

DOGGY DAYCARE

Database Design Project

Jillian Preece

TABLE OF CONTENTS

Executive Summary.....	3
Entity-Relationship Diagram.....	4
Tables.....	5
Views.....	23
Reports.....	27
Stored Procedures.....	32
Triggers.....	35
Security.....	37
Implementation Notes.....	41
Known Problems.....	42
Future Enhancements.....	43



EXECUTIVE SUMMARY

The purpose of this database is to store the records for a Doggy Daycare. It will store all people associated with the daycare, including dog owners and staff, the rooms and stays at the daycare, and most importantly the dogs and all of their attributes.

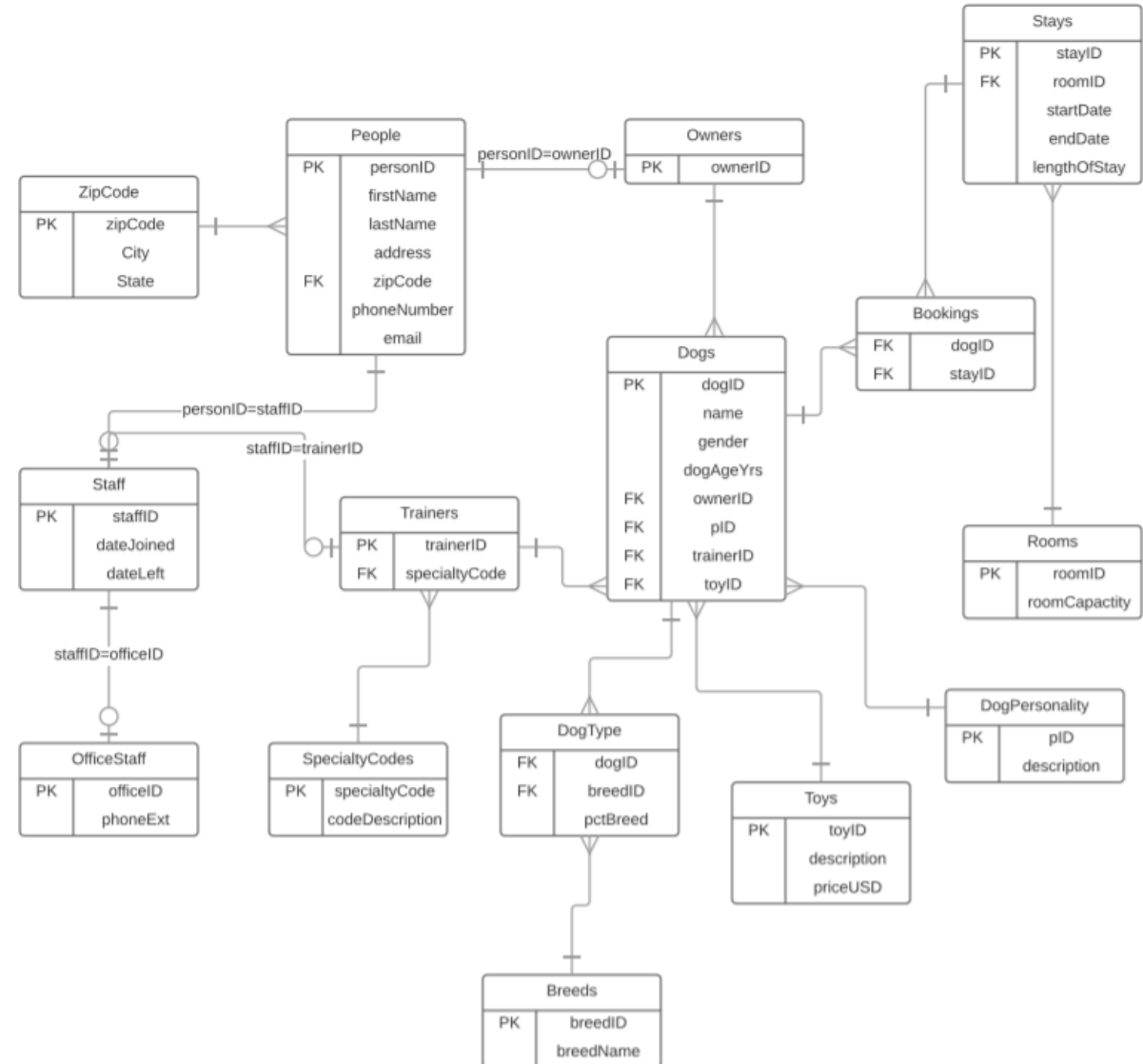
As dogs are often part of the family, it is understood that all records must be careful and accurate to make the owners feel safe with boarding their dogs at the daycare. It is very important for this database to be accurate and secure, as well as have high attention to detail with each dog. Each dog will be assigned a specific trainer that is compatible with their personality, and the use of this database will make the assignments much easier.

