Animal Shelter Database

Mini Project Report - Database Lab (DSE 2260)

Department of Data Science & Computer Applications

B. Tech Data Science

4th Semester – Batch: B3

Submitted By

Gowri Dinesh Nair	200968001
Shreya Nayak A.	200968004
Mohammed Hassan Raza	200968005
Sai Anish Malla	200968009

Mentored By

Vinayak M	Archana H/Shameem
Assistant Professor-Senior	Assistant Professor-Senior
DSCA, MIT	DSCA, MIT

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

Date: 10/05/2022

CERTIFICATE

This is to certify that Gowri Dinesh Nair (200968001), Shreya Nayak A. (200968004), Mohammed Hassan Raza (200968005), Sai Anish Malla (200968009), have successfully executed a mini project titled “Animal Shelter Database.” rightly bringing fore the competencies and skillsets they have gained during the course- Database Lab (DSE 2262 & DSE), thereby resulting in the culmination of this project.

Vinayak M	**Archana H / Shameem**
Assistant Professor-Senior	**Assistant Professor-Senior**
DSCA, MIT	**DSCA, MIT**

ABSTRACT

Around 5,000 independently run animal shelters are estimated to exist around the globe today. The most crucial aspect of maintaining an animal shelter is data. The animal sheltering industry lacks standardized methods of data collection and analysis. The resulting lack of available data limits our understanding of the homeless animal population There is information to be collected regarding the animals, the operation costs, the employees, visitors, adopters, and so on. Having this information on file will make communication quicker and easier in finding a foster for an animal in need or asking people to attend a fundraiser to save more lives, reduce operating expenses, and link animals with potential donors at a shelter.

Contents

6. Result

7. Conclusion and Future Work

Chapter 1

Introduction

There are an estimated 200 million stray dogs worldwide, nearly 20% of which live on India's streets. Of these, only a few are fortunate to make it into well-established animal shelters. Shelters protect stray, lost, abandoned, or surrendered pets. Each pet demands unique care and attention, which the shelter provides. While providing rescued dogs with adequate care has been an issue for decades, animal shelters today are far better developed than their predecessors. When the first animal shelters were established in the early 19th century, a significant number of the animals taken in were deemed "unwanted" and routinely killed or euthanized in the interest of dog population management.

However, the 20th century saw a more humane shift in mindset regarding animal welfare ignited by the general public's mission to limit the spread of zoonotic diseases. Thus, animal shelters witnessed a large drop in euthanasia and began offering strays better accommodation and protection. But the resulting increase in stray dog populations has introduced new spacing and logistical issues in animal shelters that still exist today.

The quality of these shelters is highly dependent on their ability to maintain and analyze data to provide refuge to the sheltered dogs. Considering that the majority of animal shelters are not-for-profit highly dependent on public donations, it is essential that all data regarding rescues and logistical issues are well defined and accurate for transparency with donors and adopters. As stray populations increase in India, shelters must be able to quantify their successes and needs to support the growth of their activities. Due to a lack of resources, many shelters resort to the time-consuming manual entry of data into spreadsheets. Thus there must exist a reliable means of accessing and retrieving data efficiently and frequently.

Typically animal shelters collect, maintain, and analyze data on the number of animals rescued, adopted, and returned. Metrics such as the duration of the animal's stay at the shelter and the animal's healthcare are also involved. In this project, we create a database that includes details of the above in addition to staff, shelter, and cage information (cages available, cages occupied, etc.). This allows the shelter to keep track of all logistics required to ensure that as many needs are met.

Chapter 2

Synopsis

2.1 Proposed System

Problem statement: To construct a dog shelter database that can be searched using a range of characteristics to analyze and collect data about the dogs, shelter, and visitors. The solution we suggest is **a database application that gathers and stores information about the shelter, available pets and their attributes, cage numbers, shelter employee, and visitor information.**

It is high time that all animal shelters adopt data-driven practices if they are to aid more animals. Having clear data-driven practices can also help the shelter run more smoothly ,perform better and gauge their practices

The shelter database tracks:

Animal data including breed, color, weather spayed or neutered, and descriptive data such as approximate age, name, and dog id. Data related to prior history: date of arrival ,date of adoption

- Cage occupancy and number of cages per shelter need to be tracked.
- Employee information that would aid the shelter in forecasting staffing requirements.
- Visitor personal details, date of visit, dog visited, etc.
- Event details which are further linked to shelter, dog, and visitor relations.

