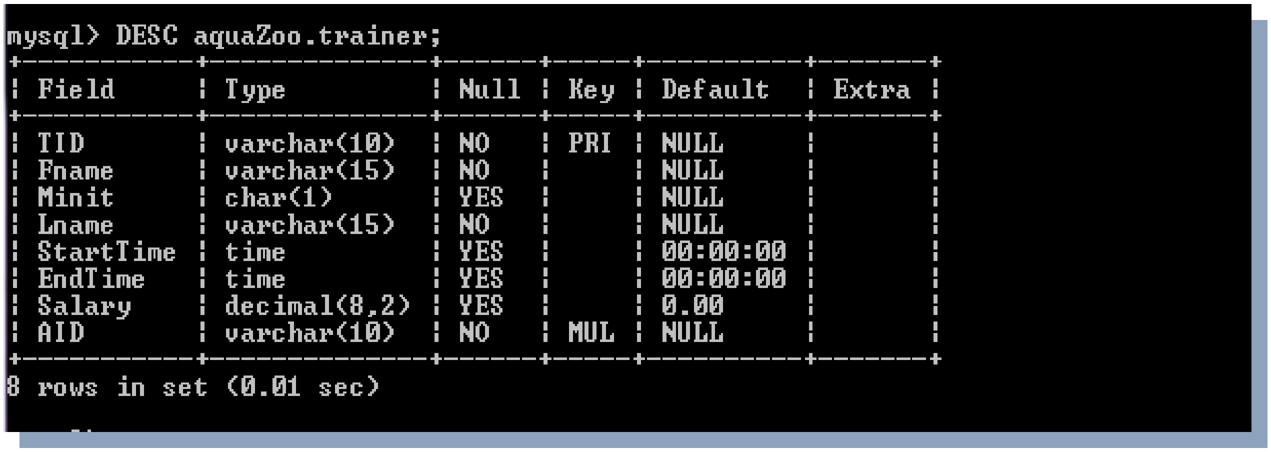
```
mysql> DESC aquaZoo.description;
+----------------------+----------------+------+-----+----------+--------+
| Field                | Type           | Null | Key | Default  | Extra  |
+----------------------+----------------+------+-----+----------+--------+
| SID                  | varchar(10)    | NO   | PRI | NULL     |        |
| ImageSrc             | varchar(50)    | NO   | UNI | NULL     |        |
| CommonName           | varchar(40)    | NO   | UNI | NULL     |        |
| ScientificName       | varchar(40)    | NO   | UNI | NULL     |        |
| AvgLifeSpan          | int(11)        | YES  |     | 0        |        |
| AvgWeight            | decimal(5,2)   | YES  |     | 0.00     |        |
| AvgHeight            | decimal(5,2)   | YES  |     | 0.00     |        |
| StatusOfConservation | varchar(25)    | YES  |     | NA       |        |
| Habitat              | varchar(15)    | YES  |     | NA       |        |
| Type                 | varchar(15)    | YES  |     | NA       |        |
| Classification       | varchar(25)    | YES  |     | NA       |        |
| Information          | varchar(2000)  | YES  |     | NA       |        |
+----------------------+----------------+------+-----+----------+--------+
12 rows in set (0.03 sec)

mysql>
```

Figure 8. Description Table Information.



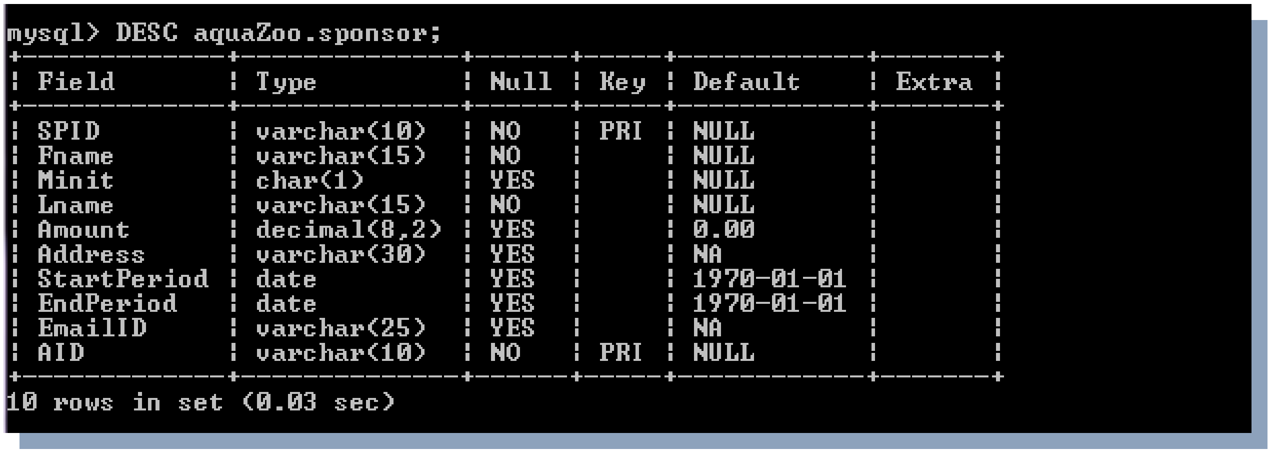
```
mysql> DESC aquaZoo.animal;
+----------+--------------+------+-----+----------+--------+
| Field    | Type         | Null | Key | Default  | Extra  |
+----------+--------------+------+-----+----------+--------+
| AID      | varchar(10)  | NO   | PRI | NULL     |        |
| Name     | varchar(10)  | YES  |     | EMPTY    |        |
| Sex      | char(1)      | YES  |     | NULL     |        |
| Age      | int(11)      | YES  |     | 0        |        |
| Height   | decimal(5,2) | YES  |     | 0.00     |        |
| Weight   | decimal(7,2) | YES  |     | 0.00     |        |
| Location | varchar(10)  | YES  |     | NA       |        |
| SID      | varchar(10)  | NO   | MUL | NULL     |        |
+----------+--------------+------+-----+----------+--------+
8 rows in set (0.05 sec)

mysql>
```

Figure 9. Animal Table Information.



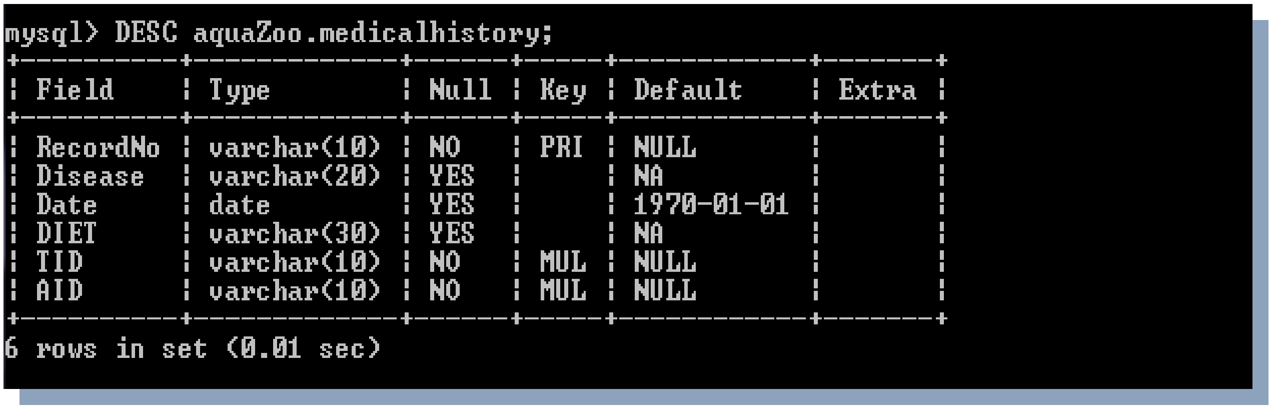
```
mysql> DESC aquaZoo.trainer;
+-----------+--------------+------+-----+----------+-------+
| Field     | Type         | Null | Key | Default  | Extra |
+-----------+--------------+------+-----+----------+-------+
| TID       | varchar(10)  | NO   | PRI | NULL     |       |
| Fname     | varchar(15)  | NO   |     | NULL     |       |
| Minit     | char(1)      | YES  |     | NULL     |       |
| Lname     | varchar(15)  | NO   |     | NULL     |       |
| StartTime | time         | YES  |     | 00:00:00 |       |
| EndTime   | time         | YES  |     | 00:00:00 |       |
| Salary    | decimal(8,2) | YES  |     | 0.00     |       |
| AID       | varchar(10)  | NO   | MUL | NULL     |       |
+-----------+--------------+------+-----+----------+-------+
8 rows in set (0.01 sec)
```