Since the Doggy Daycare is just beginning, this database is intended for the company to store all of their new information so that appropriate records can be kept from this day forward.

ENTITY-RELATIONSHIP DIAGRAM



Murphy, the inspiration for this project



TABLES

ZIP CODES

Lists all zip codes with their respective city and state

```
DROP TABLE IF EXISTS zipCodes cascade;  
CREATE TABLE ZipCodes (  
    zipCode          int NOT NULL,  
    city             text NOT NULL,  
    state            text NOT NULL,  
    primary key (zipCode)  
);
```

Functional Dependencies

zipCode → city, state

zipcode integer	city text	state text
18064	Nazareth	Pennsylvania
12106	Kinderhook	New York
16450	Meriden	Connecticut
19992	Danbury	Connecticut
33004	Hollywood	California
28754	Marion	North Carolina
11735	Farmingdale	New York
11572	Oceanside	New York
48185	Westland	Michigan
64151	Kansas City	Missouri
24073	Christiansburg	Virginia
60062	Northbrook	Illinois
48174	Romulus	Michigan

PEOPLE

Lists all people involved with the daycare and their basic attributes

```
DROP TABLE IF EXISTS People cascade;  
CREATE TABLE People (
```

personID	char(3) NOT NULL,
firstName	text NOT NULL,
lastName	text NOT NULL,
address	text NOT NULL,
zipCode	int NOT NULL references ZipCodes(zipCode),
phoneNumber	text NOT NULL,
email	text,

```
primary key (personID)  
);
```

Functional Dependencies

personID → firstName, lastName, address, zipCode, phoneNumber, email

personid character (3)	firstname text	lastname text	address text	zipcode integer	phonenumber text	email text
p01	Jonny	Noxon	109 Oxford Street	18064	748-384-2838	jonny.noxon13@aol.com
p02	Ted	Tuttle	456 North Lane	12106	929-495-2038	teddytut@yahoo.com
p03	Kimberly	Preece	65 West Oak Street	16450	473-327-4532	ilovedogs@dog.net
p04	Faity	Kearns	3 Devonshire Drive	19992	939-214-5832	faityk1994@gmail.com
p05	Jennifer	Aniston	444 Hollywood Boulevard	33004	438-294-3820	rachelgreen@hotmail.com
p06	Frederick	Slayton	9007 Sunbeam Court	28754	192-438-2340	freddieslay@gmail.com
p07	Anne	Matheus	325 Boston Street	11735	214-542-5784	cococorzine@yahoo.com
p08	Deb	Rothwell	8203 Foster Street	28754	789-243-6453	debroth@snet.net
p09	Gaston	Jones	390 Pulaski Avenue	11572	523-643-6859	gjones@yahoo.com
p10	Dylan	Quincy	46 Rockville Avenue	48185	890-557-4308	dqblizzard@yahoo.com
p11	Lacy	Goodson	7741 Jennings Road	64151	432-524-8941	
p12	Neil	Patrick Harris	163 Gartner Drive	24073	903-312-5429	barneystinson@himym.com
p13	Shelly	Marger	249 Acacia Circle	60062	329-325-8904	shellymargs@aol.com
p14	Taylor	Murray	4320 Annadale Road	11572	573-984-2149	tmurray@gmail.com
p15	Rick	Sanchez	577 Logan Street	48174	471-987-0426	rickysanchez@gmail.com

