Pet Adoption Center Management System

CIS 9340

Submitted By

Anna Green : annaloi.green@baruchmail.cuny.edu
Clement Huang : clement.huang@baruchmail.cuny.edu
Jielian Ma : jielian.ma@baruchmail.cuny.edu
Michelle Tsvitman : michelle.tsvitman@baruchmail.cuny.edu
Nikita Gautam : Nikita.gautam@baruchmail.cuny.edu
Rezwan Islam : rezwan.islam@baruchmail.cuny.edu

Date: Dec 16, 2024

TABLE OF CONTENTS

Pet Adoption Center

Executive Summary

The challenge for Pet Adoption Centers is the lack of a centralized system to manage the huge amount of data required for day-to-day operations. This data includes animals available for adoption, their medical records, adopters and staff. The decentralized system leads to inefficiencies in caring for animals, screening adoption applications and matching animals with friendly homes.

Our goal is to design a database system to efficiently manage and streamline the operations of the Pet Adoption Center. As well as keeping searching for potential adopters and ensuring proper veterinary care is provided. This new system will store information on all the animals that are available for adoption, track their medical history and manage the adoption applications in order to match the best pet with the right family. We believe this system will support the business to streamline day-to-day operations, minimize data entry errors and increase the adoption rates.

Business Scenario

The Pet Adoption Center is a non profit organization whose main goal is to find loving homes for animals in need. However, the overall management of the organization can get a little tricky or complex at times. The center has to oversee a wealth of information including the animals in its care, their medical histories, potential adopters, foster homes, and the staff involved in day-to-day operations. Without a centralized system, it will be quite difficult to make optimal use of this adoption center. In order to overcome this issue, the center has decided to implement a comprehensive database system which will be described in detail moving forward.

The database's main drivers are the Animal identities and the adoption process. Each animal is classified as either a dog or a cat, with specific attributes tracked for each type, such as breed and behavioral traits. Animals are placed in foster homes until they are adopted, with each foster home having a unique ID, location, contact information, and capacity. The center needs to determine the foster home's availability to make sure that no animal is left without care.
As mentioned earlier, the adoption process is another crucial aspect of the center's work. Families interested in adopting must go through a process to ensure the best match for both the animal and the adopter. Managing these applications will require careful coordination, from scheduling meetings between adopters and animals to processing application approvals or denials, and this is where the database will come in handy.

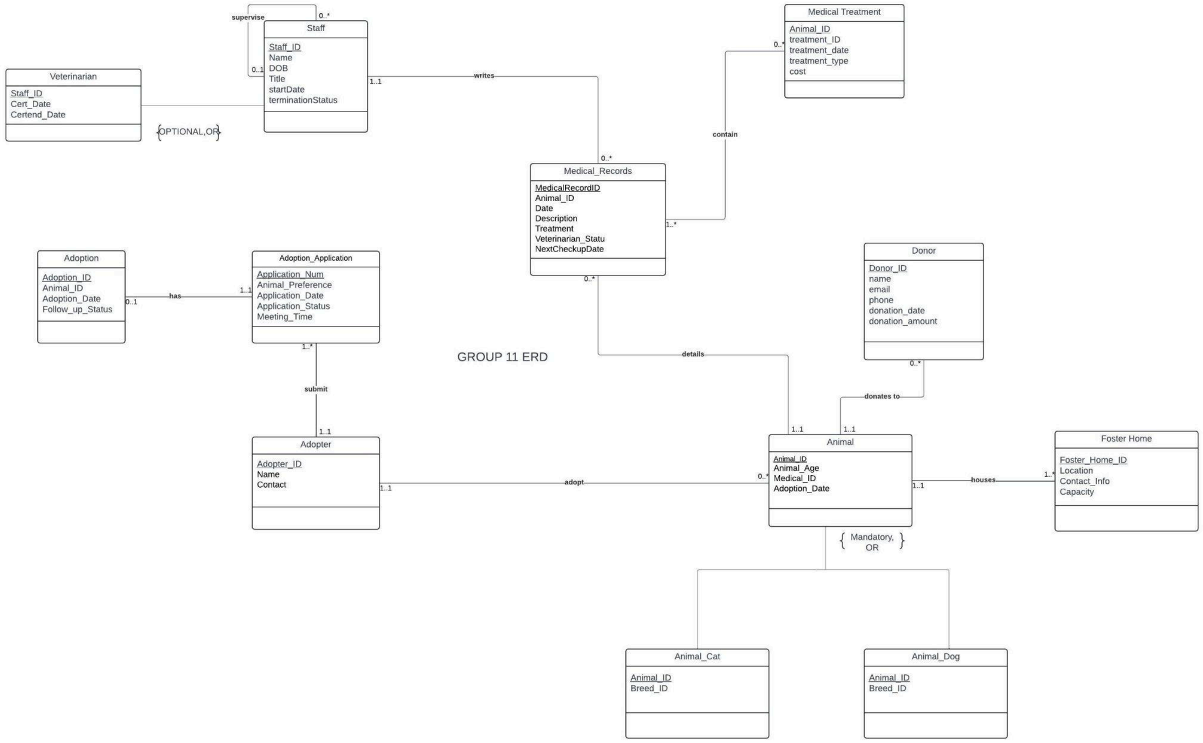
Another key priority for the center is to provide timely and effective medical care for all animals. From routine checkups to critical treatments, maintaining accurate and up-to-date medical records is essential. This center also relies on the support of the donors to maintain its operations since it is a non profit. So, it's crucial to keep a good track record of their contributions for the continued success of the adoption center.

Finally, all the daily operations in the adoption center are handled by the staff so a database system to effectively manage staff scheduling and task management is important to ensure the day to day operations run smoothly. With all these scenarios in mind, the Adoption Center will implement a database system that will streamline the process and ensure data accuracy. This approach will provide better care for the animals, the adopters and increase their current adoption rates.

Information to Be Tracked

- Medical Records and Treatments for each animal. The center tracks medical records including dates, descriptions, and treatment statuses to maintain comprehensive health histories for each animal. As shown in the Medical_Records and Medical_Treatment tables, each treatment has an associated cost and staff member, which is crucial for a non-profit organization to monitor expenses and ensure appropriate veterinary care. The NextCheckup field helps staff stay on top of follow-up care requirements.
- Adoption Process Status Through the Adoption and Adoption_Application tables. The center monitors the entire adoption journey. Each application is tracked with dates and status (Approved/Pending/Rejected), along with the type of animal desired (Dog/Cat/Rabbit/Bird). Meeting times are scheduled and recorded to help manage staff time and facilitate adopter-animal introductions. The status progression from application to completed adoption helps ensure no potential adoption falls through the cracks.
- Foster Home Information. The Foster_Home table maintains critical data about temporary care locations including geographic location (city), curator contact information, and the capacity of each home. This information is essential for managing animal placement, as shown by the capacity numbers ranging from 5-12 animals per location. The curator contact information ensures clear communication channels with those providing temporary care.
- Staff and Veterinarian Information Through the Staff and Veterinarian tables. The center tracks all personnel details including roles (Senior Vet, Junior Vet, Vet Assistant), start dates, and active/inactive status. Certification tracking ensures all veterinary staff maintain proper credentials, with certification expiration dates clearly recorded. This information is crucial for assigning appropriate staff to medical procedures and maintaining quality of care.

ER Model Using UML Notation



Relationship Sentences and Analysis

- **Animal - Veterinary_Record**

Relationship Sentence: One Animal can have many Veterinary_Records, but each Veterinary_Record belongs to only one Animal.

Multiplicity: 1..1 to 1..*

- **Animal - Medical_Treatment**

Relationship Sentence: One Animal can have multiple Medical_Treatments, but each Medical_Treatment is associated with only one Animal.

Multiplicity: 1..1 to 1..*

- **Animal - Adoption_Application**

Relationship Sentence: One Animal can have multiple Adoption_Applications, but each Adoption_Application is linked to only one Animal.

Multiplicity: 1..1 to 1..*

- **Adopter - Adoption_Application**

Relationship Sentence: One Adopter can submit multiple Adoption_Applications, but each Adoption_Application is submitted by only one Adopter.

Multiplicity: 1..1 to 1..*

- **Adoption_Application - Animal & Adopter**

Relationship Sentence: Each Adoption_Application is linked to one Animal and one Adopter.

Multiplicity: 1..1 to 1..1

- **Staff - Supervisor (Recursive Relationship)**

Relationship Sentence: One Staff member can supervise multiple Staff members, but each Staff member reports to only one Supervisor.