Figure 10. Trainer Table Information.



```
mysql> DESC aquaZoo.sponsor;
+-------------+--------------+------+-----+------------+-------+
| Field       | Type         | Null | Key | Default    | Extra |
+-------------+--------------+------+-----+------------+-------+
| SPID        | varchar(10)  | NO   | PRI | NULL       |       |
| Fname       | varchar(15)  | NO   |     | NULL       |       |
| Minit       | char(1)      | YES  |     | NULL       |       |
| Lname       | varchar(15)  | NO   |     | NULL       |       |
| Amount      | decimal(8,2) | YES  |     | 0.00       |       |
| Address     | varchar(30)  | YES  |     | NA         |       |
| StartPeriod | date         | YES  |     | 1970-01-01 |       |
| EndPeriod   | date         | YES  |     | 1970-01-01 |       |
| EmailID     | varchar(25)  | YES  |     | NA         |       |
| AID         | varchar(10)  | NO   | PRI | NULL       |       |
+-------------+--------------+------+-----+------------+-------+
10 rows in set (0.03 sec)
```

Figure 11. Sponsor Table Information.



```
mysql> DESC aquaZoo.medicalhistory;
+----------+-------------+------+-----+------------+-------+
| Field    | Type        | Null | Key | Default    | Extra |
+----------+-------------+------+-----+------------+-------+
| RecordNo | varchar(10) | NO   | PRI | NULL       |       |
| Disease  | varchar(20) | YES  |     | NA         |       |
| Date     | date        | YES  |     | 1970-01-01 |       |
| DIET     | varchar(30) | YES  |     | NA         |       |
| TID      | varchar(10) | NO   | MUL | NULL       |       |
| AID      | varchar(10) | NO   | MUL | NULL       |       |
+----------+-------------+------+-----+------------+-------+
6 rows in set (0.01 sec)
```

Figure 12. CPU Component Table Information.

3. Architectural Design

3.1. System Architectural Design

The Aqua-Zoo Management Database has been implemented using a cloud-based architecture. The group utilized the online cloud based development service called “Cloud9”.

The backend database system has been implemented utilizing the MySQL feature of the “Cloud9” development service. The frontend access to the database has been developed using HTML, PHP, and CSS in order to provide the most essential features on a web browser. The Aqua-Zoo Management system currently resides in the “Cloud9” development service.

3.2. System Architectural General Issues

- Our Aqua-Zoo Management System is designed to be a multi-user application. Aqua-Zoo Administrators and regular users that exist in the database can access it. A regular user can utilize the online website that connects to the database to retrieve the various animals present in the Aqua-Zoo. They can also view the details of each animal and the location of that particular animal in the Aqua-Zoo. Aqua-Zoo Administrators are allowed to utilize the online website to add, update, delete, and view trainer, sponsor information, animals, animal description, and the medical history of the animal.
- The architecture is designed to allow the Aqua-Zoo Administrators to read and write entries into the database. General users will only be able to read the information of the animals present in the Aqua-Zoo.
- Indexing is currently not handled by our implementation of our Aqua-Zoo Management Database. As the data in our database grows, this issue will be addressed in order to increase efficiency of our database system.
- Concurrency is currently not handled by our implementation of our Aqua-Zoo Management Database. We are aware of the issues that can arise with a multi-user application and maintaining concurrent data. This issue will be addressed in future iterations of our project.
- Currently, the database was populated with a small set of realistic data. This was completed for in class demonstration purposes. Our database is capable of being expanded with additional data entries.
- A transaction failure is currently not handled by our implementation of our Aqua-Zoo Management Database. We determined that a transaction failure is not a major issue for our database application. If a transaction fails, then System Administrators can reenter the data into the database.
- Our implementation of the backend MySQL database server and front-end website is based on the cloud development environment, called Cloud9. This allows for high mobility for use on mobile laptops, tablets, and phones by utilizing a web browser. This is also helpful in order to develop an App for iOS and Android platforms in the future. Our implementation of the database resides on the server side within the cloud9 MySQL cloud database.
- We utilized the drivers provided by MySQL to connect to the backend database server and utilized the PHP and HTML to develop the frontend website.

3.3. System Architecture Diagram

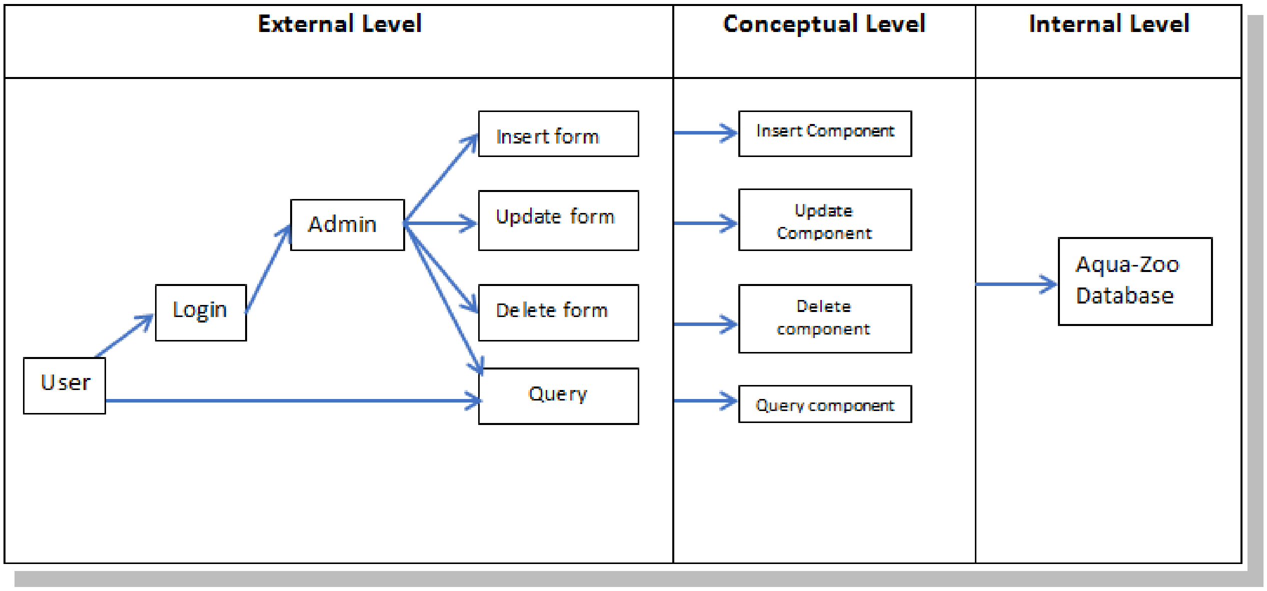


Figure 13. Architecture Drawing of External Level, Conceptual Level, and Internal Level

3.4. Roles of Architecture Components and Interactions

- Users are users of the Aqua-Zoo Management database.
- The Login Component is utilized to validate the users of the Aqua-Zoo Management System. This component will also determine if the user is a member of the Aqua-Zoo Administrators or a regular user.
- The Administrator Component is utilized by the administrator in the Management Group to insert, update, delete, and retrieve information from the database.
- The User Component is utilized by regular users for viewing the details of the Aqua-Zoo, various animals and their description, that are present in the Aqua-Zoo.
- The Insert Form Component is designed to allow the admin of the Aqua-Zoo to insert details of the animals, their description, trainers available, sponsorship for animals and the medical history of the animals.
- The Update Form Component is designed to allow the admin of the Aqua-Zoo to update the details of the animals, their description, trainers available, sponsorship for animals and the medical history of the animals.
- The Delete Form Component is designed to allow admin of the Aqua-Zoo to delete the details of the animals, their description, trainers available, sponsorship for animals and the medical history of the animals.
- The Query Component is designed to allow admin of the Aqua-Zoo to query the database of the Aqua-Zoo to retrieve the desired information about the animals, their description, details about the present trainers and sponsors and medical history of the animals. In case of a normal user, it will allow the user to query the

database to retrieve the desired information about animal description and sponsor details.

- The Insert Component is designed to submit the insert data request to the backend database.
- The Update Component is designed to submit the updated data request to the backend database.
- The Delete Component is designed to submit the delete data request to the backend database.
- The Aqua-Zoo Database Component is the backend MySQL database implementation that contains the database utilized by the Aqua-Zoo Management System.

4. Team Progress

4.1. Project Plan

We will be setting up weekly deadlines to complete the following task:

- The group plans to complete the ER diagrams by October 20.
- The database schema will be completed by October 22.
- A draft of the formal description document will be completed by October 22 for a formal technical review and submission of the formal description on October 24.
- The user interface will be completed by November 4.
- A draft of the final report document will be completed by November 14 for a formal technical review and submission of the final report on November 16.
- A draft of the database demonstration will be completed by November 14 for a review before giving the in class demonstration on November 16.

4.2. Responsibilities

- Aadil Ahmed Adam will do the login portion; will create forms for the insertion of animals, their description, the trainer and sponsor insertion forms and design of the database tables.
- Akash Rajendra Ventekar is the team leader and will work on the back-end of the application providing the ability to update and delete the information about the Animals, their Description, Sponsors and Trainers.
- Poornima Byre Gowda will take care of the insertion and querying of the information present in the database after login.
- Jeevan Chandrashekar will work on the creation of the forms for all the update and deletion of the data in the Animals, their Description, Sponsors and Trainers entities.