STAFF

Subtype of People, lists people that are staff and their specific attributes

```
DROP TABLE IF EXISTS Staff cascade;
```

```
CREATE TABLE Staff (
```

```
    staffID
```

```
    dateJoined
```

```
    dateLeft
```

```
    primary key (staffID)
```

```
);
```

char(3) NOT NULL references People(personID),
date NOT NULL,
date,

Functional Dependencies

staffID → dateJoined, dateLeft

staffid character (3)	datejoined date	dateleft date
p04	2017-01-24	[null]
p06	2017-01-13	[null]
p09	2017-01-26	[null]
p10	2017-02-01	[null]
p11	2017-01-13	[null]
p13	2017-01-13	[null]
p14	2017-01-20	[null]

SPECIALTY CODES

Lists the different specialties a trainer can have

```
DROP TABLE IF EXISTS specialtyCodes cascade;  
CREATE TABLE specialtyCodes (  
    specialtyCode          char(3) NOT NULL,  
    codeDescription        text NOT NULL,  
    primary key (specialtyCode)  
);
```

Functional Dependencies

specialtyCode → codeDescription

specialtycode character (3)	codedescription text
s01	obedience
s02	small dogs
s03	big dogs
s04	dog tricks

TRAINERS

Lists the staff that are trainers and their specialty code

DROP TABLE IF EXISTS Trainers cascade;

CREATE TABLE Trainers (

 trainerID

 specialtyCode

 primary key (trainerID)

);

Functional Dependencies

trainerID → specialtyCode

char(3) NOT NULL references People(personID),
char(3) NOT NULL references SpecialtyCodes(specialtyCode),

trainerid character (3)	specialtycode character (3)
p06	s04
p09	s01
p10	s02
p13	s01
p14	s03

OFFICE STAFF

Lists staff that work in the office and their phone extension, if they have one

```
DROP TABLE IF EXISTS OfficeStaff cascade;  
CREATE TABLE OfficeStaff (  
    officeID          char(3) NOT NULL references People(personID),  
    phoneExt          int,  
    primary key (officeID)  
);
```

Functional Dependencies
officeID → phoneExt

officeid character (3)	phoneext integer
p04	1313
p11	8374

OWNERS

Lists the people that are dog owners

DROP TABLE IF EXISTS Owners cascade;

CREATE TABLE Owners (

ownerID

char(3) NOT NULL references People(personID),

primary key(ownerID)

);

No functional dependencies exist within this table

ownerid character (3)
p01
p02
p03
p05
p06
p07
p08
p11
p12
p14
p15

ROOMS

Lists all of the rooms that dogs can stay in and their capacity

```
DROP TABLE IF EXISTS Rooms cascade;
```

```
CREATE TABLE Rooms (
```

```
    roomID
```

```
    roomCapacity
```

```
    primary key (roomID)
```

```
);
```

```
    char(3) NOT NULL,  
    int NOT NULL,
```

Functional Dependencies

roomID → roomCapacity

roomid character (3)	roomcapacity integer
r01	2
r02	1
r03	2
r04	1
r05	1
r06	1
r07	2
r08	4
r09	2
r10	2

STAYS

Lists each stay at the daycare, including the room, start date, and end date

```
DROP TABLE IF EXISTS Stays cascade;
```

```
CREATE TABLE Stays (
```

```
    stayID
```

```
    roomID
```

```
    startDate
```

```
    endDate
```

```
    lengthOfStay
```

```
    primary key (stayID)
```

```
);
```

```
    char(3) NOT NULL,
```

```
    char(3) NOT NULL references Rooms(roomID),
```

```
    date NOT NULL,
```

```
    date NOT NULL CHECK (endDate > startDate),
```

```
    int NOT NULL,
```