Multiplicity: 1..1 to 1..*

- **Adoption - Animal & Adopter**

Relationship Sentence: Each Adoption is a final record linking one Animal and one Adopter.

Multiplicity: 1..1 to 1..1

- **Donor - Animal**

Relationship Sentence: One animal can have multiple Donors, but each Donor can have zero to one animal.

Multiplicity: 1..1 to 1..*

- **Foster_Home - Animal**

Relationship Sentence: One Foster_Home can have one animal, but each Animal belongs to many Foster_Home.

Multiplicity: 1..* to 1..1

●

Multiplicity: 1..1 to 1..*

Converting the ERD to a Relational Model

Animal_Cat(Animal_ID, Animal_Age, Medical_ID, Adoption_Date, Breed_ID, AdopterID(FK), Foster_Home_ID(FK))

Animal_Dog(Animal_ID, Animal_Age, Medical_ID, Adoption_Date, Breed_ID, AdopterID(FK), Foster_Home_ID(FK))

Foster_Home (Foster_Home_ID, Location, Curator_Info, Capacity, Animal_ID(FK))

Donor(Donor_ID, name, email, phone, donation_date, donation_amount, Animal_ID(FK))

Animal_Donor(Animal_ID, Donor_ID)

Medical Records (MedicalRecordID, Date, Description, Treatment, Veterinarian_Status, NextCheckupDate, Staff_ID (FK)Animal_ID (FK))

Medical_Treatment_Records(Animal_ID, MedicalRecordID)

Medical_Treatment(Animal_ID, treatment_ID, treatment_date, treatment_type, cost)

Staff(Staff_ID, Name, DOB, Title, Start_Date, Termination_Status)

Supervisor(Sup_ID, Staff_ID(fk))

Veterinarian(Staff_ID, DOB, Title, Start_Date, Termination_Status, Cert_Date, Certend_Date)

Staff_Adoption_application(Staff_ID, Application_Num)

Adoption application(Application_Num, Animal_Preference, Application_Date, Application_Status, Meeting_Time, Adopter_ID(fk))

Adoption (Adoption_ID, Animal_ID, Adoption_Date, Follow_up_Status,Application_Num(fk))

Adopter(Adopter_ID, Name, Contact)

Normalization

Entity 1: Animal

- **Attributes**: Animal_ID, Animal_Age, Medical_ID, Adoption_Date, AdopterID(FK), Foster_Home_ID(FK)
    - **Step 1**: Do we have a key? Yes, Animal_ID is a key - in 1NF.

Functional dependencies:

FD1: Animal_ID-> Animal_Age, Medical_ID, Adoption_Date, Adopter_ID (FK), Foster_Home_ID (FK)

- **Step 2**: Do we have partial key dependency? No partial key dependency - 2NF.
- **Step 3**: Do we have transitive dependency? No transitive dependency - 3NF.
- **Step 4**: Any determinant that is not a candidate key? No - BCNF.

Entity 2: Animal_Dog

- **Attributes**: Animal_ID, Animal_Age, Medical_ID, Adoption_Date, Breed_ID, size, AdopterID(FK), Foster_Home_ID(FK)

FD1: Animal_ID -> Animal_Age, Medical_ID, Adoption_Date, Breed_ID, AdopterID(FK), Foster_Home_ID(FK)

- **Step 1**: Do we have a key? Yes, Animal_ID is a key - in 1NF.
- **Step 2**: Do we have partial key dependency? No partial key dependency - 2NF.
- **Step 3**: Do we have transitive dependency? No transitive dependency - 3NF.
- **Step 4**: Any determinant that is not a candidate key? No - BCNF.

Entity 3: Animal_Cat

- **Attributes**: Animal_ID, Animal_Age, Medical_ID, Adoption_Date, Breed_ID, AdopterID(FK), Foster_Home_ID(FK)
    - **Step 1**: Do we have a key? Yes, Animal_ID is a key - in 1NF.

Functional dependencies:
FD1: Animal_ID -> Animal_Age, Medical_ID, Adoption_Date, Breed_ID, AdopterID(FK), Foster_Home_ID(FK)

- **Step 2**: Do we have partial key dependency? No partial key dependency - 2NF.
- **Step 3**: Do we have transitive dependency? No transitive dependency - 3NF.
- **Step 4**: Any determinant that is not a candidate key? No - BCNF.

Entity 4: Foster_Home

- **Attributes**: Foster_Home_ID, Location, Curator_Info, Capacity, Animal_ID (FK)
    - **Step 1**: Do we have a key? Yes, Foster_Home_ID is a key - in 1NF.

      FD1: Foster_Home_ID -> Location, Curator_Info, Capacity, Animal_ID(FK)
    - **Step 2**: Do we have partial key dependency? No partial key dependency - 2NF.
    - **Step 3**: Do we have transitive dependency? No transitive dependency - 3NF.
    - **Step 4**: Any determinant that is not a candidate key? No - BCNF.

Entity 5: Adopter

- **Attributes**: Adopter_ID, Name, Contact_Info
    - **Step 1**: Do we have a key? Yes, Adopter_ID is a key - in 1NF.

      FD1: Adopter_ID -> Name, Contact_Info
    - **Step 2**: Do we have partial key dependency? No partial key dependency - 2NF.
    - **Step 3**: Do we have transitive dependency? No transitive dependency - 3NF.
    - **Step 4**: Any determinant that is not a candidate key? No - BCNF.

Entity 6: Medical Records

- **Attributes**: MedicalRecordID, Date, Description, Treatment, Veterinarian_Status, NextCheckupDate, Veterinarian_ID
    - **Step 1**: Do we have a key? Yes, MedicalRecordID is a key - in 1NF.

Functional dependencies:

FD1: MedicalRecordID -> Animal_ID, Date, Description, Treatment, Veterinarian_Status, NextCheckupDate, Veterinarian_ID

- **Step 2**: Do we have partial key dependency? No partial key dependency - 2NF.
- **Step 3**: Do we have transitive dependency? No transitive dependency - 3NF.
- **Step 4**: Any determinant that is not a candidate key? No - BCNF.

Entity 7: Adoption Application

- **Attributes**: Application_Num, Animal_Preference, Application_Date, Application_Status, Meeting_Time
    - **Step 1**: Do we have a key? Yes, Application_Num is a key - in 1NF.

      FD1: Application_Num -> Animal_Preference, Application_Date, Application_Status, Meeting_Time
    - **Step 2**: Do we have partial key dependency? No partial key dependency - 2NF.
    - **Step 3**: Do we have transitive dependency? No transitive dependency - 3NF.
    - **Step 4**: Any determinant that is not a candidate key? No - BCNF.

Entity 8: Adoption

- **Attributes**: Adoption_ID, Animal_ID, Adoption_Date, Follow_up_Status
    - **Step 1**: Do we have a key? Yes, Adoption_ID is a key - in 1NF.

      FD1: Adoption_ID -> Animal_ID ,Adoption_Date, Follow_up_Status
    - **Step 2**: Do we have partial key dependency? No partial key dependency - 2NF.
    - **Step 3**: Do we have transitive dependency? No transitive dependency - 3NF.
    - **Step 4**: Any determinant that is not a candidate key? No - BCNF.

Entity 9: Staff

- **Attributes**: Staff_ID, Name, DOB, Title, Start_Date, Termination_Status
    - **Step 1**: Do we have a key? Yes, Staff_ID is a key - in 1NF.

      FD1: Staff_ID-> Name, DOB, Title, Start_Date, Termination_Status
    - **Step 2**: Do we have partial key dependency? No partial key dependency - 2NF.
    - **Step 3**: Do we have transitive dependency? No transitive dependency - 3NF.
    - **Step 4**: Any determinant that is not a candidate key? No - BCNF.

Entity 10: Supervisor

- **Attributes**: Sup_ID, Staff_ID(FK)
    - **Step 1**: Do we have a key? Yes, Sup_ID is a key - in 1NF.

    FD1: Sup_ID-> Staff_ID

- **Step 2**: Do we have partial key dependency? No partial key dependency - 2NF.
- **Step 3**: Do we have transitive dependency? No transitive dependency - 3NF.
- **Step 4**: Any determinant that is not a candidate key? No - BCNF.

Entity 11: Veterinarian

- **Attributes**: Staff_ID, DOB, Title, Start_Date, Termination_Status, Cert_Date, Certend_Date
  - **Step 1**: Do we have a key? Yes, Staff_ID is a key - in 1NF.
  - FD1: Staff_ID-> Staff_ID
  - **Step 2**: Do we have partial key dependency? No partial key dependency - 2NF.
  - **Step 3**: Do we have transitive dependency? No transitive dependency - 3NF.
  - **Step 4**: Any determinant that is not a candidate key? No - BCNF.

