Database Project Design – Final Report

Project Group 17 - Aidan Kierans, Kenneth Richardson, Sreepradha Sreekesh

This database is designed to be used at an animal shelter, based on the needs of real animal shelters.

Requirements: A shelter needs to keep track of the animals that come in and people involved at the shelter. The animals can be adopted or reserved by customers while the employees care for the animals. Like many shelters, ours only accept cats and dogs.

There are different types of People involved in the animal shelter. Customers go to the shelter and eventually adopt the animals whereas the employees have different titles defining their role within the shelter. The two main departments in the shelter are medical and non-medical. Non-medical staff have titles like Admission, Kennel, and Adoption—the employees in these roles intake animals, take care of all of the animals' non-medical needs, and interact with potential adopters for dogs and cats.

The medical staff treats the animals as well as provides them with prescriptions if needed. A medical staff person will either have a certification (in which case their title will likely be Veterinarian or Veterinary Anesthesiologist) or not (in which case their title will likely be a Veterinary Technician or Veterinary Assistant). Depending on the level of certification i.e. certificate, degree, or license, the medical staff will treat the animals based on their medical needs and keeps track of these treatments and prescribe any required medications.

The users of the database include all of the staff of the shelter. The non-medical staff would use the database to process animal intake and to find the right animal match for a potential adopter. The database would also be used by non-medical staff to keep track of all the animals that have been adopted out, to periodically check with customers to see how their animals are doing, and/or to verify that a person is allowed to adopt animals from this shelter in case they have a history of mistreating animals. Additionally, the database could track all of the information that might be interesting or important to know about an animal, such as unusual needs (medical or otherwise), behavioral issues, whether it's okay to keep the animal with other animals or with kids, and the animal's medications. Any history of biting people would also be important to know so that they could track which animals are potentially dangerous, to find out who might have been exposed to a pathogen the animal is carrying, or simply for legal/insurance purposes.

The medical staff would use the database to maintain a comprehensive record of an animal's health and the care it has received. With that information, the medical staff would be able to diagnose and give proper treatment based on current symptoms as well as historical data of the given animal. For example, if an animal is coughing, a medical employee could query the database to find out whether that animal's previously recorded symptoms include the word "cough".

Advanced features: The database also includes several advanced features such as views, triggers, events, and procedures.

- Views: Two views were used to limit the amount of private information visible to those who shouldn't be able to see it, and one view was used to easily show which pairs of animals/people are compatible for adoption.
- Triggers: Since check constraints aren't in MySQL 5.7, which is the version used by Google Cloud Platform, most of the triggers we used were to keep the data consistent, by preventing future dates from being recorded as if they were in the present/past. Two triggers were to keep emails formatted as specified, and one was used to log updates to the employee table.
- Events: An event was used to check every day whether any of the reservations were more than six months old, and delete those that were.
- Procedures: A function was used to simplify the process of checking whether a particular animal was compatible with a particular customer, and a procedure was used to simplify the process of recording when an animal bit a human.

Security: Each employee has a username and a sha256 encrypted password. Since SQL roles aren't implemented in MySQL 5.7, we were unable to grant privileges the way we wanted to, so we had to assign the appropriate privileges for each job to each person who had that job. We experimented with creating a pseudo-role procedure for each job, so that a user with that job could be granted all of the right privileges by calling the procedure for their job, but doing so would require that the caller specify the username *and* the host, or the username *and* the password. Neither option felt right, so we scrapped the idea and simply formatted the GRANT statements (in the import script) into blocks that could be easily copied/pasted/edited as needed.

Entity sets:

- Animal
- Cat
- Cat_Immunizations
- Dog
- Dog_Immunizations
- Person

- Customer
- Employee
- Employee_Info_Changes
- Medical

Operations for maintaining the database:

- Increment n_times when a particular animal bites a particular human.
- Insert the reservation data for an animal that was reserved.
- Insert the adoption data for an animal that was adopted.
- Update the adoption data for a pet that was returned to the shelter.
- Insert the treatment data when the medical staff treats an animal.
- Insert the prescription, and the date it was written when the medical staff prescribes something for an animal.