4.3. Major Contribution

- Aadil Ahmed adam focused on mapping the ER Diagram to the Relational Schema and ensuring the tables had the correct attributes and constraints. Also, Aadil created the login and Insertion view of the web-interface
- Jeevan Chandrashekar worked on setting up the team meetings and maintaining the material developed in the meetings. Also, Jeevan created the Update and Deletion views of the web-interface.
- Akash Rajendra Ventekar was the team leader and focused on the updation and deletion of data using the web-interface. Additionally, Akash focused on the testing of the web-based interface. He also worked on the authorization of the users to the web–interface.
- Poornima Byre Gowda focused on setting up the cloud9 implementation and focused on the querying of the database and was also involved in the design and testing of the web- interface.

4.4. Major Issues with Design and Implementation of System

The ternary relation between Animal, Trainer and Medical History was a major issue that had to be solved in our original database design. Originally we were attempting to relate the Animal entity with Medical History entity. However, after discovering issues with attempting to find the cardinality and participation constraints, we determined that a ternary relationship would solve the problem. The team discussed and came to a conclusion that a ternary relationship between Animal, Trainer and Medical History would solve the problem of determining cardinality and participation constraints between these three entities.

We had a few issues with determining the weak entities of our database design. There was confusion within the group concerning which entities should be considered as a weak entity and identified by a strong entity. We resolved this issue by examining our ER diagram and looking at examples in our Fundamentals of Database Systems book. Additionally, the team had decided to work on the implementation of the database using MongoDB and Node.js environment for the development of the database. The team then realized that it would not be feasible within the given span of time and based on the skills of all the team members, we determined to have the implementation using MongoDB has advanced features of this project.

4.5. Advanced Features of Aqua-Zoo Management Database

4.5.1. Cloud9 Implementation

Our cloud9 implementation can be accessed at:
http://www.c9.io
Username: pg0018@uah.edu
Password: databaseproject

The web-site interface can be accessed at:
https://test-123-poori19.c9users.io/?_c9_id=livepreview0&_c9_host=https://ide.c9.io

Default Sample System Administrators:
Username: pg0018
Password: password

Username: Akash
Password: password

4.5.2. Cloud-based MySQL Implementation

We choose to utilize the MySQL database that runs on Cloud9 for our backend implementation of our system. This was chosen to allow the database run on a remote system. Having the database in a remote cloud allowed the team to access the same database while working on the project individually and then combining their sections into one application. This improved the efficiency of the team working and brings together the individual sections that each member worked on for the front-end and middle layer of our system.

Having the database on the server side allows for one database to be utilized by multiple users of a system or users of a company. This helps with the synchronization of data that multiple users may access and provide consistency within the data.

This was chosen over the Amazon Web Services because of its availability and free accessibility, as well as for the development and deployment of the front-end access to the database.

4.5.3. Web-based Access to Database

Cloud9 provided the ability to develop the front-end and middle layers of our architecture. We utilized HTML and PHP programming code to develop web pages to validate admin, show forms to the admin in order to insert, update, delete, and retrieve information from the database.

We chose to utilize a cloud9 for our implementation in order to gain access to the web-based interface utilizing a web browser and an internet connection. This provided the ability to easy access our database during our in class demonstration.

4.5.4. Referential MongoDB Implementation

As an extension of the project, we have implemented the same logic on Mongo data base and Node js.

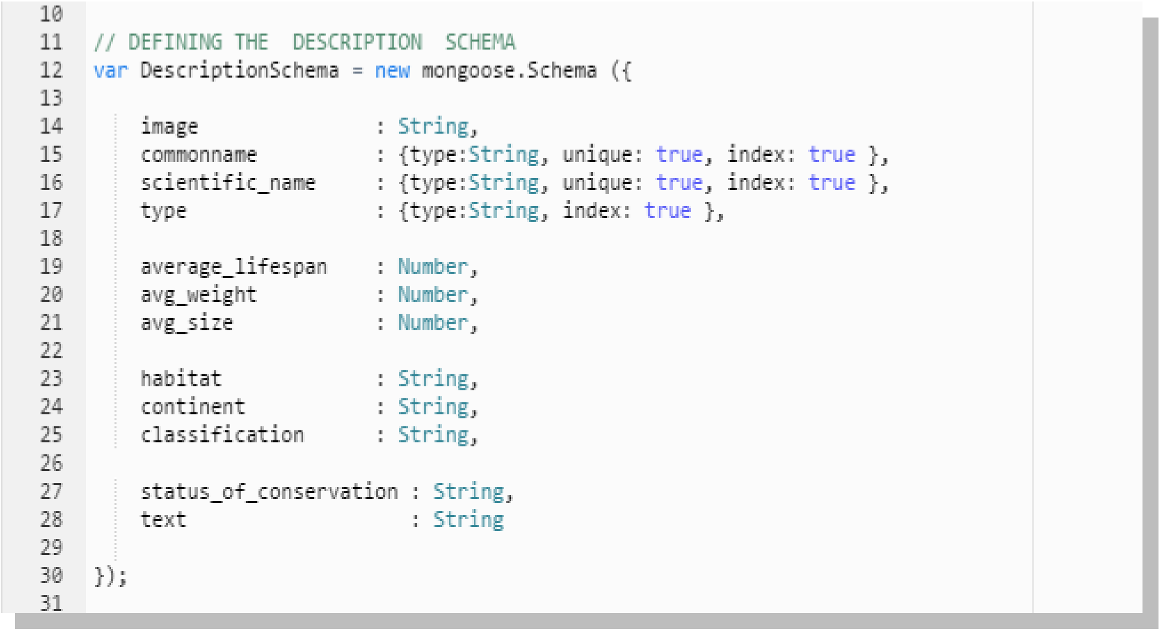
Mongo DB is a NoSQL database program. MongoDB uses JSON-like documents with schemas. Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient. Node.js package ecosystem, NPM, is the largest ecosystem of open source libraries. NPM allows you to install/publish node packages.

NPM package holds basic two package dependencies.

1. ExpressJS framework is used to run on Node.js. It allows you to build web applications and API endpoints.
2. Mongoose is an NPM package that allows you to interact with MongoDB.

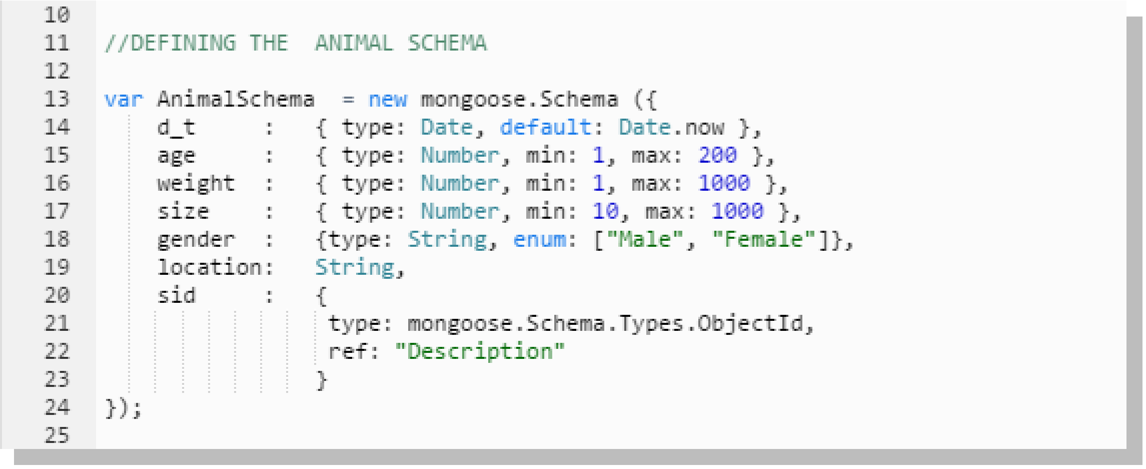
Create, read, update and delete (CRUD) data from MongoDB is done using Mongoose/Node. CRUD functionality is implemented through the following routes. POST for create, GET for read, PUT for update, DELETE for delete.