Functional Dependencies

stayID → roomID, startDate, endDate, lengthOfStay

stayid character (3)	roomid character (3)	startdate date	enddate date	lengthofstay integer
s01	r08	2017-01-20	2017-01-28	8
s02	r01	2017-01-21	2017-01-27	6
s03	r06	2017-01-22	2017-02-05	14
s04	r02	2017-01-23	2017-02-01	9
s05	r03	2017-01-24	2017-01-29	5
s06	r07	2017-01-25	2017-02-09	15
s07	r09	2017-01-26	2017-02-06	11
s08	r05	2017-01-27	2017-02-01	5
s09	r10	2017-01-28	2017-02-02	5
s10	r04	2017-01-29	2017-02-03	5
s11	r02	2017-02-02	2017-02-10	8
s12	r05	2017-02-02	2017-02-07	5

BREEDS

Lists all breeds of dogs who stay at the daycare

DROP TABLE IF EXISTS Breeds cascade;

```
CREATE TABLE Breeds (  
    breedID  
    breedName  
    primary key (breedID)  
);
```

char(3) NOT NULL,
text NOT NULL,

Functional Dependencies

breedID → breedName

breedid character (3)	breedname text
b01	Bernese Mountain Dog
b02	Bulldog
b03	German Shepherd
b04	Labrador Retriever
b05	Yorkshire Terrier
b06	Schnauzer
b07	Poodle
b08	Jack Russell
b09	Chihuahua
b10	Springer Spaniel
b11	Great Dane
b12	Newfoundland
b13	Dachshund
b14	St. Bernard
b15	Basset Hound

TOYS

Lists all toys at the daycare and their prices

```
DROP TABLE IF EXISTS Toys cascade;
```

```
CREATE TABLE Toys (
```

```
    toyID
```

```
    description
```

```
    priceUSD
```

```
    decimal,
```

```
    char(3) NOT NULL,  
    text NOT NULL,
```

```
    primary key (toyID)  
);
```

Functional Dependencies

toyID → description, priceUSD

toyid character (3)	description text	priceusd numeric
t01	stick	0.00
t02	stuffed fox butt	10.00
t03	giraffe	5.00
t04	rope	9.00
t05	bone	4.50
t06	squeaky taco	14.00
t07	sushi	15.00
t08	tennis ball	4.99
t09	peanut butter kong	10.99
t10	stuffed starbucks cup	22.50

DOG PERSONALITY

Lists personalities of dogs at the daycare

```
DROP TABLE IF EXISTS DogPersonality cascade;  
CREATE TABLE dogPersonality(  
    dpID                char(4),  
    description          text,  
    primary key (dpID)  
);
```

Functional Dependencies

dpID → description

dpid character (4)	description text
dp01	cuddly
dp02	disobedient
dp03	jumpy
dp04	playful
dp05	loner
dp06	feisty

DOGS

Lists the dogs who stay at the daycare, along with their specific attributes

```
DROP TABLE IF EXISTS Dogs cascade;
```

```
CREATE TABLE Dogs(
```

```
    dogID                char(3) NOT NULL,
```

```
    name                 text,
```

```
    gender               text CHECK (gender = 'M' OR gender = 'F'),
```

```
    ownerID              char(3) NOT NULL references Owners(ownerID),
```

```
    dpID                 char(4) NOT NULL references dogPersonality(dpID),
```

```
    trainerID            char(3) NOT NULL references Trainers(trainerID),
```

```
    toyID                char(3) NOT NULL references Toys(toyID),
```

```
    primary key (dogID)
```

```
);
```

Functional Dependencies

dogID → name, gender, ownerID, dpID, trainerID, toyID

dogid character (3)	name text	gender text	dogageyrs integer	ownerid character (3)	dpid character (4)	trainerid character (3)	toyid character (3)
d01	Murphy	M	3	p03	dp04	p09	t07
d02	Tequila	M	7	p01	dp01	p06	t02
d03	Diesel	M	4	p02	dp02	p13	t01
d04	Alan	M	3	p07	dp03	p06	t09
d05	Princess	F	8	p08	dp06	p10	t10
d06	Bonnie	F	3	p15	dp05	p06	t04
d07	April	F	6	p05	dp03	p09	t10
d08	Moose	M	10	p12	dp02	p09	t03
d09	Missy	F	6	p12	dp03	p13	t08
d10	Bentley	M	8	p06	dp03	p14	t04
d11	Godiva	F	7	p11	dp02	p13	t05
d12	Wrinkles	M	4	p03	dp01	p06	t10
d13	Kooza	M	6	p08	dp05	p14	t02
d14	Wallace	M	5	p14	dp02	p09	t07
d15	Reginald	M	3	p08	dp06	p14	t10
d16	Libby	F	8	p08	dp01	p06	t06
d17	Taquito	M	4	p01	dp04	p10	t03
d18	Chip	M	16	p07	dp06	p10	t06