Entity 12: Donor

- **Attributes**: Donor_ID, name, email, phone, donation_date, donation_amount, Animal_ID
  - **Step 1**: Do we have a key? Yes, Donor_ID is a key - in 1NF.
    FD1: email -> name, phone
  - **Step 2**: Do we have partial key dependency? No partial key dependency - 2NF.
  - **Step 3**: Do we have transitive dependency? Yes, email -> name, phone. Split into:
    - R1 (email, name, phone)
    - R2 (Donor_ID, email, donation_date, donation_amount, Animal_ID) - 3NF.
  - **Step 4**: Any determinant that is not a candidate key? No - BCNF.

Entity 13: Medical_Treatment

- **Attributes**: Animal_ID, treatment_ID, treatment_date, treatment_type, cost
  - **Step 1**: Do we have a key? Yes, treatment_ID is a key - in 1NF.

    FD1: treatment_ID → treatment_type, cost
  - **Step 2**: Do we have partial key dependency? No partial key dependency - 2NF.

- **Step 3**: Do we have transitive dependency? Yes, treatment_ID -> treatment_type, cost. Split into:
    - R1 (treatment_ID, treatment_type, cost)
    - R2 (Animal_ID, treatment_ID, treatment_date) - 3NF.
- **Step 4**: Any determinant that is not a candidate key? No - BCNF.

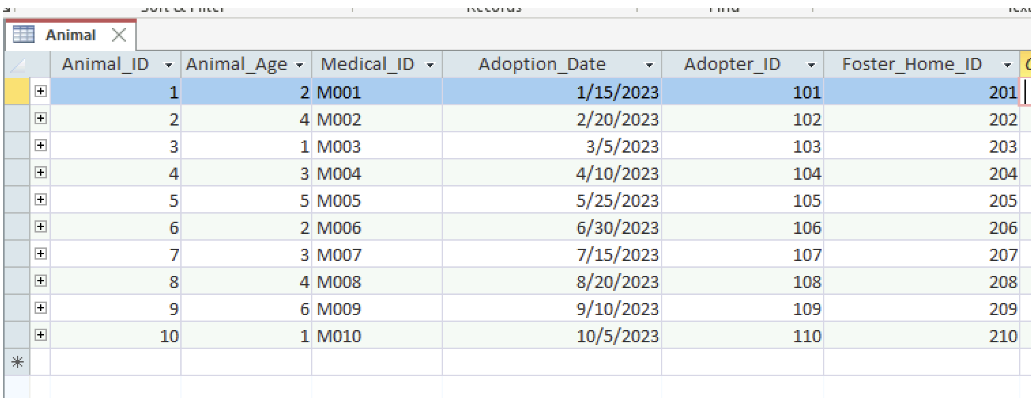
Creating Tables with SQL

Animal

```
CREATE TABLE Animal (
 Animal_ID NUMBER NOT NULL,
   Animal_Age NUMBER,
   Medical_ID VARCHAR(50),
   Adoption_Date DATE,
   AdopterID NUMBER,
   Foster_Home_ID NUMBER,
CONSTRAINT pk_Animal PRIMARY KEY (Animal_ID),

CONSTRAINT fk_ Animal FOREIGN KEY (AdopterID)
REFERENCES Foster_Home(AdopterID),

CONSTRAINT fk_Animal FOREIGN KEY (Foster_Home_ID)
REFERENCES Foster_Home (Foster_Home_ID) );
```

Inserting Animal

```
 INSERT INTO Animal (Animal_ID, Animal_Age, Medical_ID, Adoption_Date, Adopter_ID,
Foster_Home_ID) VALUES (1, 2, 'M001', #2023-01-15#, 101, 201);

INSERT INTO Animal (Animal_ID, Animal_Age, Medical_ID, Adoption_Date, Adopter_ID,
Foster_Home_ID) VALUES (2, 4, 'M002', #2023-02-20#, 102, 202);
```



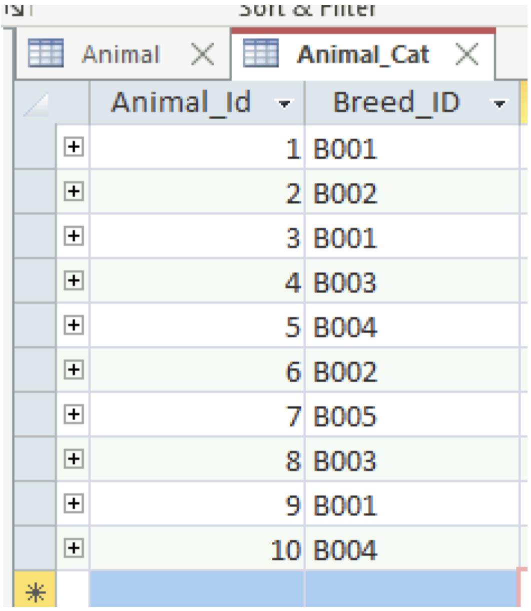
Animal_ID	Animal_Age	Medical_ID	Adoption_Date	Adopter_ID	Foster_Home_ID
1	2	M001	1/15/2023	101	201
2	4	M002	2/20/2023	102	202
3	1	M003	3/5/2023	103	203
4	3	M004	4/10/2023	104	204
5	5	M005	5/25/2023	105	205
6	2	M006	6/30/2023	106	206
7	3	M007	7/15/2023	107	207
8	4	M008	8/20/2023	108	208
9	6	M009	9/10/2023	109	209
10	1	M010	10/5/2023	110	210

Animal_Cat

```
CREATE TABLE Animal_Cat (
Animal_ID NUMBER NOT NULL,
  Breed_ID VARCHAR(50),
CONSTRAINT pk_Animal_Cat PRIMARY KEY (Animal_ID)
);
```

Inserting Animal_Cat

```
INSERT INTO Animal_Cat (Animal_ID, Breed_ID) VALUES (1, 'B001');
INSERT INTO Animal_Cat (Animal_ID, Breed_ID) VALUES (2, 'B002');
```



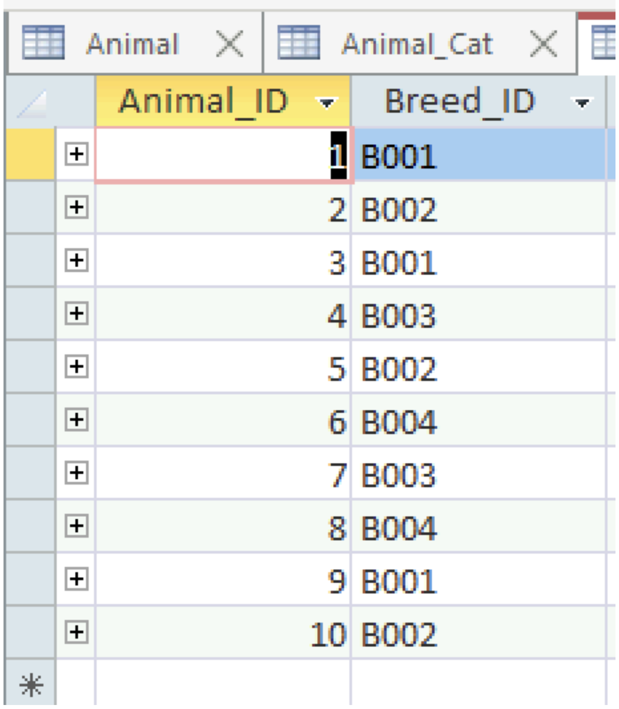
Animal_Id	Breed_ID
1	B001
2	B002
3	B001
4	B003
5	B004
6	B002
7	B005
8	B003
9	B001
10	B004

Animal_Dog

```
CREATE TABLE Animal_Dog (
Animal_ID NUMBER NOT NULL,
  Breed_ID VARCHAR(50),
CONSTRAINT pk_Animal_Dog PRIMARY KEY (Animal_ID) );
```

Inserting Animal_Dog

```
INSERT INTO Animal_Dog (Animal_ID, Breed_ID) VALUES (1, 'B001');
INSERT INTO Animal_Dog (Animal_ID, Breed_ID) VALUES (2, 'B002');
```