The following below figures depict the creation of the various schemas in our database on Mongo dB:



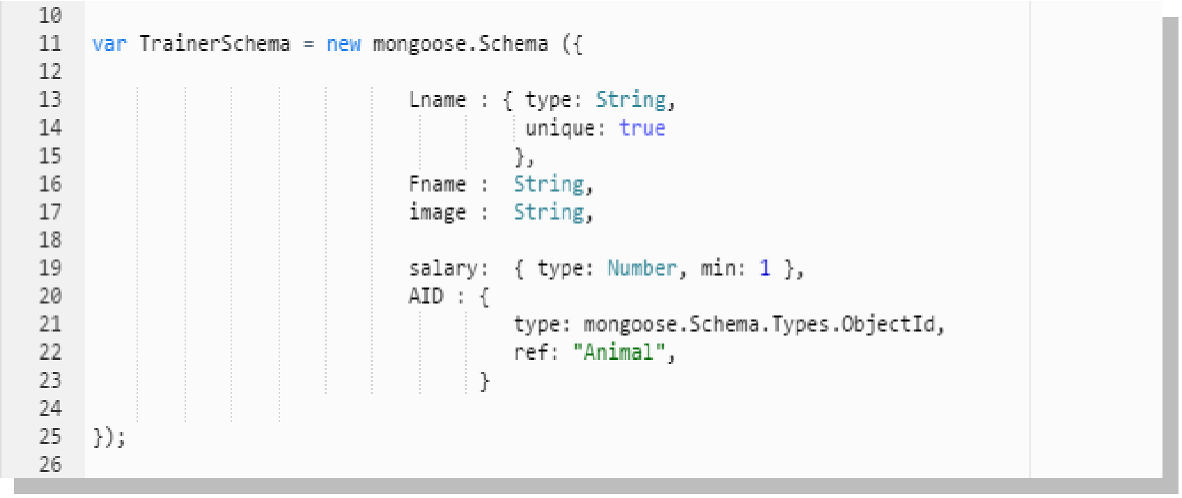
```

// DEFINING THE  DESCRIPTION  SCHEMA
var DescriptionSchema = new mongoose.Schema ({

    image                 : String,
    commonname            : {type:String, unique: true, index: true },
    scientific_name       : {type:String, unique: true, index: true },
    type                  : {type:String, index: true },

    average_lifespan      : Number,
    avg_weight            : Number,
    avg_size              : Number,

    habitat               : String,
    continent             : String,
    classification        : String,

    status_of_conservation : String,
    text                   : String

});

```

Figure 13. Description Schema with Primary Key SID.



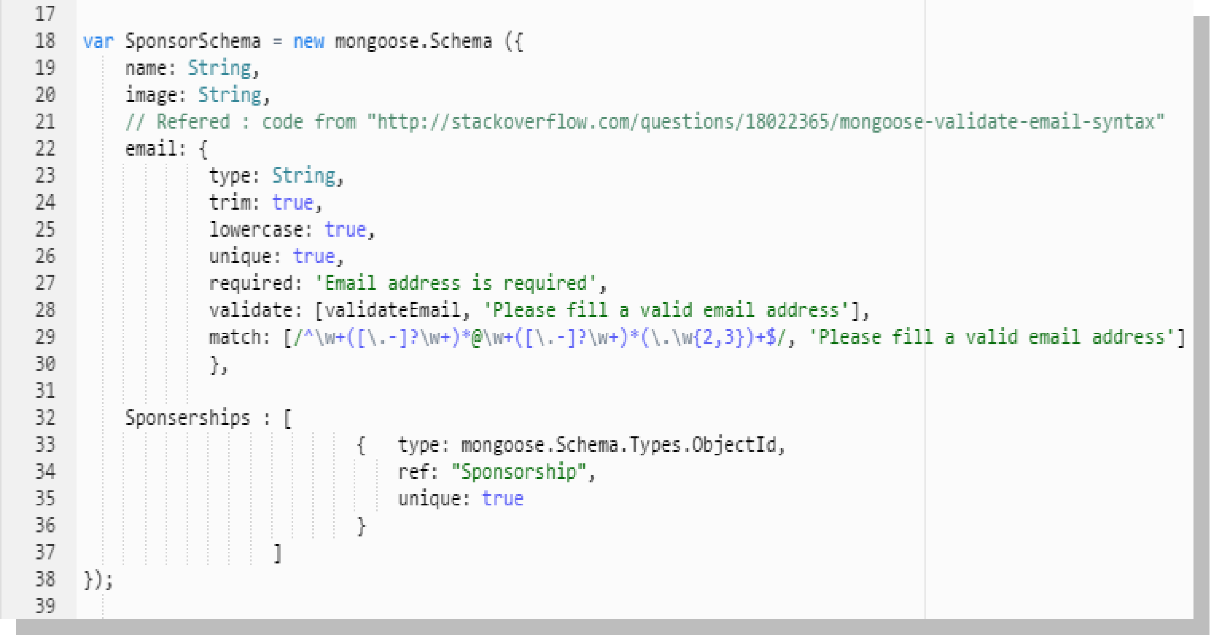
```

//DEFINING THE  ANIMAL SCHEMA

var AnimalSchema  = new mongoose.Schema ({
    d_t      :   { type: Date, default: Date.now },
    age      :   { type: Number, min: 1, max: 200 },
    weight   :   { type: Number, min: 1, max: 1000 },
    size     :   { type: Number, min: 10, max: 1000 },
    gender   :   {type: String, enum: ["Male", "Female"]},
    location:    String,
    sid      :   {
                  type: mongoose.Schema.Types.ObjectId,
                  ref: "Description"
                 }
});

```

Figure 14. Animal Schema with Primary Key AID



```

var TrainerSchema = new mongoose.Schema ({

                          Lname : { type: String,
                                    unique: true
                                  },
                          Fname :  String,
                          image :  String,

                          salary:  { type: Number, min: 1 },
                          AID : {
                                   type: mongoose.Schema.Types.ObjectId,
                                   ref: "Animal",
                                }

});

```

Figure 15. Trainer Schema with Primary Key TID



```

var SponsorSchema = new mongoose.Schema ({
    name: String,
    image: String,
    // Refered : code from "http://stackoverflow.com/questions/18022365/mongoose-validate-email-syntax"
    email: {
            type: String,
            trim: true,
            lowercase: true,
            unique: true,
            required: 'Email address is required',
            validate: [validateEmail, 'Please fill a valid email address'],
            match: [/^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/, 'Please fill a valid email address']
            },

    Sponserships : [
                        {   type: mongoose.Schema.Types.ObjectId,
                            ref: "Sponsorship",
                            unique: true
                        }
                    ]
});

```

Figure 16. Sponsor Schema with Primary Key SPID

4.5.5. Web-based Snapshots of User Interface

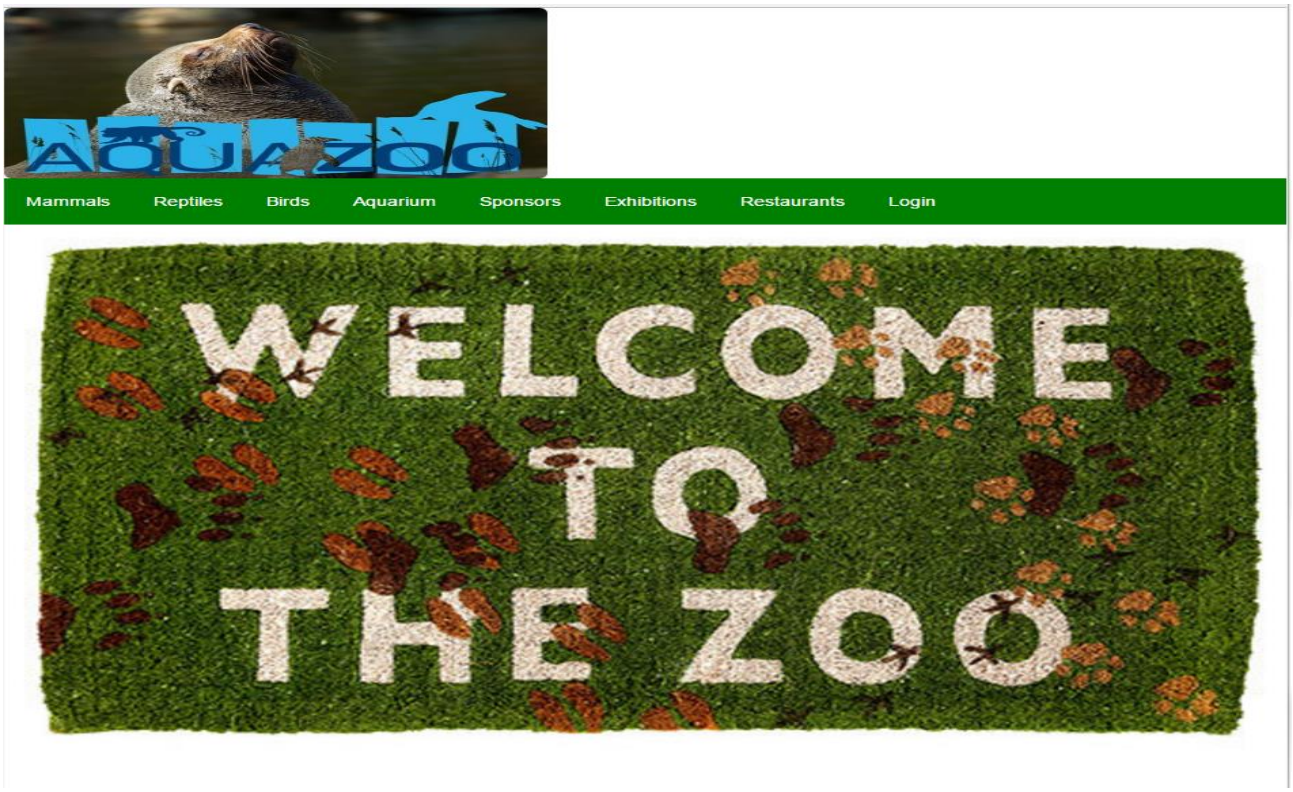


Figure 17. Homepage of the Aqua-Zoo website designed to allow users to view the user interface of the zoo.