.

2.2 Objectives

Having a database creates quantifiable views of the shelter's activities and hence acts like a common point for each of the concerned parties to discuss.

The Main Objective of the work are

1. Obtain information related to the dog
2. To maintain a record of visitors and their activities such as adoption of a dog
3.In case, we need to find the dogs condition before adoption we can contact visitors,this will be easier when we have stored in an organized manner by creating respective tables
4. To find out the number of adoptions during a certain time period.

Chapter 3

Functional Requirements

This project will help in storing all the information of a dog shelter very efficiently.

It allows user to insert and update all the information related to a dog. For a user to be able to make changes to the database he/she will have to enter the correct password.

3.1 User Registering/Login module

Two lines about module briefly and it supports functionalities- New user registration, Login, Forgot password

3.1.1 New User Registration

The user must be able to create user id and password by supplying appropriate details.

INPUT	New username, Password, phone number, permissions
Processing	Username must be valid (no spaces and no special characters) and unique (user as Primary key) Passwords must follow criteria: minimum 9 characters, at least one capital, one number and one special character Ensure the phone number is Indian (starts with +91) and is 10 numbers in length By default all users are set as “users” special permissions can be given to admins, staff and more.
OUTPUT	User created successfully message and stored in the database

3.1.2 Login

The existing user must be able to login upon entering proper username and password.

INPUT	username, Password
Processing	Ensure the username exists in the database and compare the passwords.
OUTPUT	If user entered correct user name & Password Login successful and open main application menu Else Display Login not successful, retry logging in

3.1.3 Forgot password

If an existing username is not able to login, “forgot password” can be used to reset the password.

INPUT	Prompt the user to enter username, Phone

Processing	
	```
	If username and corresponding phone exist in the data storage
	    Send OTP to Phone.
	    Prompt the user to enter OTP
	    If OTP matching
	        Prompt user to change password.
	    Else
	        OTP not matching, give option to resend OTP.
	Else
	    User name and corresponding Phone not existing in the storage, ask again or create a new login.
	```

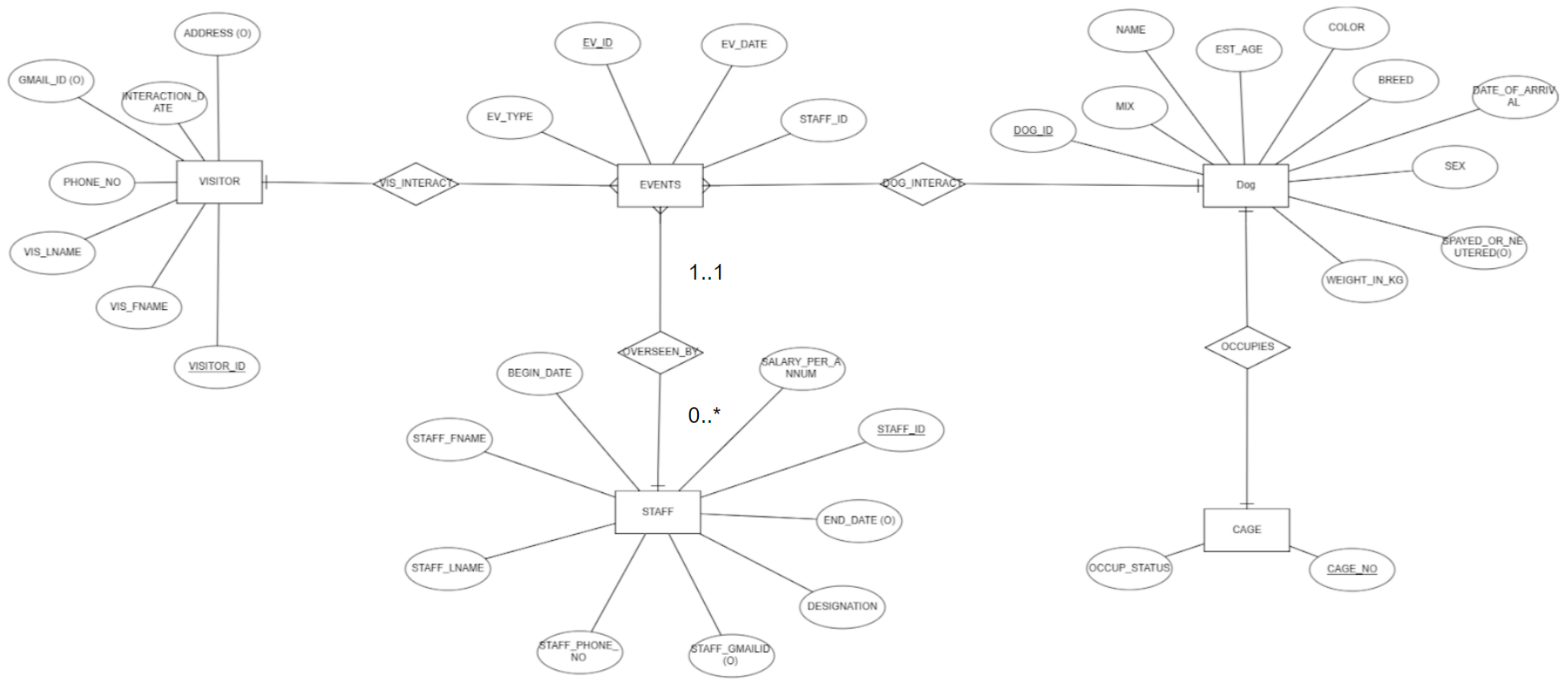
OUTPUT	Password successfully changed / User name, phone not matching