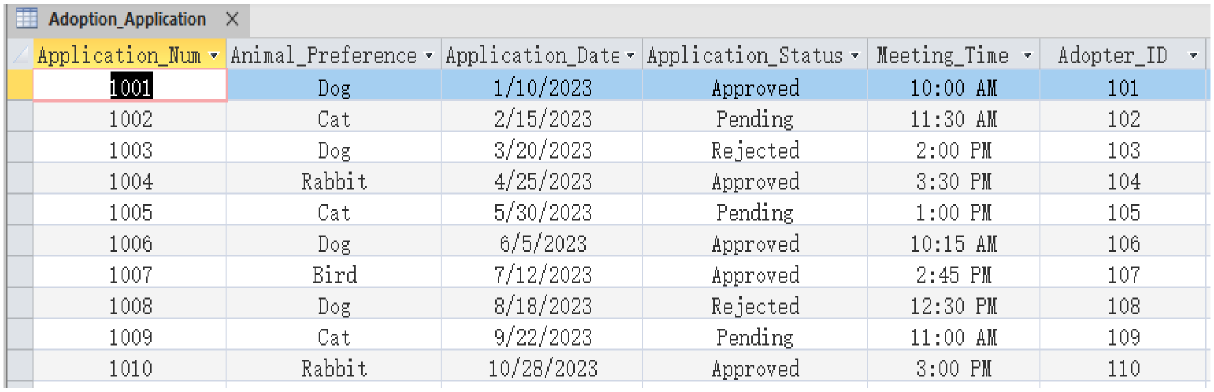
Animal_ID	Breed_ID
1	B001
2	B002
3	B001
4	B003
5	B002
6	B004
7	B003
8	B004
9	B001
10	B002

Adoption_Application

```
CREATE TABLE Adoption_Application (

Application_Num NUMBER NOT NULL,

    Animal_Preference VARCHAR (50),

    Application_Date DATE,

    Application_Status VARCHAR (30),

    Meeting_Time TIME,

    Adopter_ID NUMBER NOT NULL,
CONSTRAINT pk_Adoption_Application PRIMARY KEY (Application_Num),

CONSTRAINT fk_Adoption_Application FOREIGN KEY (Adopter_ID) REFERENCES Adopter(Adopter_ID));
```

Insert Data

```
INSERT INTO Adoption_Application  VALUES (1001, ‘Dog’, ‘10-JAN-23’, ‘Approved’, 10:00, 101)

INSERT INTO Adoption_Application  VALUES (1002, ‘Cat’, ‘12-FEB-23’, ‘Pending’, 11:30, 102)
```



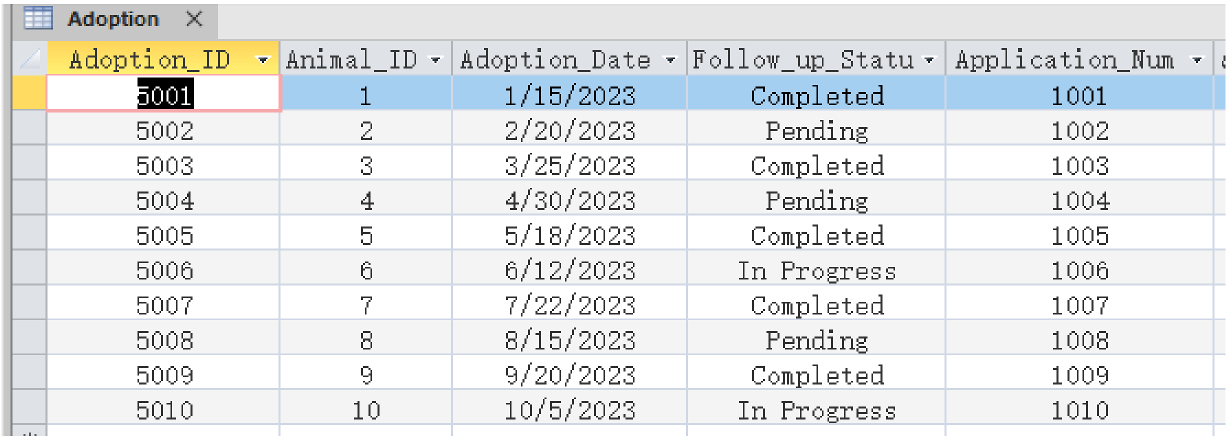
Application_Num	Animal_Preference	Application_Date	Application_Status	Meeting_Time	Adopter_ID
1001	Dog	1/10/2023	Approved	10:00 AM	101
1002	Cat	2/15/2023	Pending	11:30 AM	102
1003	Dog	3/20/2023	Rejected	2:00 PM	103
1004	Rabbit	4/25/2023	Approved	3:30 PM	104
1005	Cat	5/30/2023	Pending	1:00 PM	105
1006	Dog	6/5/2023	Approved	10:15 AM	106
1007	Bird	7/12/2023	Approved	2:45 PM	107
1008	Dog	8/18/2023	Rejected	12:30 PM	108
1009	Cat	9/22/2023	Pending	11:00 AM	109
1010	Rabbit	10/28/2023	Approved	3:00 PM	110

Adoption

```
CREATE TABLE Adoption (
Adoption_ID NUMBER NOT NULL,
    Animal_ID NUMBER,
    Adoption_Date DATE,
    Follow_up_Status VARCHAR (50),
    Application_Num NUMBER NOT NULL,
CONSTRAINT pk_Adoption PRIMARY KEY (Adoption_ID),
CONSTRAINT fk_Adoption FOREIGN KEY (Application_Num) REFERENCES
Adoption_Application(pplication_Num));
```

Insert Data

```
INSERT INTO Adoption  VALUES (5001, 1, ‘15-JAN-23’, ‘Completed’,  1001)

INSERT INTO Adoption  VALUES (5002, 2, ‘20-FEB-23’, ‘Pending’, 1002)
```



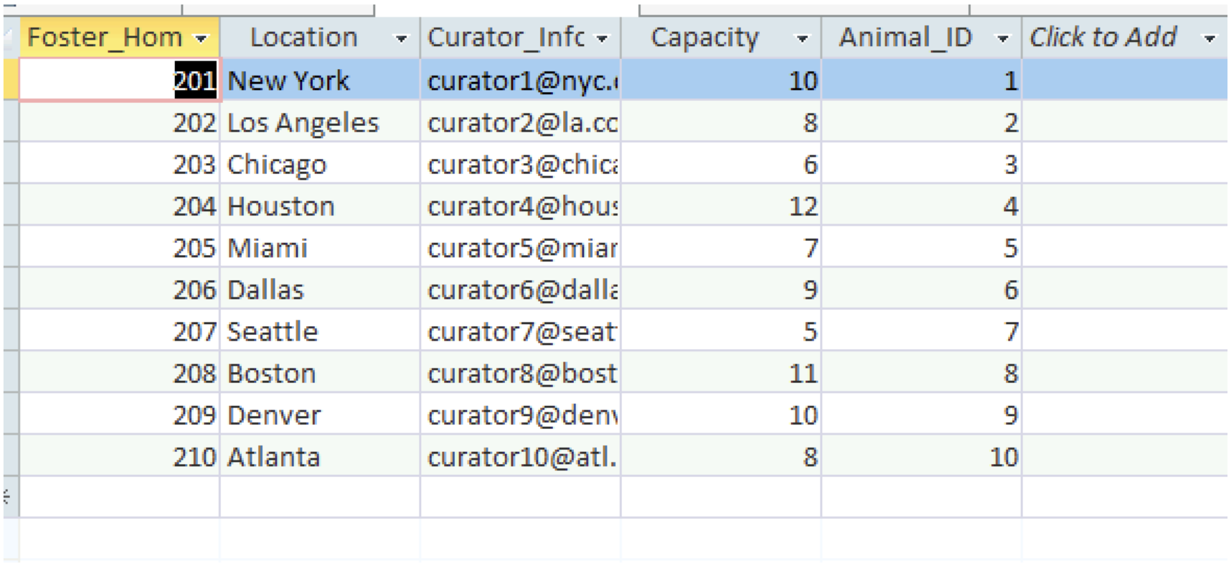
Adoption_ID	Animal_ID	Adoption_Date	Follow_up_Statu	Application_Num
5001	1	1/15/2023	Completed	1001
5002	2	2/20/2023	Pending	1002
5003	3	3/25/2023	Completed	1003
5004	4	4/30/2023	Pending	1004
5005	5	5/18/2023	Completed	1005
5006	6	6/12/2023	In Progress	1006
5007	7	7/22/2023	Completed	1007
5008	8	8/15/2023	Pending	1008
5009	9	9/20/2023	Completed	1009
5010	10	10/5/2023	In Progress	1010