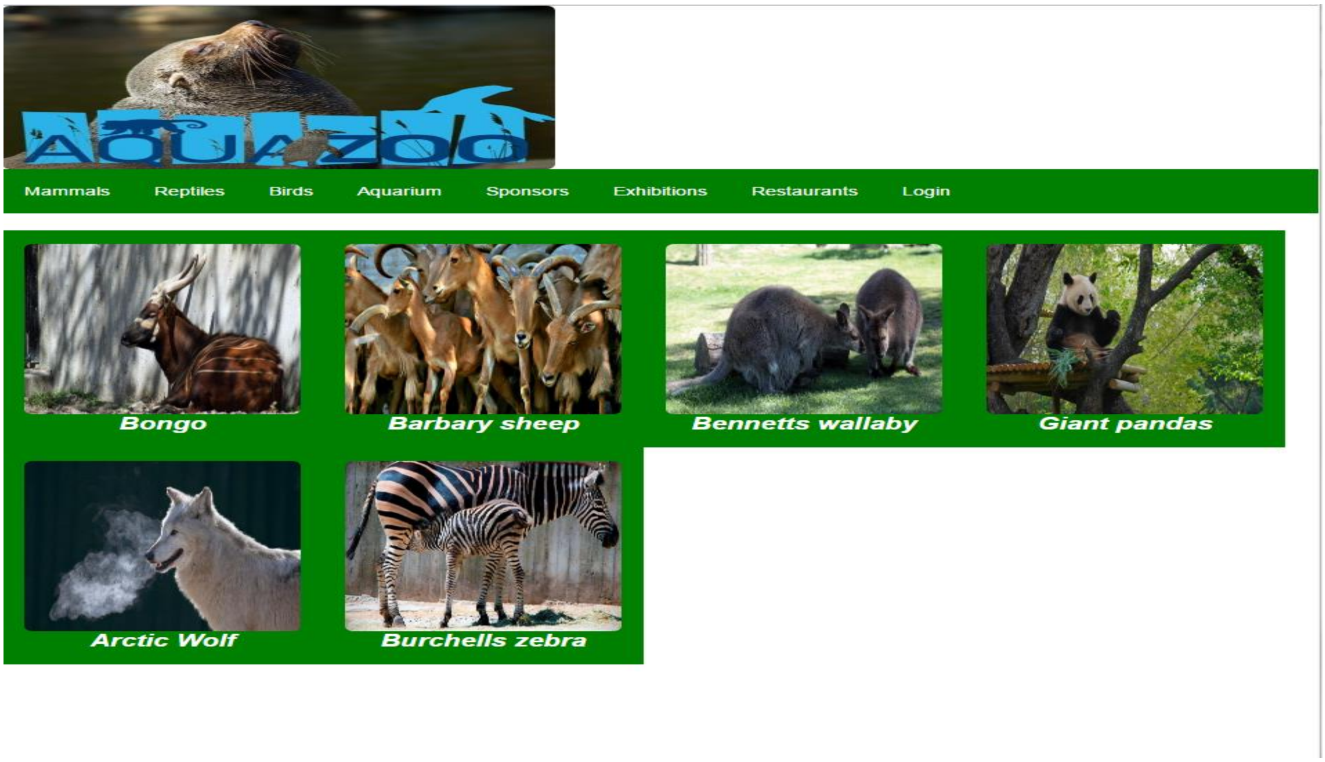


Figure 18. Regular user page shows animals classified as "mammals" when the user clicks on the Mammals tab.

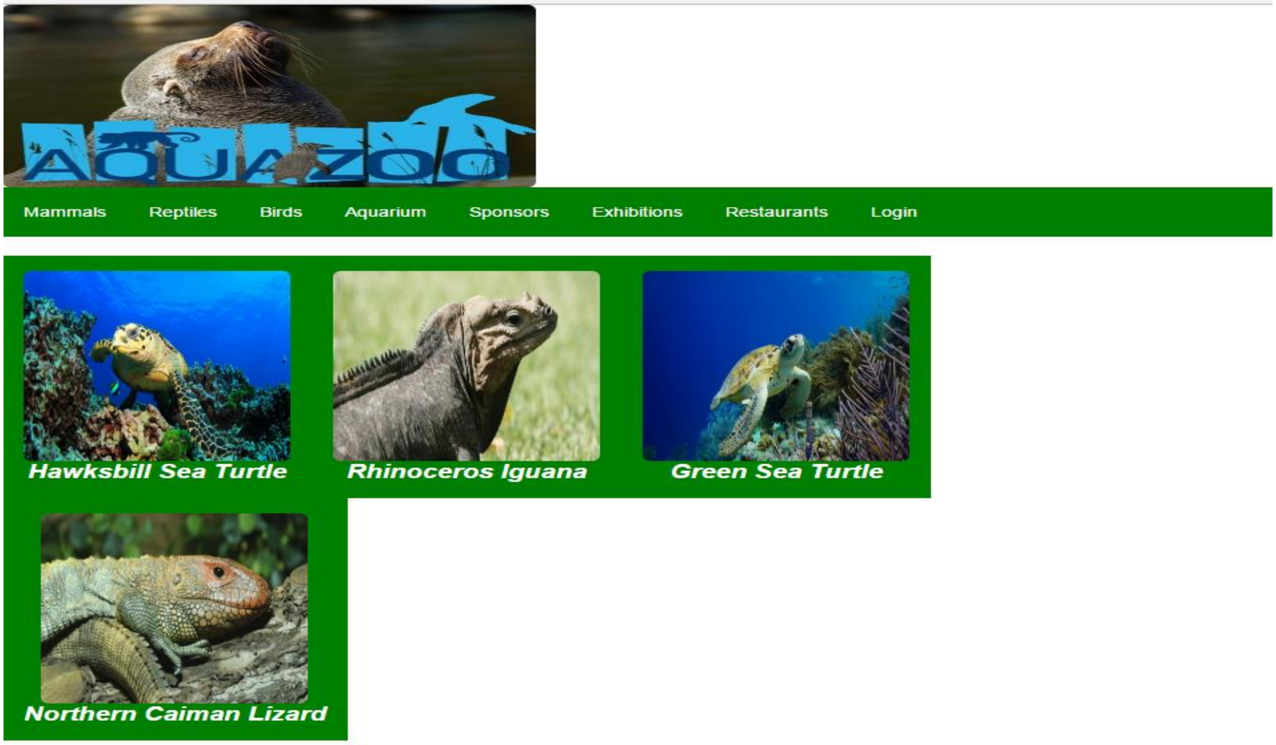


Figure 19. Regular user page shows animals classified as "reptiles" when the user clicks on the Reptiles tab.

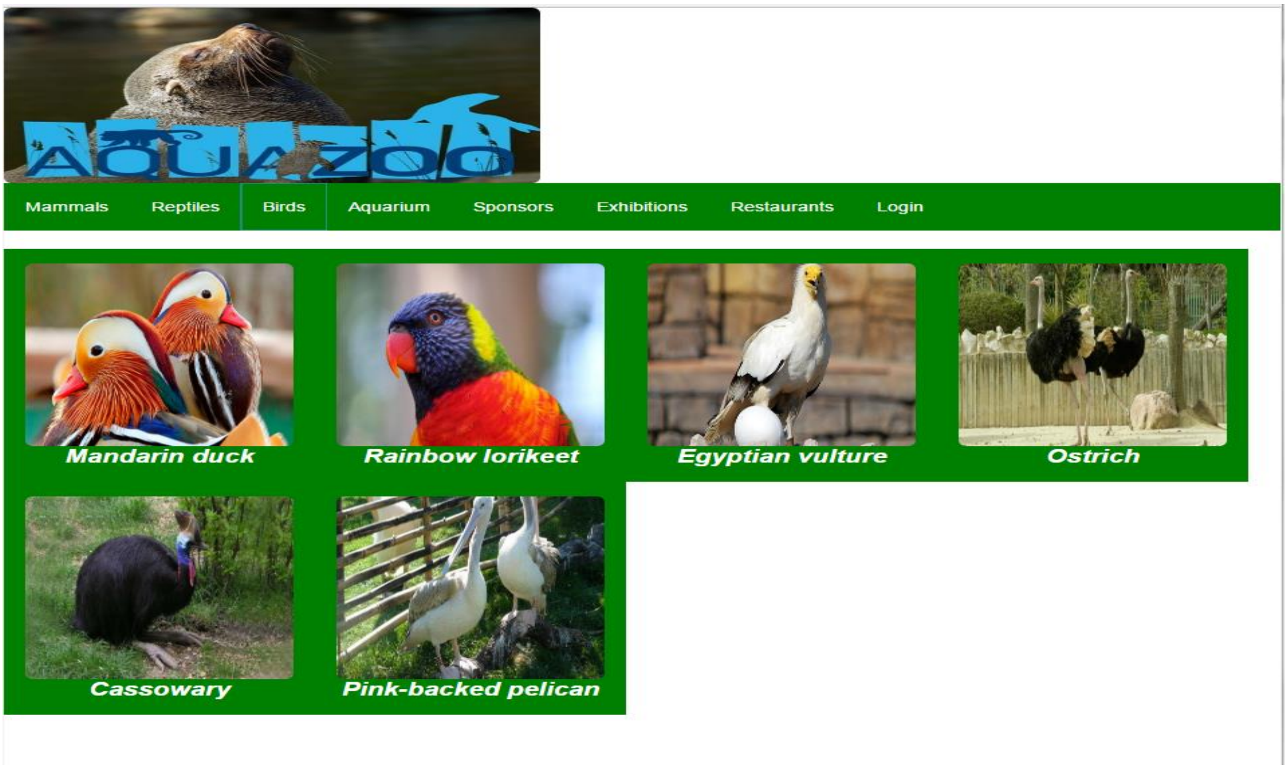


Figure 20. Regular user page shows animals classified as "birds" when the user clicks on the Birds tab.