Chapter 4

Detailed Design

4.1

ER Diagram



4.2 Schema Diagram

Visitor (VISITOR_ID, VIS_FNAME, VIS_LNAME, PHONE_NO, GMAIL_ID, ADDRESS, INTERACTION_DATE, EV_ID)

EV_ID References Events

Events (EV_ID, EV_TYPE, DOG_ID, EV_DATE, VISITOR_ID , STAFF_ID)

DOG_ID references Dog, VISITOR_ID References Visitor

Staff(STAFF_ID, STAFF_FNAME, STAFF_LNAME, STAFF_PHONE_NO, STAFF_GMAILID, DESIGNATION, SUPERV_ID, SALARY_PER_ANNUM, BEGIN_DATE, END_DATE)

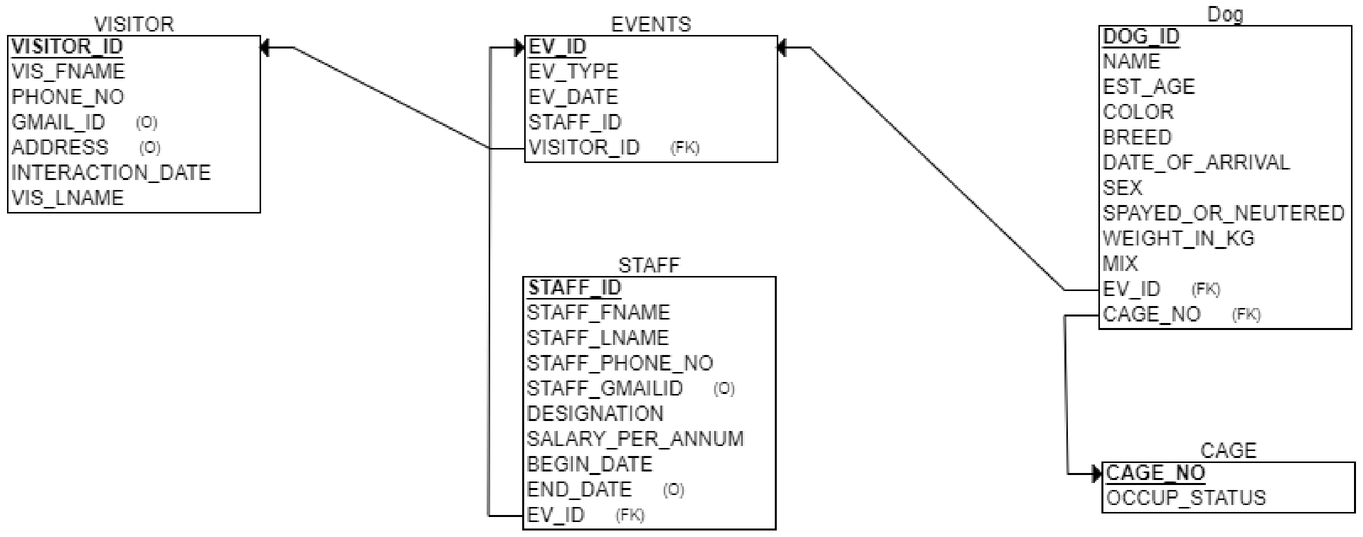
SUPERV_ID references Staff(Staff_ID),

Cage (CAGE_NO, DOG_ID, OCCUP_STATUS, DOG_ID)

Dog_ID references Dog

Dog(DOG_ID, NAME, EST_AGE, COLOR, BREED, DATE_OF_ARRIVAL, SEX, SPAYED_OR_NEUTERED, WEIGHT_IN_KG, MIX, EV_ID)

EV_ID references Events



4.3 Data Dictionary

CAGE

Column	Data type (size)	Constraint	Constraint Name
CAGE_NO	number(5)		
DOG_ID	varchar2(10)	Foreign Key referencing Dog	fkey
OCCUP_STATUS	varchar2(5)	Valid - ('YES','yes','NO','no')	t_or_n

STAFF

Column	Data type (size)	Constraint	Constraint Name
STAFF_ID	varchar2(4)	Primary Key	
STAFF_FNAME	varchar2(20)		
STAFF_LNAME	varchar2(20)		
STAFF_PHONE_NO	varchar2(14)	Starts with ‘+91-’	staff_phone_start_with_91
STAFF_GMAILID	varchar2(30)	Ends with ‘@gmail.com’	staff_idEndWith_gmail_dot_com