Foster Home

```
CREATE TABLE Foster_Home (
   Foster_Home_ID NUMBER NOT NULL,
   Location VARCHAR(50) NOT NULL,
   Curator_Info VARCHAR(50),
   Capacity NUMBER,
   Animal_ID NUMBER,
   CONSTRAINT pk_Foster_Home PRIMARY KEY (Foster_Home_ID),
   CONSTRAINT fk_Animal FOREIGN KEY (Animal_ID) REFERENCES Animal (Animal_ID)
);
```

INSERT INTO FOSTER HOME

```
INSERT INTO Foster_Home (Foster_Home_ID, Location, Curator_Info, Capacity, Animal_ID) VALUES
(201, 'New York', 'curator1@nyc.com', 10, 1);

INSERT INTO Foster_Home (Foster_Home_ID, Location, Curator_Info, Capacity, Animal_ID) VALUES
(202, 'Los Angeles', 'curator2@la.com', 8, 2);
```



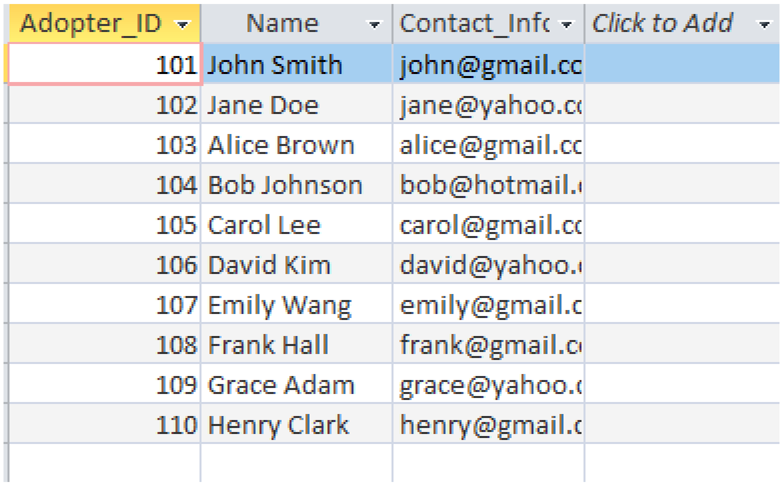
Foster_Hom	Location	Curator_Infc	Capacity	Animal_ID	*Click to Add*
201	New York	curator1@nyc.	10	1	
202	Los Angeles	curator2@la.cc	8	2	
203	Chicago	curator3@chica	6	3	
204	Houston	curator4@hous	12	4	
205	Miami	curator5@miar	7	5	
206	Dallas	curator6@dalla	9	6	
207	Seattle	curator7@seat	5	7	
208	Boston	curator8@bost	11	8	
209	Denver	curator9@denv	10	9	
210	Atlanta	curator10@atl.	8	10	

Adopter

```
CREATE TABLE Adopter (
    Adopter_ID NUMBER NOT NULL,
    Name VARCHAR(50) NOT NULL,
    Contact_Info VARCHAR(50) NOT NULL,
    CONSTRAINT pk_Adopter PRIMARY KEY (Adopter_ID)
);
```

INSERT INTO ADOPTER

```
INSERT INTO Adopter (Adopter_ID, Name, Contact_Info) VALUES (101, 'John Smith', 'john@gmail.com');
INSERT INTO Adopter (Adopter_ID, Name, Contact_Info) VALUES (102, 'Jane Doe', 'jane@yahoo.com');
```



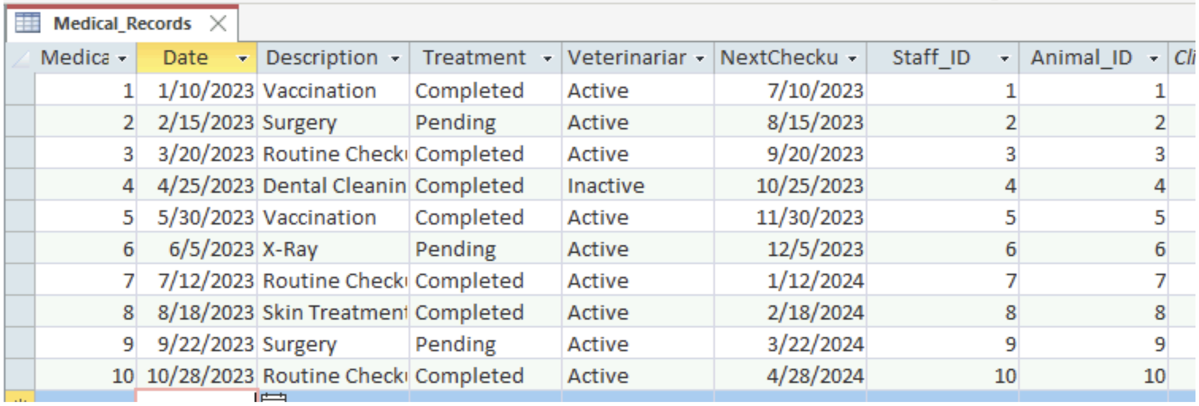
Adapter_ID	Name	Contact_Infc	Click to Add
101	John Smith	john@gmail.cc	
102	Jane Doe	jane@yahoo.c	
103	Alice Brown	alice@gmail.cc	
104	Bob Johnson	bob@hotmail.	
105	Carol Lee	carol@gmail.cc	
106	David Kim	david@yahoo.	
107	Emily Wang	emily@gmail.c	
108	Frank Hall	frank@gmail.c	
109	Grace Adam	grace@yahoo.	
110	Henry Clark	henry@gmail.c	

Medical Records

```
CREATE TABLE Medical_Records (
      MedicalRecordID NUMBER NOT NULL,
      Date DATE,
      Description VARCHAR(50),
      Treatment VARCHAR(50),
      Veterinarian_Status VARCHAR(50),
      NextCheckupDate DATE,
      Staff_ID NUMBER NOT NULL,
      Animal_ID NUMBER NOT NULL,

CONSTRAINT pk_Medical_Records PRIMARY KEY (MedicalRecordID),

CONSTRAINT fk_Medical_Records FOREIGN KEY (STAFF_ID) REFERENCES Medical_Treatment(Staff_ID))

CONSTRAINT fk_Medical_Records FOREIGN KEY (Animal_ID) REFERENCES Animal(Animal_ID))
```

Insert into Medical Records

```
INSERT INTO Medical_Records (MedicalRecordID, Date, Description, Treatment,
Veterinarian_Status,NextCheckupDate, Staff_ID, Animal_ID)
VALUES (1, #1/10/2023#, 'Vaccination', 'Completed', 'Active', #7/10/2023#, 1, 1);

INSERT INTO Medical_Records (MedicalRecordID, Date, Description, Treatment,
Veterinarian_Status,NextCheckupDate, Staff_ID, Animal_ID)
VALUES (2, #2/15/2023#, 'Surgery', 'Pending', 'Active', #8/15/2023#, 2, 2);
```



Medical_Records

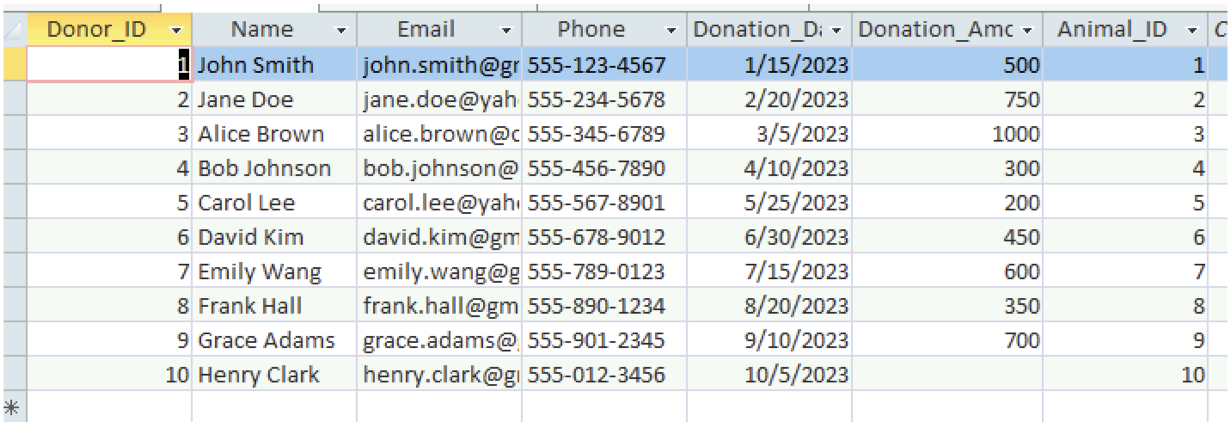
Medica	Date	Description	Treatment	Veterinariar	NextChecku	Staff_ID	Animal_ID	Cli
1	1/10/2023	Vaccination	Completed	Active	7/10/2023	1	1	
2	2/15/2023	Surgery	Pending	Active	8/15/2023	2	2	
3	3/20/2023	Routine Checki	Completed	Active	9/20/2023	3	3	
4	4/25/2023	Dental Cleanin	Completed	Inactive	10/25/2023	4	4	
5	5/30/2023	Vaccination	Completed	Active	11/30/2023	5	5	
6	6/5/2023	X-Ray	Pending	Active	12/5/2023	6	6	
7	7/12/2023	Routine Checki	Completed	Active	1/12/2024	7	7	
8	8/18/2023	Skin Treatment	Completed	Active	2/18/2024	8	8	
9	9/22/2023	Surgery	Pending	Active	3/22/2024	9	9	
10	10/28/2023	Routine Checki	Completed	Active	4/28/2024	10	10	