DOG TYPE

Lists the percentage of a breed that a dog is

```
DROP TABLE IF EXISTS dogType cascade;
```

```
CREATE TABLE dogType (
```

```
    dogID          char(3) NOT NULL references Dogs(dogID),
```

```
    breedID        char(3) NOT NULL references Breeds(breedID),
```

```
    pctBreed       decimal,
```

```
    primary key (dogID, breedID)
);
```

Functional Dependencies

dogID, breedID → pctBreed

dogid character (3)	breedid character (3)	pctbreed numeric
d01	b01	1.0
d02	b08	0.50
d02	b10	0.50
d03	b04	1.0
d04	b06	0.30
d04	b07	0.70
d05	b05	1.0
d06	b14	1.0
d07	b12	0.50
d07	b01	0.50
d08	b12	0.20
d08	b11	0.30
d08	b01	0.50
d09	b13	0.52
d09	b08	0.48
d10	b11	1.0
d11	b04	0.50
d11	b03	0.50
d12	b02	1.0
d13	b07	0.75
d13	b12	0.25
d14	b10	0.60
d14	b15	0.40
d15	b07	1.0
d16	b06	0.50
d16	b08	0.50
d17	b09	1.0
d18	b13	1.0

BOOKINGS

Lists all stays with all dogs booked for the stay

DROP TABLE IF EXISTS Bookings cascade;

CREATE TABLE Bookings (

 dogID

 stayID

 primary key (dogID, stayID)

);

char(3) NOT NULL references Dogs(dogID),

char(3) NOT NULL references Stays(stayID),

No functional dependencies exist within this table

dogid character (3)	stayid character (3)
d05	s01
d13	s01
d15	s01
d16	s01
d08	s02
d09	s02
d03	s03
d10	s04
d02	s05
d17	s05
d04	s07
d18	s07
d14	s08
d01	s09
d12	s09
d06	s10
d07	s11
d11	s12

VIEWS

TRAINER_ROSTER

This view is intended to create a table for each trainer to look at and see what dogs they are training, and the personality of those dogs

```
CREATE VIEW Trainer_Roster AS
  SELECT d.dogid, d.name, dp.description
  FROM trainers t INNER JOIN people p ON t.trainerID = p.personID
               INNER JOIN dogs d ON t.trainerID = d.trainerID
               INNER JOIN dogpersonality dp ON dp.dpid = d.dpid
 WHERE d.trainerID = 'p13'
 ORDER BY d.dogid ASC;
```

dogid character (3)	name text	description text
d03	Diesel	disobedient
d09	Missy	jumpy
d11	Godiva	disobedient

OWNER_INFORMATION

This view is intended to create a table for the office staff to see the owners' contact information for each dog

```
CREATE VIEW Owner_Information AS
SELECT p.firstname, p.lastname, p.phonenumber, p.address, z.city, z.state, z.zipcode, d.name AS dogname
FROM dogs d INNER JOIN owners o ON d.ownerid = o.ownerid
      INNER JOIN people p ON o.ownerID = p.personid
      INNER JOIN zipcodes z ON p.zipcode = z.zipcode
WHERE d.dogid = 'd04';
```


firstname text	lastname text	phonenumber text	address text	city text	state text	zipcode integer	dogname text
Anne	Matheus	214-542-5784	325 Boston Street	Farmingdale	New York	11735	Alan

ROOMS_OCCUPIED

This view is intended for the office staff to be able to see what rooms are occupied at any given time.

```
CREATE VIEW Rooms_Occupied AS
SELECT d.name, r.roomID
      FROM bookings b INNER JOIN stays s ON b.stayid = s.stayid
                        INNER JOIN rooms r ON s.roomID = r.roomID
                        INNER JOIN dogs d ON b.dogID = d.dogID
     WHERE s.startDate <= '2017-02-02' AND s.endDate >= '2017-02-07';
```

name text	roomid character (3)
April	r02
Godiva	r05

A decorative border of black paw prints runs vertically along both the left and right edges of the page. The paw prints are stylized with four toes and a central pad.