DESIGNATION	varchar(20)	Valid - 'janitor','vet','animal_caretaker','receptionist','animal_trainer','adoption_manager','shelter_manager'	desigin
SUPERV_ID	varchar2(8)	Unique	
SALARY_PER_ANNUM	number(10,2)		
BEGIN_DATE	DATE		
END_DATE	DATE		

EVENTS

Column	Data type (size)	Constraint	Constraint Name
EV_TYPE	varchar2(30)	Valid - 'adoption','surrender'	giveortake
DOG_ID	varchar2(10)		
EV_DATE	DATE		
VISITOR_ID	varchar(4)	Unique	
EV_ID	varchar(3)		
STAFF_ID	varchar2(4)		

DOG

Column	Data type (size)	Constraint	Constraint Name
DOG_ID	varchar2(10)	Primary Key	
DOG_NAME	varchar2(10)		
EST_AGE	number(2)	0 to 20	ageYear_between
COLOR	varchar2(20)		
BREED	varchar2(30),		
DATE_OF_ARRIVAL	DATE		
SEX	char(1)	Valid - 'F','f' ,'m','M’	sex_inF_or_M
SPAYED_OR_NEUTERED	varchar2(3)	Valid - ‘Y’,’N’	SPAYED_OR_NEUTERED_yes_or_no
WEIGHT_IN_KG	number(5,2)		
MIX	char(1)	NOT NULL	

VISITOR

Column	Data type (size)	Constraint	Constraint Name
VIS_FNAME	varchar2(20)		
PHONE_NO	varchar2(14)	Starts with ‘+91-’	phone_start_with_91
GMAIL_ID	varchar2(30)	Ends with '%@gmail.com'	idEndWith_gmail
ADDRESS	varchar(50)		
INTERACTION_DATE	DATE		
VISITOR_ID	varchar(4)	Primary Key	
VIS_LNAME	varchar2(20)		
DOG_ID	varchar2(10)		