Donor

```
CREATE TABLE Staff (
    Staff_ID NUMBER NOT NULL,
    Name VARCHAR(50),
    DOB DATE,
    Title VARCHAR(50),
    Start_Date DATE,
    Termination VARCHAR(50),
    Animal_ID NUMBER,
    CONSTRAINT pk_Staff PRIMARY KEY (Staff_ID),
    CONSTRAINT fk_Animal FOREIGN KEY (Animal_ID) REFERENCES Animal(Animal_ID)
);
```

Insert into Donor

```
INSERT INTO Donor (Donor_ID, Name, Email, Phone, Donation_Date, Donation_Amount, Animal_ID)
VALUES (1, 'John Smith', 'john.smith@gr', '555-123-4567', #2023-01-15#, 500, 1);

INSERT INTO Donor (Donor_ID, Name, Email, Phone, Donation_Date, Donation_Amount, Animal_ID)
VALUES (2, 'Jane Doe', 'jane.doe@yah', '555-234-5678', #2023-02-20#, 750, 2);
```



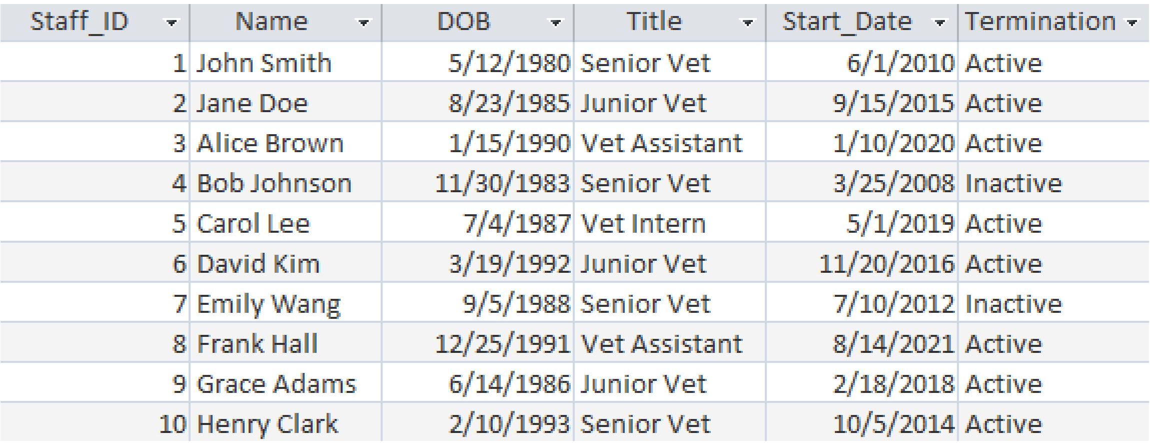
Donor_ID	Name	Email	Phone	Donation_D	Donation_Amc	Animal_ID	C
1	John Smith	john.smith@gr	555-123-4567	1/15/2023	500	1	
2	Jane Doe	jane.doe@yah	555-234-5678	2/20/2023	750	2	
3	Alice Brown	alice.brown@c	555-345-6789	3/5/2023	1000	3	
4	Bob Johnson	bob.johnson@	555-456-7890	4/10/2023	300	4	
5	Carol Lee	carol.lee@yah	555-567-8901	5/25/2023	200	5	
6	David Kim	david.kim@gm	555-678-9012	6/30/2023	450	6	
7	Emily Wang	emily.wang@g	555-789-0123	7/15/2023	600	7	
8	Frank Hall	frank.hall@gm	555-890-1234	8/20/2023	350	8	
9	Grace Adams	grace.adams@	555-901-2345	9/10/2023	700	9	
10	Henry Clark	henry.clark@gi	555-012-3456	10/5/2023		10	
*							

Staff

```
CREATE TABLE Staff (
Staff_ID NUMBER NOT NULL,
Name VARCHAR(50),
DOB DATE,
Title VARCHAR(50),
Start_Date DATE,
Termination VARCHAR(50),
CONSTRAINT pk_Staff PRIMARY KEY (Staff_ID)
);
```

Insert into Staff

```
INSERT INTO Staff (Staff_ID, Name, DOB, Title, Start_Date, Termination)
VALUES (1, 'John Smith', #5/12/1980#, 'Senior Vet', #6/1/2010#, "Active")

INSERT INTO Staff (Staff_ID, Name, DOB, Title, Start_Date, Termination)
VALUES (2, 'Jane Doe', #8/23/1985#, 'Junior Vet', #9/15/2015#, 'Active')

INSERT INTO Staff (Staff_ID, Name, DOB, Title, Start_Date, Termination)
VALUES (3, 'Alice Brown', #1/15/1990#, 'Vet Assistant', #1/10/2020#, 'Active')
```



Staff_ID	Name	DOB	Title	Start_Date	Termination
1	John Smith	5/12/1980	Senior Vet	6/1/2010	Active
2	Jane Doe	8/23/1985	Junior Vet	9/15/2015	Active
3	Alice Brown	1/15/1990	Vet Assistant	1/10/2020	Active
4	Bob Johnson	11/30/1983	Senior Vet	3/25/2008	Inactive
5	Carol Lee	7/4/1987	Vet Intern	5/1/2019	Active
6	David Kim	3/19/1992	Junior Vet	11/20/2016	Active
7	Emily Wang	9/5/1988	Senior Vet	7/10/2012	Inactive
8	Frank Hall	12/25/1991	Vet Assistant	8/14/2021	Active
9	Grace Adams	6/14/1986	Junior Vet	2/18/2018	Active
10	Henry Clark	2/10/1993	Senior Vet	10/5/2014	Active

Supervisor

CREATE TABLE Supervisor (

Sup_ID NUMBER ,

Staff_ID NUMBER ,

CONSTRAINT pk_Supervisor PRIMARY KEY (Sup_ID),

CONSTRAINT fk_Staff FOREIGN KEY (Staff_ID) REFERENCES Staff (Staff_ID));

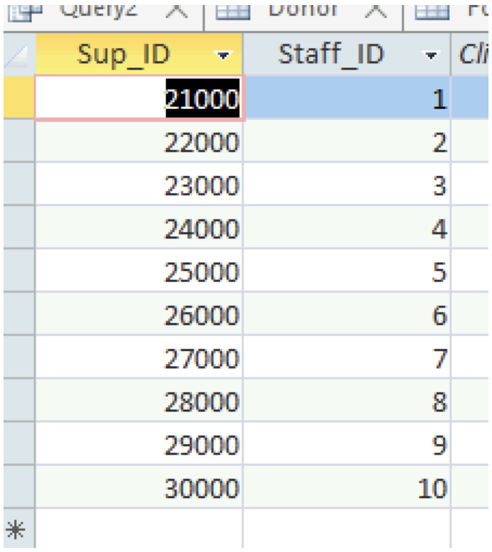
INSERT Supervisor

INSERT INTO Supervisor (Sup_ID, Staff_ID)

VALUES (21000, 1);

INSERT INTO Supervisor (Sup_ID, Staff_ID)

VALUES (22000, 2);