REPORTS

NOT OWNERS

This lists all people associated with the daycare who are not owners (unfortunately for them)

```
SELECT p.firstName, p.lastName  
FROM people p LEFT JOIN owners o ON p.personID = o.ownerID  
WHERE ownerID IS NULL  
ORDER BY personID ASC;
```

firstname text	lastname text
Faity	Kearns
Gaston	Jones
Dylan	Quincy
Shelly	Marger

DOG BREEDS

This will return all dogs who are purebred at the daycare

```
SELECT d.name, b.breedname
FROM dogtype dt INNER JOIN dogs d on dt.dogID = d.dogID
              INNER JOIN breeds b on dt.breedID = b.breedID
WHERE dt.pctBreed = '1.0';
```

Optional modification: Will return all dogs who are more than 50% of one breed.

```
SELECT d.name, b.breedname
FROM dogtype dt INNER JOIN dogs d on dt.dogID = d.dogID
              INNER JOIN breeds b on dt.breedID = b.breedID
WHERE dt.pctBreed > '0.5';
```

name text	breedname text
Murphy	Bernese Mountain Dog
Diesel	Labrador Retriever
Princess	Yorkshire Terrier
Bonnie	St. Bernard
Bentley	Great Dane
Wrinkles	Bulldog
Reginald	Poodle
Taquito	Chihuahua
Chip	Dachshund

name text	breedname text
Murphy	Bernese Mountain Dog
Diesel	Labrador Retriever
Alan	Poodle
Princess	Yorkshire Terrier
Bonnie	St. Bernard
Missy	Dachshund
Bentley	Great Dane
Wrinkles	Bulldog
Kooza	Poodle
Wallace	Springer Spaniel
Reginald	Poodle
Taquito	Chihuahua
Chip	Dachshund

EXPENSIVE TOYS

This will return the breeds of dogs at the daycare that like expensive toys

```
SELECT b.breedName as BreedName, t.priceUSD as ToyPrice
FROM dogType dt INNER JOIN Dogs d ON dt.dogID = d.dogID
                INNER JOIN toys t ON t.toyID = d.toyID
                INNER JOIN breeds b ON dt.breedID = b.breedID
WHERE t.priceUSD > 10.00
GROUP BY b.breedName, t.priceUSD
ORDER BY t.priceUSD desc;
```

breedname text	toyprice numeric
Bernese Mountain Dog	22.50
Bulldog	22.50
Newfoundland	22.50
Poodle	22.50
Yorkshire Terrier	22.50
Basset Hound	15.00
Bernese Mountain Dog	15.00
Springer Spaniel	15.00
Dachshund	14.00
Jack Russell	14.00
Schnauzer	14.00
Poodle	10.99
Schnauzer	10.99

AVERAGE AGE BY OWNER

Lists all owners and the average age of the dogs they own

```
SELECT o.ownerID, AVG(dogAgeYrs)
FROM dogs d INNER JOIN owners o ON d.ownerID = o.OwnerID
      INNER JOIN people p ON o.ownerID = p.personID
GROUP BY o.ownerID
ORDER BY o.ownerID ASC;
```

ownerid character (3)	avg numeric
p01	5.50
p02	4.00
p03	3.50
p05	6.00
p06	8.00
p07	9.50
p08	6.25
p11	7.00
p12	8.00
p14	5.00
p15	3.00

A decorative border of black paw prints runs vertically along both the left and right edges of the page. The paw prints are arranged in a slightly irregular, hand-drawn style.