4.4 Relational Model Implementation

Create Table Commands

```
CREATE TABLE DOG (

        DOG_ID varchar2(10),

        DOG_NAME varchar2(10),

        EST_AGE number(2) constraint ageYear_between check (EST_AGE between 0 and 20),

        COLOR varchar2(20),

        BREED varchar2(30),

        DATE_OF_ARRIVAL DATE,

        SEX char(1) constraint sex_inF_or_M check(SEX in('F','f' ,'m','M')),

        SPAYED_OR_NEUTERED varchar2(3) constraint SPAYED_OR_NEUTERED_yes_or_no
check(SPAYED_OR_NEUTERED in ('Y','N')),

        WEIGHT_IN_KG number(5,2),

        MIX char(1) constraint MIX_True_or_False check (MIX in ('T', 't', 'f', 'F')) NOT NULL,

        PRIMARY KEY (DOG_ID)

);


CREATE TABLE CAGE

(

        CAGE_NO number(5),

        DOG_ID varchar2(10) UNIQUE,

        OCCUP_STATUS varchar2(5) constraint t_or_n check(OCCUP_STATUS
in('YES','yes','NO','no')),

        constraint fkey FOREIGN KEY (DOG_ID) references Dog

);
```

```
);

CREATE TABLE STAFF

(

        STAFF_ID char(4),

        STAFF_FNAME varchar2(20),

        STAFF_LNAME varchar2(20),

        STAFF_PHONE_NO varchar2(14) constraint staff_phone_start_with_91
check(STAFF_PHONE_NO LIKE   '+91-__________'),

        STAFF_GMAILID varchar2(30) constraint staff_idEndWith_gmail_dot_com
check(STAFF_GMAILID LIKE '%@gmail.com') ,

        DESIGNATION varchar(20) constraint desigin check (DESIGNATION in
('janitor','vet','animal_caretaker','receptionist','animal_trainer','adoption_manager','shelter_manager')) ,

        SUPERV_ID varchar2(8) UNIQUE,

        SALARY_PER_ANNUM number(10,2),

        BEGIN_DATE DATE,

        END_DATE DATE,

        PRIMARY KEY (STAFF_ID)

);

CREATE TABLE EVENTS

(

        EV_ID varchar(3),

        EV_TYPE varchar(30) constraint giveortake check(EV_TYPE in ('adoption','surrender')) ,

        DOG_ID varchar2(10),

        EV_DATE DATE,

        VISITOR_ID varchar(4),

        UNIQUE (VISITOR_ID, EV_ID)

 );
```

```
CREATE TABLE VISITORS

(

  VIS_FNAME varchar2(20),

  PHONE_NO varchar2(14) constraint phone_start_with_91 check(PHONE_NO LIKE
'+91-__________'),

  GMAIL_ID varchar2(30) constraint idEndWith_gmail check (GMAIL_ID LIKE '%@gmail.com'),

  ADDRESS varchar2(50),

  INTERACTION_DATE DATE,

  VISITOR_ID varchar(4),

  VIS_LNAME varchar2(20),

  DOG_ID varchar2(10),

  PRIMARY KEY (VISITOR_ID)

);
```

INSERT COMMANDS

-- DOG TABLE INSERTIONS

```
insert into DOG VALUES('dog10000','bruno',5,'light brown','husky',TO_DATE('01-02-2020',
'DD-MM-YYYY'),'M','Y',20,'F');

insert into DOG VALUES('dog10001','julie',6,'black','labrador', TO_DATE('12-02-2021',
'DD-MM-YYYY'),'F','Y',18,'F');

insert into DOG VALUES('dog10002','pluto',1,'black and white','dalmation',TO_DATE('24-11-2020',
'DD-MM-YYYY'),'M','Y',10,'F');

insert into DOG VALUES('dog10003','jackie',2,'brown','labradoodle', TO_DATE('04-02-2019',
'DD-MM-YYYY'),'M','Y',12,'T');

insert into DOG VALUES('dog10004','jilka',4,'golden brown','pug',TO_DATE('10-10-2021',
'DD-MM-YYYY'),'F','Y',12,'F');
```