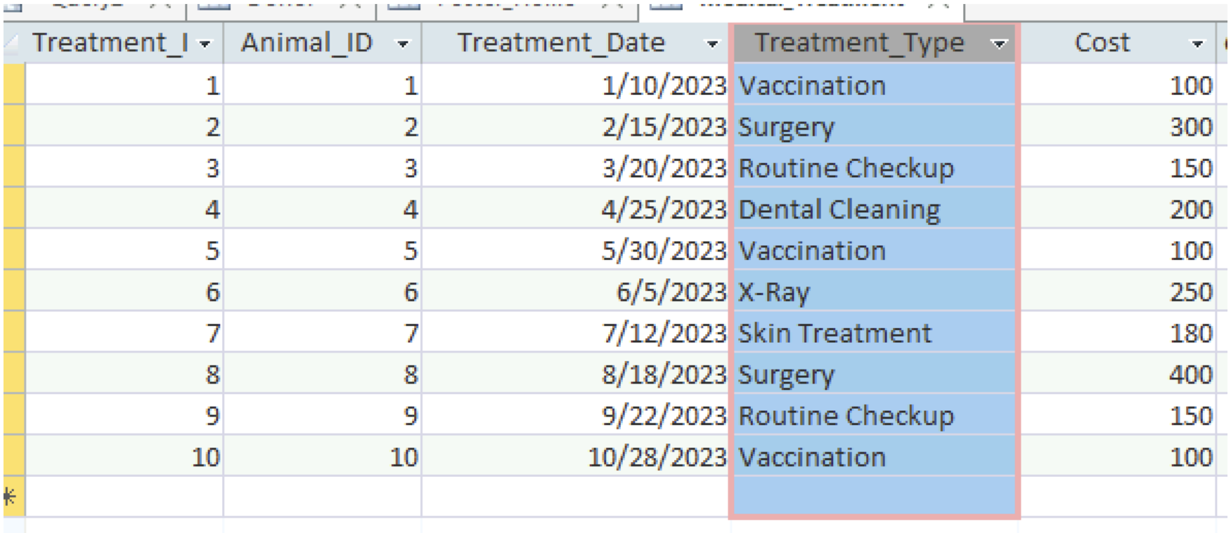
Sup_ID	Staff_ID	Cli
21000	1	
22000	2	
23000	3	
24000	4	
25000	5	
26000	6	
27000	7	
28000	8	
29000	9	
30000	10	

Medical Treatment

```
CREATE TABLE Medical_Treatment (
   Treatment_ID NUMBER NOT NULL,
   Animal_ID NUMBER NOT NULL,
   Treatment_Date DATE,
   Treatment_Type VARCHAR(50),
   Cost NUMBER,
   CONSTRAINT pk_Medical_Treatment PRIMARY KEY (Treatment_ID)
);
```

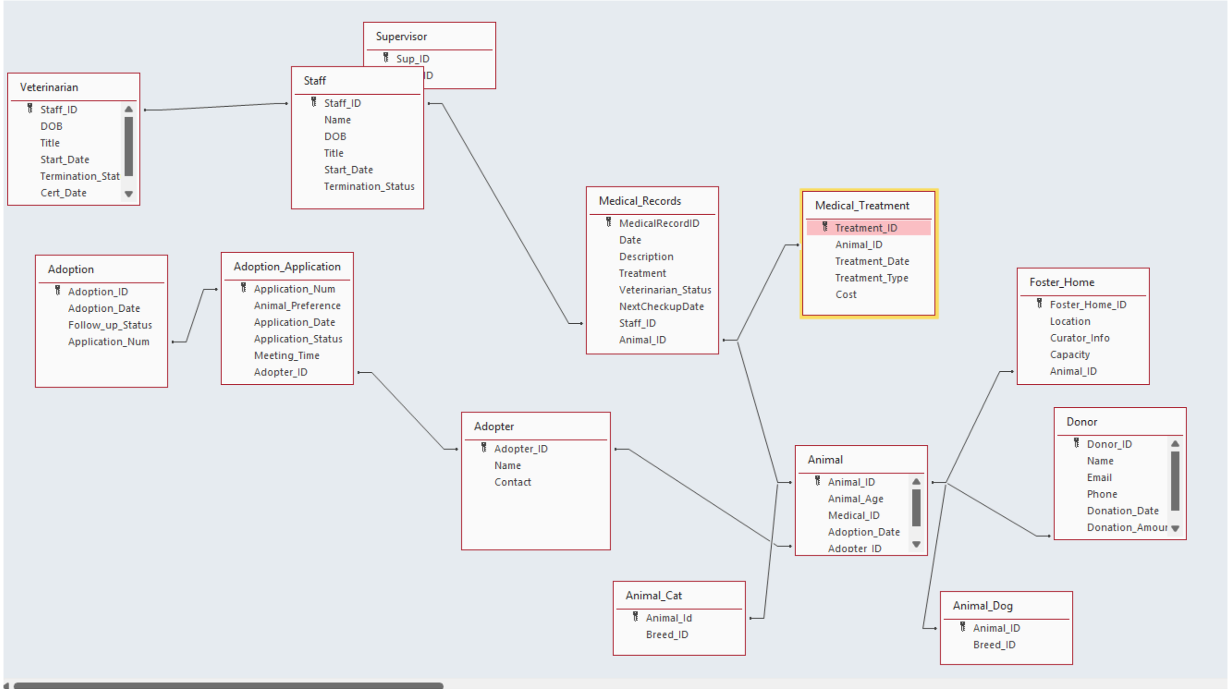
Insert into Medical Treatment

```
INSERT INTO Medical_Treatment (Treatment_ID, Animal_ID, Treatment_Date, Treatment_Type, Cost)
VALUES (1, 1, #1/10/2023#, 'Vaccination', 100);

INSERT INTO Medical_Treatment (Treatment_ID, Animal_ID, Treatment_Date, Treatment_Type, Cost)
VALUES (2, 1, #2/15/2023#, 'Surgery', 300);
```



Treatment_I	Animal_ID	Treatment_Date	Treatment_Type	Cost
1	1	1/10/2023	Vaccination	100
2	2	2/15/2023	Surgery	300
3	3	3/20/2023	Routine Checkup	150
4	4	4/25/2023	Dental Cleaning	200
5	5	5/30/2023	Vaccination	100
6	6	6/5/2023	X-Ray	250
7	7	7/12/2023	Skin Treatment	180
8	8	8/18/2023	Surgery	400
9	9	9/22/2023	Routine Checkup	150
10	10	10/28/2023	Vaccination	100

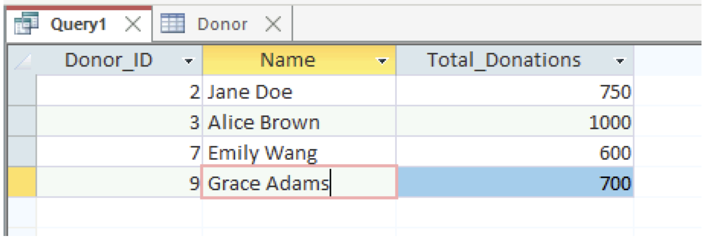
```
ALTER Medical_Treatment

ADD CONSTRAINT fk_Medical_Treatment_Animal FOREIGN KEY (Animal_ID) REFERENCES Animal(Animal_ID)
```


Supervisor
Sup_ID
D
Veterinarian
Staff_ID
DOB
Title
Start_Date
Termination_Stat
Cert_Date
Staff
Staff_ID
Name
DOB
Title
Start_Date
Termination_Status
Medical_Records
MedicalRecordID
Date
Description
Treatment
Veterinarian_Status
NextCheckupDate
Staff_ID
Animal_ID
Medical_Treatment
Treatment_ID
Animal_ID
Treatment_Date
Treatment_Type
Cost
Adoption
Adoption_ID
Adoption_Date
Follow_up_Status
Application_Num
Adoption_Application
Application_Num
Animal_Preference
Application_Date
Application_Status
Meeting_Time
Adopter_ID
Foster_Home
Foster_Home_ID
Location
Curator_Info
Capacity
Animal_ID
Adopter
Adopter_ID
Name
Contact
Animal
Animal_ID
Animal_Age
Medical_ID
Adoption_Date
Adopter_ID
Donor
Donor_ID
Name
Email
Phone
Donation_Date
Donation_Amour
Animal_Cat
Animal_Id
Breed_ID
Animal_Dog
Animal_ID
Breed_ID

Scenario Queries

Scenario 1: Who has donated more than $500 in total, and how much did they donate?