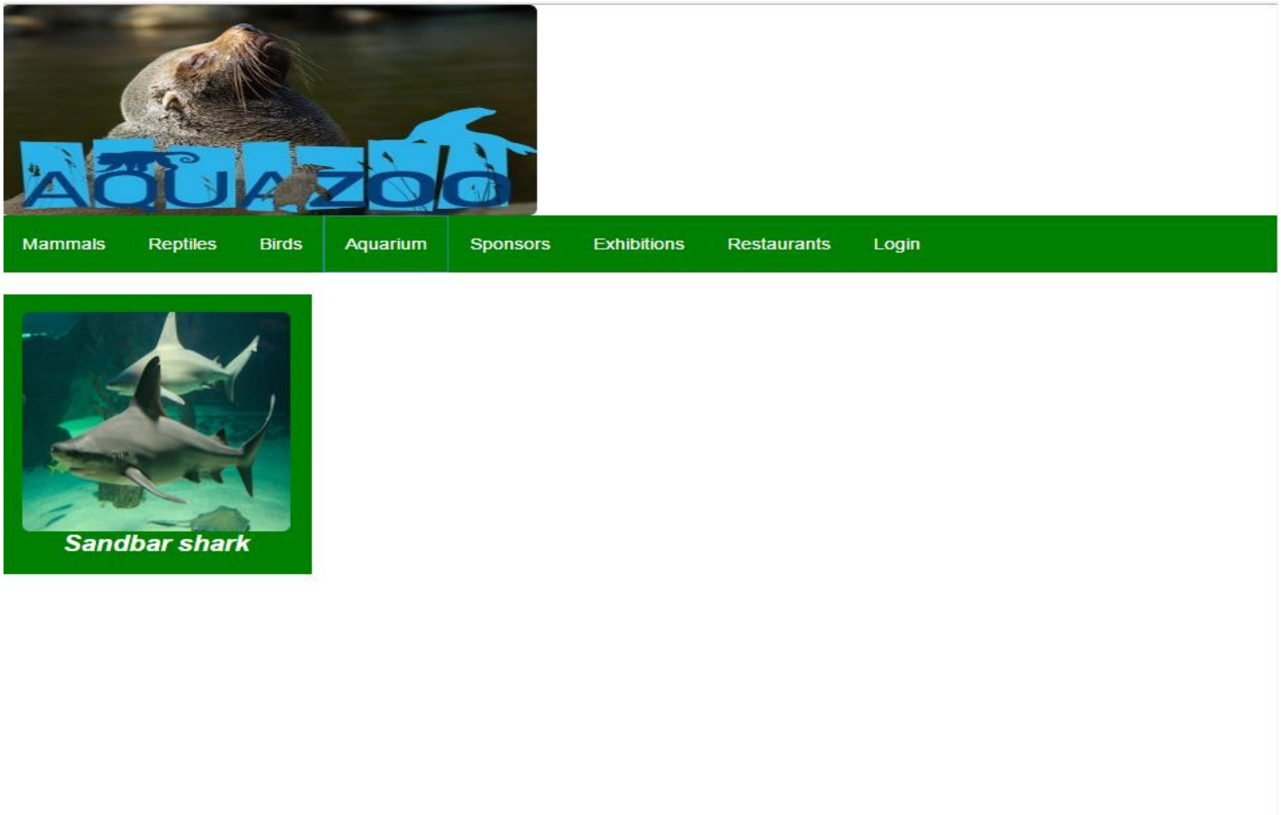


Figure 21. Regular user page shows animals classified as "fish" when the user clicks on the Aquarium tab.

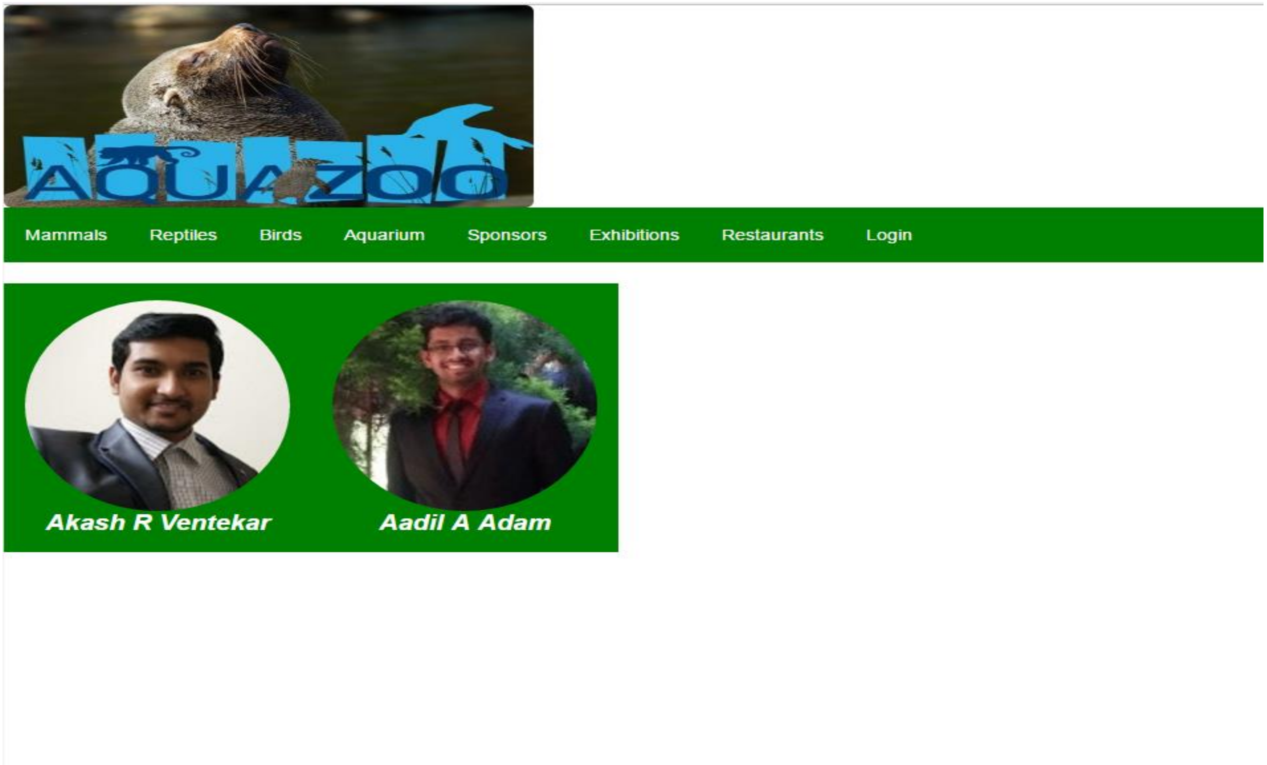


Figure 22. Regular user page shows the details about various sponsors sponsoring animals in the zoo when the user clicks on the sponsor tab.