-- CAGE TABLE INSERTIONS

```
insert into CAGE VALUES(00000,'dog10002','YES');

insert into CAGE VALUES(00001,'dog10001','YES');

insert into CAGE VALUES(00011,'dog10003','YES');

insert into CAGE VALUES(00111,'dog10004','YES');

insert into CAGE VALUES(10000,'dog10000','YES');

insert into CAGE VALUES(10001, NULL,'NO');

insert into CAGE VALUES(11001, NULL,'NO');
```

-- STAFF TABLE INSERTIONS

```
insert into STAFF VALUES('s400','raj','singh','+91-9123465456',
'rajsinghing88@gmail.com','vet','s412',50000,TO_DATE('01-01-2019', 'DD-MM-YYYY'),NULL);

insert into STAFF
VALUES('s401','rahul','kohli','+91-9988998855','rahul776@gmail.com','janitor','s413',10000,
TO_DATE('01-01-2019', 'DD-MM-YYYY'), TO_DATE('01-01-2020', 'DD-MM-YYYY'));
```

```sql
insert into STAFF VALUES('s402','rania','Hegde','+91-7654321211',
'rania33@gmail.com','receptionist','s414',30000,TO_DATE('20-01-2019', 'DD-MM-YYYY'),NULL);

insert into STAFF VALUES('s406','mahesh','arjun','+91-9945598858',
'mahesh776@gmail.com','janitor','s415',10000,TO_DATE('02-01-2019', 'DD-MM-YYYY'),NULL);

insert into STAFF
VALUES('s410','arpita','pillai','+91-8945666858','arpita876@gmail.com','adoption_manager','s416',20000,
TO_DATE('04-06-2022', 'DD-MM-YYYY'),NULL);

insert into STAFF VALUES('s412','arohi','patel','+91-9900008858',
'arohi54@gmail.com','shelter_manager','s417',500000,TO_DATE('02-01-2019',
'DD-MM-YYYY'),NULL);
```

-- EVENTS TABLE INSERTIONS

```sql
INSERT INTO EVENTS VALUES('001','surrender','dog10000', TO_DATE('31-01-2020',
'DD-MM-YYYY'),'v100');

INSERT INTO EVENTS VALUES('002','adoption','dog10001', TO_DATE('20-06-2022',
'DD-MM-YYYY'),'v102');

INSERT INTO EVENTS VALUES('003','adoption','dog10002',TO_DATE('14-03-2022',
'DD-MM-YYYY'),'v115');

INSERT INTO EVENTS VALUES('004','surrender','dog10003',TO_DATE('03-02-2019',
'DD-MM-YYYY'),'v103');

INSERT INTO EVENTS VALUES('005','adoption','dog10004',TO_DATE('12-10-2021',
'DD-MM-YYYY'),'v166');
```

-- VISITOR TABLE INSERTIONS

```sql
INSERT INTO VISITORS VALUES('Aruna','+91-2378456789','aruna28@gmail.com','A101, Mandavi
Towers,Udupi',TO_DATE('31-01-2020', 'DD-MM-YYYY'),'v100','Pai','dog10000');

INSERT INTO VISITORS VALUES('Shristi','+91-7895645545','shristi11@gmail.com','B101, Mandavi
Towers,Udupi',TO_DATE('14-03-2022', 'DD-MM-YYYY'),'v115','Acharya','dog10002');

INSERT INTO VISITORS VALUES('Shilpa','+91-9800676876','shettyshilpa778@gmail.com','ab5,
blueberry woods ,Manipal',TO_DATE('12-10-2021', 'DD-MM-YYYY'),'v166','Shetty','dog10004');
```

INSERT INTO VISITORS VALUES('Arjun','+91-6980746545','arjunarjun568@gmail.com','c88, prime buildings ,Ajjarkhad',TO_DATE('20-06-2022', 'DD-MM-YYYY'),'v102','Nayak','dog10001');

INSERT INTO VISITORS VALUES('Siddharth','+91-9899823415','sid24shenoy@gmail.com','a102,Raj Towers,Manipal',TO_DATE('03-02-2019', 'DD-MM-YYYY'),'v103','Shenoy','dog10003');

4.5 Queries

List of queries used to retrieve data

4.5.1 display huskies below 6 years of age .