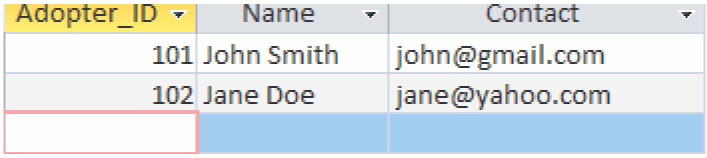
```
SELECT Donor_ID, Name, ROUND(SUM(Donation_Amount), 2) AS Total_Donations
FROM Donor
GROUP BY Donor_ID, Name
HAVING SUM(Donation_Amount) > 500;
```



Donor_ID	Name	Total_Donations
2	Jane Doe	750
3	Alice Brown	1000
7	Emily Wang	600
9	Grace Adams	700

Scenario 2 : Write an SQL query using a subquery to find the details (Adopter_ID, Name, and Contact_Info) of adopters whose Adopter_ID is less than 105 and whose contact info starts with the letter 'j'.

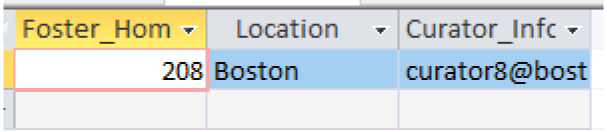
```
SELECT Adopter_ID, Name, Contact
FROM Adopter
WHERE Adopter_ID IN (
        SELECT Adopter_ID
        FROM Adopter
        WHERE Adopter_ID <105 AND Contact LIKE ‘j*’
)
```



Adopter_ID	Name	Contact
101	John Smith	john@gmail.com
102	Jane Doe	jane@yahoo.com

Scenario 3: A non-profit organization wants to identify foster homes that can accommodate a larger number of animals and are located in cities starting with the letter "B." Specifically, they need the Foster_Home_ID, Location, and Curator_Info for foster homes with a Capacity greater than 8 in such cities. Write a query to retrieve this information'.

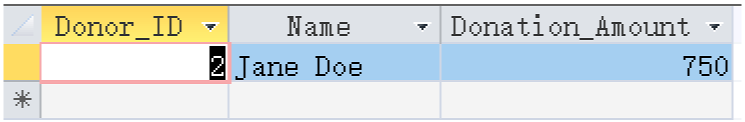
```
SELECT Foster_Home_ID, Location, Curator_Info
FROM Foster_Home
WHERE Foster_Home_ID IN (
        SELECT Foster_Home_ID
        FROM Foster_Home
        WHERE Capacity > 8 AND Location LIKE 'B*'
)
```



Foster_Hom	Location	Curator_Infc
208	Boston	curator8@bost

Scenario 4: Write a query that displays the name of the donor with the largest amount of money donated and contains the letter D.

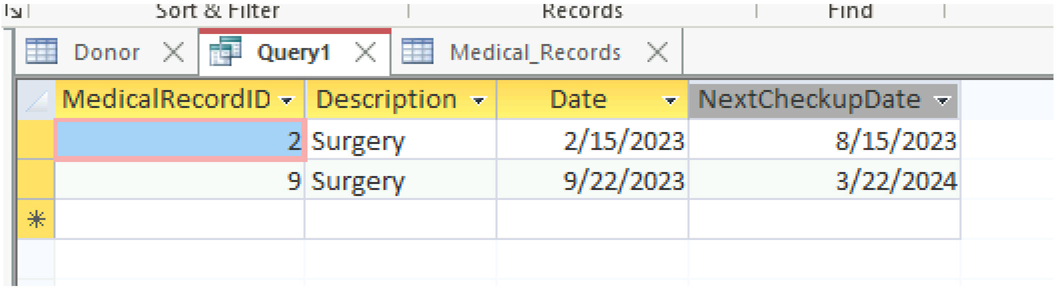
```
SELECT TOP 1 Donor_ID, Name, Donation_Amount

FROM Donor

WHERE Name like '*D*'

ORDER BY Donation_Amount DESC
```



Donor_ID	Name	Donation_Amount
2	Jane Doe	750

Scenario 5: The veterinary clinic needs to identify all animals who had surgery-related procedures that are currently in 'Pending' status to make sure follow ups are scheduled before the end of the year.

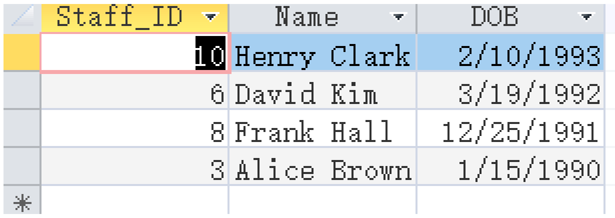
```
SELECT MedicalRecordID, Description, Date, NextCheckupDate

FROM Medical_Records

WHERE Description LIKE'*Surg*' AND Treatment LIKE 'Pending';
```



MedicalRecordID	Description	Date	NextCheckupDate
2	Surgery	2/15/2023	8/15/2023
9	Surgery	9/22/2023	3/22/2024

Scenario 6: Write a query to display all the staff who were born after the 1990s.

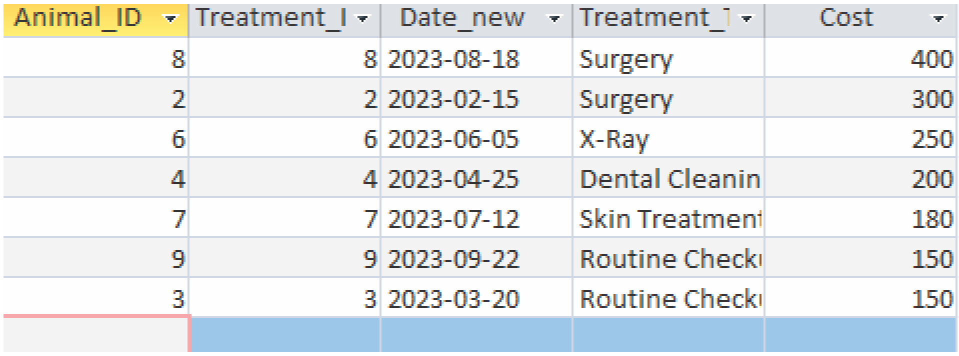
```
SELECT Staff_ID, Name, DOB
FROM Staff
WHERE DOB > #01/01/1990#
ORDER BY DOB DESC
```



Staff_ID	Name	DOB
10	Henry Clark	2/10/1993
6	David Kim	3/19/1992
8	Frank Hall	12/25/1991
3	Alice Brown	1/15/1990

Scenario 7: The management wants to generate a report on medical treatments where the cost exceeds a certain threshold (e.g., $100) for animals that underwent treatments. This report will include details of the treatment and the corresponding animal's ID. Also rewrite the dates in a different format

```
SELECT Animal_ID, Treatment_ID, FORMAT (Treatment_Date, 'YYYY-MM-DD') AS Date_new,
Treatment_Type, Cost
FROM Medical_Treatment
WHERE Cost > 100
ORDER BY Cost DESC;
```



Animal_ID	Treatment_I	Date_new	Treatment_T	Cost
8	8	2023-08-18	Surgery	400
2	2	2023-02-15	Surgery	300
6	6	2023-06-05	X-Ray	250
4	4	2023-04-25	Dental Cleanin	200
7	7	2023-07-12	Skin Treatment	180
9	9	2023-09-22	Routine Check	150
3	3	2023-03-20	Routine Check	150

Conclusion

The Pet Adoption Management System project was an invaluable learning experience that provided us with a deep understanding of how to design and implement a data management system. Throughout the process, we gained hands-on knowledge of key database concepts, including drawing Entity-Relationship Diagrams (**ERDs**), converting them into Relational Data Models (**RDMs)**, **normalizing** the data to ensure consistency, and finally creating and populating tables using **SQL**.

At the start of the project, we created challenges in creating the ERD. The iterative process of refining the ERD to accurately represent the business scenario was one of the most difficult tasks. Initially, we overcomplicated the design by trying to include unnecessary complexity, which made the initial ERD unclear.

However, RDM conversion and normalization turned out to be more straightforward. These steps allowed us to break the data down into logical tables, reducing redundancy and improving efficiency. Understanding normalization from 1NF to BCNF was essential in ensuring that our database structure was optimized for performance.

The system effectively supports the functionality of a pet adoption center by providing a streamlined way to manage and retrieve data in a timely and organized manner. In the future, to further enhance the project, we could consider adding a few more entities, such as "Donor Information" or "Medical History Logs," to make the system more robust and applicable to real-world scenarios.
Overall, this project was an excellent opportunity to gain practical experience in database design and management. It allowed us to apply theoretical concepts to a real-world problem, deepening our understanding of how IT systems are used to support organizational needs.