STORED PROCEDURES

SIBLINGS

This function will return all dogs that have the same owner

```
CREATE OR REPLACE FUNCTION Siblings(ownerID char(3))
RETURNS TABLE (OwnerName text, DogName Text) AS
$$
BEGIN
    RETURN QUERY SELECT d.name as DogName, p.firstName as OwnerName
                  FROM dogs d INNER JOIN owners o on d.ownerID = o.ownerID
                  INNER JOIN people p on p.personID = o.ownerID
                  WHERE o.ownerID is NOT NULL
                  ORDER BY p.firstName;
END;
$$ LANGUAGE plpgsql;
```

Sample input:
SELECT Siblings(p08);

DOGNAME

Makes sure that dogs being entered into the system cannot have the incorrect spelling of 'Alan'

```
CREATE OR REPLACE FUNCTION DogName()  
RETURNS TRIGGER AS $$  
BEGIN  
    IF NEW.name IS 'Allen' THEN  
        RAISE EXCEPTION 'Dog name cannot be ridiculously spelled version of Alan';  
    END IF;  
  
    RETURN NEW;  
  
END;  
$$ LANGUAGE plpgsql;
```



TRIGGERS

DOGNAME

```
CREATE TRIGGER DogName  
BEFORE INSERT OR UPDATE ON Dogs  
FOR EACH ROW  
EXECUTE PROCEDURE DogType();
```


Example Output:

```
INSERT INTO Dogs(dogID, name, gender, ownerID, dpID, trainerID, toyID)  
VALUES('d22', 'Allen', 'M', 'p08', 'dp04', 'p13', 't01');
```

[Data Output](#) [Explain](#) [Messages](#) [History](#)

```
ERROR: Dog name cannot be ridiculously spelled version of Alan  
CONTEXT: PL/pgSQL function dogname() line 4 at RAISE  
***** Error *****
```

```
ERROR: Dog name cannot be ridiculously spelled version of Alan  
SQL state: P0001  
Context: PL/pgSQL function dogname() line 4 at RAISE
```

A decorative border of black paw prints runs vertically along both the left and right edges of the page. The paw prints are arranged in a slightly irregular, hand-drawn style.

SECURITY

TRAINERS

Trainers have the ability to update a dog's personality or favorite toy, as they spend the most time with the dogs, and know if their temperament has changed after some training

```
CREATE ROLE dogTrainer;  
GRANT SELECT, UPDATE ON Dogs, Trainer_Roster  
TO dogTrainer;
```

ADMIN

Admin are defined as the owners of the Doggy Daycare. They have the right to do anything and everything with the database, including insert, update, delete, or select anything from any table

```
CREATE ROLE Admin;  
GRANT ALL ON ALL TABLES  
IN SCHEMA PUBLIC  
To Admin;
```

A decorative border of black paw prints runs vertically along both the left and right edges of the page. The paw prints are of varying sizes and are arranged in a slightly irregular, hand-drawn style.

OTHER DATABASE INFORMATION

IMPLEMENTATION NOTES

This Doggy Daycare just opened in January, and is still relatively small, which accounts for the small number of dogs.

The Stays table provides a snapshot of the stays from the end of January to the beginning of February, and doesn't account for just one-day stays.

The percentages of breed for each dog can clearly not be 100% accurate due to the fact that the daycare doesn't perform DNA testing, but they are entered exactly as the owner has said.

The Bookings table may seem unnecessary, but it is important to remember that one person can book one stay that encompasses multiple dogs in the same room, creating the need for a Bookings table.

KNOWN PROBLEMS

For some reason, the function Siblings() SELECT statement doesn't return any values, and after much troubleshooting there was no solution that worked.

There is not a simple query to get an owners name, the most simple query that can be written involves an inner join on Owners and People.

Giving trainers Update and Select rights on the Dogs table could be risky, as they can update any part of a dog's attributes.

FUTURE ENHANCEMENTS

The age of a dog should be stored as the dog's date of birth and then calculated based off of the year. As it stands now, the age of a dog needs to be updated every year.

There should be a way to allow for one-day stays within the database, without entering them into the Stays table, because a one-day stay at the daycare doesn't require a room, and is not as important as overnight stays.

More check constraints should be implemented to ensure accurate data entry.