SELECT dog_id, dog_name from dog where breed ='husky' and est_age<6;

4.5.2 list empty cage numbers

SELECT   cage_no from cage where occup_status='NO';

4.5.3 display surrender records .

SELECT d.dog_name , e.dog_id , e.EV_TYPE,visitor.vis_fname from dog as d join events as e on d.dog_id=e.dog_id Join (visitor on visitor.dog_id=e.dog_id where e.EV_TYPE='surrender');

4.5.4 display adoption records.

SELECT d.dog_name , e.dog_id , e.EV_TYPE,visitor.vis_fname from dog as d join events as e on d.dog_id=e.dog_id Join visitor on visitor.dog_id=e.dog_id where e.EV_TYPE='adopted';

4.5.5 display details of staff who earn more than 50000

SELECT STAFF_ID , STAFF_FNAME from staff where SALARY_PER_ANNUM >=50000;

4.7 Triggers

```
CREATE OR REPLACE TRIGGER display_and_check_salary_changes

BEFORE UPDATE ON STAFF

FOR EACH ROW

DECLARE

   sal_diff number;

   sal_diff_percentage number;

BEGIN

   sal_diff := :NEW.SALARY_PER_ANNUM  - :OLD.SALARY_PER_ANNUM;

   dbms_output.put_line('UPDATING SALARY'||sal_diff_percentage);

            if (sal_diff < 0) THEN

            dbms_output.put_line('NOT allowed, get authorization for decreasing salary');

            raise_application_error(-20001,'NOT allowed, get authorization for decreasing salary');

            else

            dbms_output.put_line('UPDATING SALARY');

            dbms_output.put_line('New salary: ' || :NEW.SALARY_PER_ANNUM);

            dbms_output.put_line('Salary difference: ' || sal_diff);

            dbms_output.put_line('Old salary: ' || :OLD.SALARY_PER_ANNUM);

            end if;

END;

/
```

4.8 Stored Procedures

Procedure to fetch information of visitor from dogid

```
CREATE OR REPLACE procedure display_visitor_info (input_dog_id IN varchar) IS

        v_visitor visitors%rowtype;

        v_event events%rowtype;

BEGIN

   select * into v_event from events where dog_id = input_dog_id;

   select * into v_visitor from visitors where visitor_id = v_event.visitor_id;


          DBMS_OUTPUT.PUT_LINE('Visitor Name:' || v_visitor.VIS_FNAME || ' ' ||
v_visitor.VIS_LNAME);

   DBMS_OUTPUT.PUT_LINE('Visitor Phone Number:' || v_visitor.PHONE_NO);

   DBMS_OUTPUT.PUT_LINE('Visitor gmail ID:' || v_visitor.GMAIL_ID);

   DBMS_OUTPUT.PUT_LINE('Visitor Address:' || v_visitor.ADDRESS);

END;

/

exec display_visitor_info('dog10000');
```

Procedure to find dog details by taking dogId as user input

```
CREATE OR REPLACE procedure dogdeets(dogid IN varchar2) IS

v_dog dog%ROWTYPE;
```

```
BEGIN

  select * into v_dog from dog where DOG_ID=dogid;

  dbms_output.put_line('dog id is: '||v_dog.DOG_ID);

  dbms_output.put_line('dog name is: '||v_dog.DOG_NAME);

  dbms_output.put_line('dog age is: '||v_dog.EST_AGE);

  dbms_output.put_line('dog color is: '||v_dog.COLOR);

  dbms_output.put_line('dog breed is: '||v_dog.BREED);

  dbms_output.put_line('the dogs arrival date is: '||  v_dog.DATE_OF_ARRIVAL);

  dbms_output.put_line('dog id is: '||v_dog.SEX);

  dbms_output.put_line('dog id is: '||v_dog.SPAYED_OR_NEUTERED);

  dbms_output.put_line('dog id is: '||v_dog.WEIGHT_IN_KG);

EXCEPTION

    WHEN NO_DATA_FOUND THEN

            DBMS_OUTPUT.PUT_LINE ('DOGID not found.');

END dogdeets;

/



BEGIN

            dogdeets('dog10001');

END;

/
```