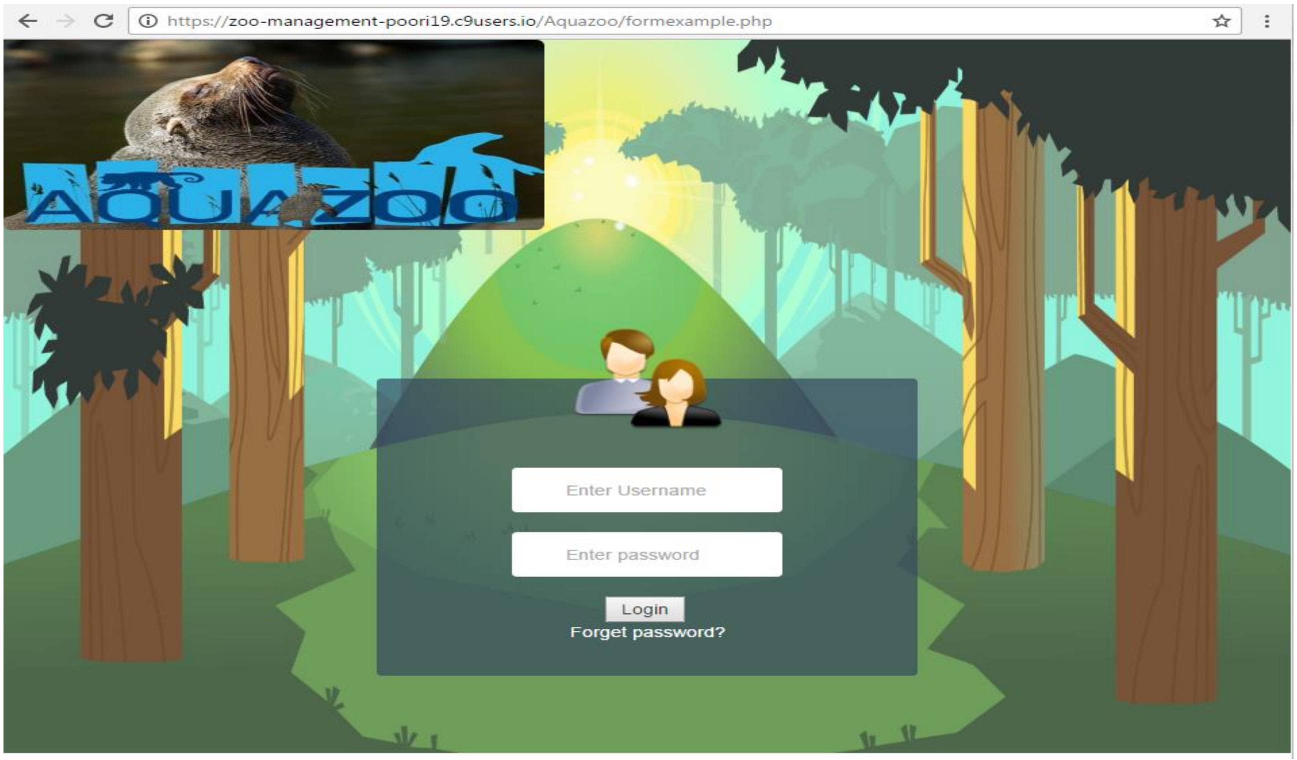
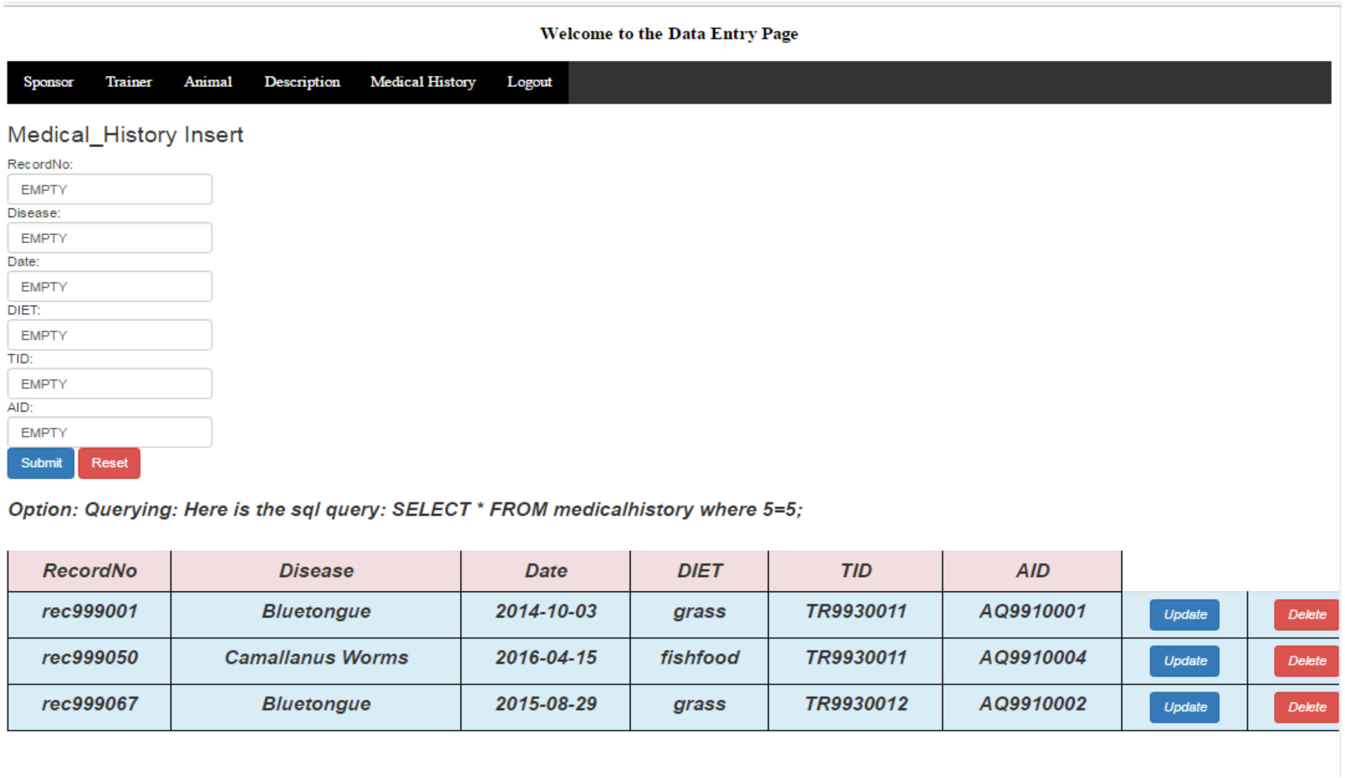

https://zoo-management-poori19.c9users.io/Aquazoo/formexample.php
AQUA ZOO
Enter Username
Enter password
Login
Forget password?

Figure 6. Login component which allows the admin of the zoo to maintain the database after providing the correct credentials.



Welcome to the Data Entry Page

Medical_History Insert

RecordNo: EMPTY

Disease: EMPTY

Date: EMPTY

DIET: EMPTY

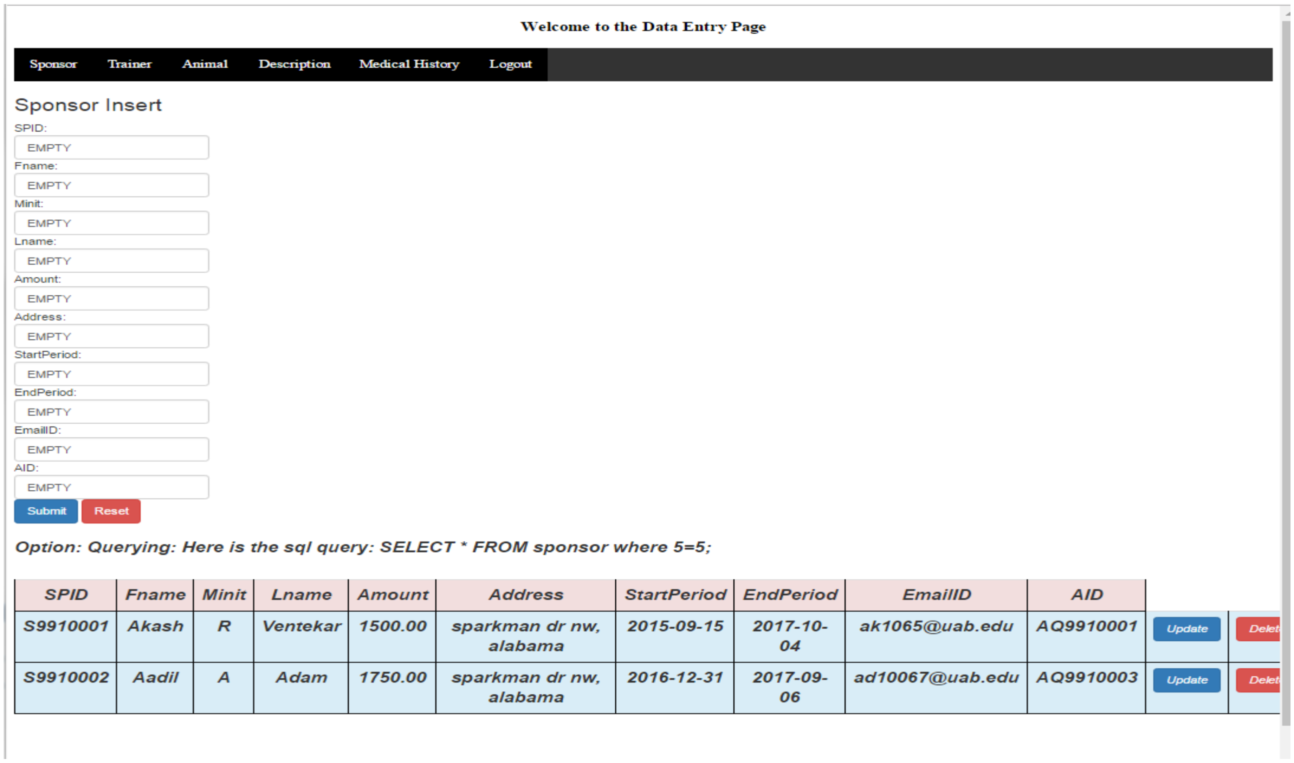
TID: EMPTY

AID: EMPTY

*Option: Querying: Here is the sql query: SELECT * FROM medicalhistory where 5=5;*

RecordNo	Disease	Date	DIET	TID	AID		
rec999001	Bluetongue	2014-10-03	grass	TR9930011	AQ9910001	Update	Delete
rec999050	Camallanus Worms	2016-04-15	fishfood	TR9930011	AQ9910004	Update	Delete
rec999067	Bluetongue	2015-08-29	grass	TR9930012	AQ9910002	Update	Delete

Figure 24. Insert page for the Medical History Component where the admin is allowed to insert, update or delete the information regarding the medical history of animals.