Procedure to generate company email ids of staff automatically using staff name,staffid as name-staffid@myshelter.com

```
cursor v_staff is select * from staff;

BEGIN

        for s in v_staff

        loop

        dbms_output.put_line('Email address of ' || s.STAFF_FNAME || ' is: ' ||s.STAFF_FNAME||'_'|s.STAFF_ID||'@myshelter.com');

end loop;

END;

/
```

4.9 Stored Functions

Functions to fetch number of dogs adopted and surrendered during a monthly period (specified by start_date and end_date):

Number of Dogs Surrendered:

```
CREATE OR REPLACE function arrival_count (start_date in DATE, end_date in DATE) Return number IS

        num_arrival number(3);

BEGIN

        select count(ev_id) into num_arrival from events where EV_TYPE = 'surrender' and TO_DATE(ev_date, 'DD-MM-YYYY') between start_date and end_date;
        return num_arrival;
END;
/
```

–PL/SQL block to execute function:

```
DECLARE
        start_date DATE := TO_DATE('03-FEB-18', 'DD/MM/YYYY');
        end_date DATE := TO_DATE('14-MAR-22', 'DD/MM/YYYY');
        arr_num number(3);
BEGIN
        arr_num := arrival_count(start_date, end_date);
        DBMS_OUTPUT.PUT_LINE('Number of dogs surrendered between ' ||
start_date || ' and ' || end_date || ': ' || arr_num);
END;
```

Number of Dogs Adopted:

```
CREATE OR REPLACE function arrival_count (start_date in DATE, end_date in
DATE) Return number IS
        num_arrival number(3);
BEGIN
        select count(ev_id) into num_arrival from events where EV_TYPE =
'adoption' and TO_DATE(ev_date, 'DD-MM-YYYY') between start_date and
end_date;
        return num_arrival;
END;
```

/

–PL/SQL block to execute function:

```
DECLARE

        start_date DATE := TO_DATE('03-FEB-18', 'DD/MM/YYYY');

        end_date DATE := TO_DATE('14-MAR-22', 'DD/MM/YYYY');

        arr_num number(3);

BEGIN

        arr_num := arrival_count(start_date, end_date);

        DBMS_OUTPUT.PUT_LINE('Number of dogs adopted between ' ||
start_date || ' and ' || end_date || ': ' || arr_num);

END;
```

Function to fetch number of dogs spayed or neutered during a monthly period (specified by start_date and end_date):

```
CREATE OR REPLACE function spay_neuter_count (start_date in DATE,
end_date in DATE) Return number IS

        num_neutered number(3);

BEGIN

        select count(distinct dog_id) into num_neutered from events natural join dog
where spayed_or_neutered = 'Y' and TO_DATE(ev_date, 'DD-MM-YYYY')
between start_date and end_date;

        return num_neutered;

END;

/
```

–PL/SQL block to execute function:

```
DECLARE
        start_date DATE := TO_DATE('03-FEB-19', 'DD/MM/YYYY');
        end_date DATE := TO_DATE('14-MAR-22', 'DD/MM/YYYY');
        num_neutered number(3);
BEGIN
        num_neutered := spay_neuter_count (start_date, end_date);
        DBMS_OUTPUT.PUT_LINE('Number of dogs neutered or spayed between
' || start_date || ' and ' || end_date || ': ' || num_neutered);
END;
```

8. Conclusion and Future Work

8.1 Conclusion

The growing populations of stray dogs in India remain a problem to this day and demand a well-organized effort to protect the health of both humans as well as strays in our communities. This can be effectively aided by implementing a database management system for maintaining information essential to the functioning of dog or animal shelters. By creating functions and procedures in this system to allow for efficient data collection and retrieval, shelters will be able to frequently and transparently provide evidence to the donors who allow such establishments to continue providing service.

8.2 Scope for future work

This database is built for one branch, this can be extended to a larger database including several branches across the country. This would allow for better tracking and understanding of the stray dogs in our community and ways to handle them. The database itself can be improved by providing stricter guidelines for inserting data. For example the breed column can be its own table, which is a better way to handle the multiple pieces of data available. The security can be improved by setting restrictions for users, administrators and such.