Welcome to the Data Entry Page

Sponsor Insert

SPID: EMPTY

Fname: EMPTY

Minit: EMPTY

Lname: EMPTY

Amount: EMPTY

Address: EMPTY

StartPeriod: EMPTY

EndPeriod: EMPTY

EmailID: EMPTY

AID: EMPTY

*Option: Querying: Here is the sql query: SELECT * FROM sponsor where 5=5;*

SPID	Fname	Minit	Lname	Amount	Address	StartPeriod	EndPeriod	EmailID	AID		
S9910001	Akash	R	Ventekar	1500.00	sparkman dr nw, alabama	2015-09-15	2017-10-04	ak1065@uab.edu	AQ9910001	Update	Delet
S9910002	Aadil	A	Adam	1750.00	sparkman dr nw, alabama	2016-12-31	2017-09-06	ad10067@uab.edu	AQ9910003	Update	Delet

Figure 25. Insert page for the Sponsor Component where the admin is allowed to insert, update or delete the information regarding the sponsors.

4.5.6. Bonus Points

We implemented both a web-based access feature to our system and cloud based infrastructure service for our database implementation. This allows for easy integration of group members work and allows our software to be easily accessible from any systems. This allowed the team to access the same database within the cloud9 environment.

We also implemented the database using the referential MongoDB concept and the application was developed using the interface and middleware scripting platform called as Node.js. Node.js is a rapidly growing technology that has been overtaking the world of server-side programming with surprising speed. MongoDB is a technology that’s revolutionizing database usage. Together, the two tools are a potent combination**.** They both employ JavaScript and JSON.

4.6. Meetings Log

All meetings are occurring in the CS building in the Computer Lab located in room 329. If the lab is unavailable, then a special note of the location where the meeting was held will be indicated with that specific meeting entry.

Date:	08/18/2016
Members:	All Members
Start Time:	3:00 PM
Duration:	20 Minutes
Description:	The group members began communication on working together to develop a common topic for the project. The group members formally agreed to work on the project together. The list of topics was narrowed down to two topics of interest. One topic was a Zoo Management database and the other topic was a Tourism Management database.

Date:	08/21/2016
Members:	All Members
Start Time:	4.00 PM
Duration:	30 Minutes
Description:	The group members decided to design the outline of the Zoo management database. Each member programming skills and database knowledge was also discussed to determine what roles each member might have for the project.

Date:	08/24/2016
Members:	All Members
Start Time:	3:30 PM
Duration:	1 Hours and 20 Minutes
Description:	We discussed the various entities which will be part of the database. We discussed about the important attributes which will be unique and need to be taken care of. The relationship between the various entities was also discussed. It was decided that the initial draft would be developed over the weekend and that team member's ideals would be shared using Google GMail and Google Drive.

Date:	09/06/2016
Members:	All Members
Start Time:	6:00 PM
Duration:	30 Minutes
Description:	The group prepared an informal description of the topic discussed and submitted in class.

Date:	09/13/2016
Members:	All Members
Start Time:	6:20 PM
Duration:	1 Hours and 10 Minutes
Description:	The group had a formal review of the topic description. The topic description (version 1.1) was prepared and shared on GMail. The group discussed the various entities and their attributes which would be part of our database. The relationship between the entities was also discussed. The group prepared the informal description. The group also discussed the possible languages that could be used to implement the database.

Date:	09/14/2016
Members:	All Members
Start Time:	12:40 PM
Duration:	5 Minutes
Description:	The team submitted the informal description. This completes version 2.0 of the document.

Date:	09/15/2016
Members:	All Members
Start Time:	12:30 PM
Duration:	1 Hours and 20 Minutes
Description:	We discussed the various entities which will be part of our database. The tables and the various columns of each table were discussed. The group also refined the possible roles and skills of different team members. The roles of different team members were formally indicated in the document.

Date:	09/26/2016
Members:	All Members
Start Time:	2:00 PM
Duration:	1 Hour 20 Minutes
Description:	The entities, attributes, the data types were formulated into a document and shared on Google EMail. The group also prepared an outline of the look of the user interface and the various features which should be part of our user interface. The group also formulated an outline of the ER diagram based on the various entities discussed.

Date:	10/07/2016
Members:	All Members

Start Time:	3:15 PM
Duration:	30 Minutes
Description:	We discussed about the various tools we need to use for the development of the project. The skills and roles of each team members were discussed. The group installed MySQL WorkBench 6.3 on all laptops and decided to learn all the Node.js and other required skills. Initial version of the tables were created and tested with some dummy entries.

Date:	10/13/2016
Members:	All Members
Start Time:	4:00 PM
Duration:	3 Hours
Description:	We reviewed over our initial entities design and further developed the attributes for each entity. We decide on the Primary key/ Foreign key which would be part of our database. The group decided to create a draft of the ER Diagram. The cardinalities and constraints between entities were developed in this meeting. We were not for sure if the “Medical History” entity should be used to link Animal entity only or be linked to Trainer entity also. The group also discussed about the implementation of the database using Mongo DB.

Date:	10/23/2016
Members:	All Members
Start Time:	2:00 PM
Duration:	1 Hour and 30 Minutes
Description:	The group determined that we need a ternary relationship to link Medical History with Animals and Trainer. The ER diagram was modified to indicate that there exists a Ternary relationship in the database. We reviewed over the ER diagram and determined some flaws in the assignment of attributes between entities and relationship type and was corrected. The mapping of ER diagram to Relational Schema Model was done. A draft of the Relational Schema drawing was documented. The

	formal description was the database with the ER diagram and Relational Schema was prepared and shared on Google EMail and was ready to be submitted in class on October 24th. This completes our tasks for the formal description of our database (version 303).

Date:	10/24/2016
Members:	All Members
Start Time:	12:40 PM
Duration:	5 Minutes
Description:	The formal description was reviewed and submitted. This completes Version 3.0.

Date:	11/01/2016
Members:	All Members
Start Time:	7:00 PM
Duration:	4 Hours
Description:	The group met to discuss the implementation of the database. The group examined in more detail the cloud based development and database service called “Cloud9”. A SQL script with the ability to create the database and tables utilizing MySQL. There were no significant modifications necessary to our database based upon feedback from our last deliverable. The group also discussed about the creation of the schema on Mongo DB and creation of various forms required for insert and updating the data.

Date:	11/05/2016
Members:	All Members
Start Time:	2:20 PM
Duration:	1 Hour and 30 Minutes
Description:	The group continued the discussion on the script that was started to create the database and tables. The script was expanded to include adding initial

	entries into the database. The group decided that the “Cloud9” environment would work for implementation and web interface of our database. An initial prototype of the web interface utilizing HTML and PHP was created to demonstrate the ability to connect the front-end web browser to the back-end MySQL database implementation.

Date:	11/07/2016
Members:	All Members
Start Time:	3:00 PM
Duration:	1 Hour
Description:	We continued our discussion of the implementation of the front-end web browser service for our database. The design of the web pages is ongoing. The back-end implementation of the database utilizing the MySQL service of “Cloud9” has been completed.

Date:	11/08/2016
Members:	All Members
Start Time:	10:00 AM
Duration:	2 Hours
Description:	The group met to discuss the front-end web-interface for accessing our database that runs in the cloud9. The group decided to utilize HTML and PHP in order to provide forms to admin and create SQL statements to submit to the database. Each member decided to work on a specific part (insert, update, delete, retrieve) of the programming of the front-end.

Date:	11/13/2016
Members:	All Members
Start Time:	9:00 PM
Duration:	2 Hours and 15 Minutes
Description:	The group met to discuss the front-end web-interface based on the work done by each team member. The work was cumulated and a draft web

	interface was done.
Date:	11/15/2016
Members:	All Members
Start Time:	5:00 PM
Duration:	6 Hours and 15 Minutes
Description:	We finalized the cloud9 implementation of our front-end web-based interface and our MySQL back-end database. The web interface was up and running with all the major implementations. The group also had some of the Mongo DB implementation working with some features. The group created a short PowerPoint presentation detailing the major design elements of our system for the in-class demonstration.

Date:	11/16/2016
Members:	All Members
Start Time:	12:30 PM
Duration:	12 Hours and 30 Minutes
Description:	Preparing the final draft for submission. We completed the System Architecture, Major Issues, and Advanced Features sections of the document. We updated all diagrams and screenshots of our system. The group performed a final review over the final report before submission.