Queries:

```
--Query 1
--Which persian cat has been here the longest?


SELECT e.id_number, e.name, MIN(e.date_sheltered) AS dateSheltered
FROM animal e, cat d
WHERE e.id_number = d.id_number
       AND breed = 'persian'
GROUP BY e.id_number, e.name;


--Query 2
--List the people that have adopted more than two dogs in the shelter


SELECT first_name, last_name, COUNT(id_number) AS numAdopted
FROM person JOIN adopted
WHERE person.human_id = adopted.human_id
GROUP BY first_name, last_name
HAVING numAdopted > 2;
```

```
--Query 3
--List the animals that have more than 1 recorded bite & are not compatible
with children

SELECT f.name, e.n_times, d.should_avoid
FROM animal f, bit e, incompatible d
WHERE e.id_number = d.id_number = f.id_number
AND (e.n_times > 1  AND d.should_avoid = "small_children");


--Query 4
--List the beagles that have been admitted in the past 2 years


SELECT name, date_sheltered, TIMESTAMPDIFF(YEAR, date_sheltered, CURDATE())
AS yearsAdmitted, breed
FROM animal JOIN dog
WHERE animal.id_number = dog.id_number
GROUP BY name, date_sheltered, breed
HAVING yearsAdmitted >= 2 AND breed = "beagle";


--Query 5
--List the animals currently at the shelter that came from RACC


SELECT name, origin FROM animal
WHERE animal.id_number NOT IN (SELECT id_number FROM adopted)
       AND origin = "RACC";

--Query 6
--List the medications has dog Martin been prescribed in the past 6 months.


SELECT name, prescription, date_written,
       TIMESTAMPDIFF(MONTH, date_written, CURDATE()) AS monthsAgo
FROM prescribed JOIN animal
WHERE animal.id_number = prescribed.id_number
GROUP BY name, date_written, prescription
HAVING monthsAgo >=6 AND name = "Martin";


--Query 7
```

```
--List the german shepard dogs that have been adopted out in the past month
& the people that adopted them


SELECT human_id, name, breed,start_date,
       TIMESTAMPDIFF(MONTH, start_date, CURDATE()) AS monthsAgo
FROM adopted JOIN dog JOIN animal
WHERE adopted.id_number = dog.id_number = animal.id_number
GROUP BY human_id, name, breed, start_date
HAVING monthsAgo <= 1;


--Query 8
--List the people that have adopted cat Thomas O'Maley


SELECT human_id, start_date, name
FROM adopted JOIN animal
WHERE name = "Thomas O Maley";


--Query 9
--List the dogs that are female, less than 1 year old, less than 30 lbs,
and have been prescribed medicine or treated by a vet.


SELECT name, gender, weight,
       TIMESTAMPDIFF(YEAR, birth_date, CURDATE()) AS Age,
       t_date, date_written
FROM animal JOIN dog JOIN treated JOIN prescribed
WHERE animal.id_number = dog.id_number = treated.id_number =
prescribed.id_number
GROUP BY name, age, gender, weight, t_date, date_written
HAVING gender = "female" AND Age < 1 AND weight < 30;


--Query 10
--List the animals have special needs and are older than 5 years.


SELECT name, TIMESTAMPDIFF(YEAR, birth_date, CURDATE()) AS Age,
special_needs
FROM animal
```

```
GROUP BY name, Age, special_needs
HAVING Age > 3 AND special_needs != NULL;


--Query 11
--What is the most common origin for persian cats?

SELECT name, origin, (SELECT COUNT(*)
       FROM animal
       WHERE animal.id_number = cat.id_number) AS commonOrigin
FROM animal JOIN cat
WHERE animal.id_number = cat.id_number
       AND breed = "persian";

--Query 12
--What breed of dog is most admitted?


SELECT MAX(breed) AS mostCommonType
FROM dog
GROUP BY breed;


--Query 13
--List any unsexed dogs that are currently at the shelter.


SELECT name, sexed
FROM animal JOIN dog
WHERE animal.id_number = dog.id_number
       AND animal.id_number NOT IN(SELECT id_number FROM adopted);

--Query 14
--List the animals that have been reserved for more than 5 days.


SELECT name, r_date, TIMESTAMPDIFF(DAY, r_date, CURDATE()) AS daysReserved
FROM animal JOIN reserved
WHERE animal.id_number = reserved.id_number
GROUP BY r_date, name
HAVING daysReserved > 5;
```

```
--Query 15
--List any customers with address, 608 Franklin St, or phone number, 555-
555-5555, that have a DNA(do not adopt) flag.


SELECT first_name, last_name, street_number, street_name, phone_number,
allowed_to_adopt
FROM person JOIN customer
WHERE person.human_id = customer.human_id
AND ((street_number = 608 AND street_name = 'Franklin St') OR phone_number
= 5555555555)
AND allowed_to_adopt = 0;


--Query 16
--List the medical staff that has treated cat Thomas O Maley.


SELECT first_name, last_name
FROM person JOIN treated
WHERE person.human_id = treated.human_id
       AND id_number = 4;

--Query 17
--List all medical staff that also own animals that were adopted or treated
by the shelter.


SELECT first_name, last_name, name
FROM person JOIN medical JOIN adopted JOIN treated JOIN animal
WHERE person.human_id = medical.human_id = adopted.human_id =
treated.human_id AND treated.id_number = animal.id_number


--Query 18
--List the number birman cats have been unsexed in the past 6 months?

SELECT COUNT(sexed) AS unsexed
FROM animal JOIN cat
WHERE animal.id_number = cat.id_number
       AND sexed = 0
       AND breed = "birman";
```

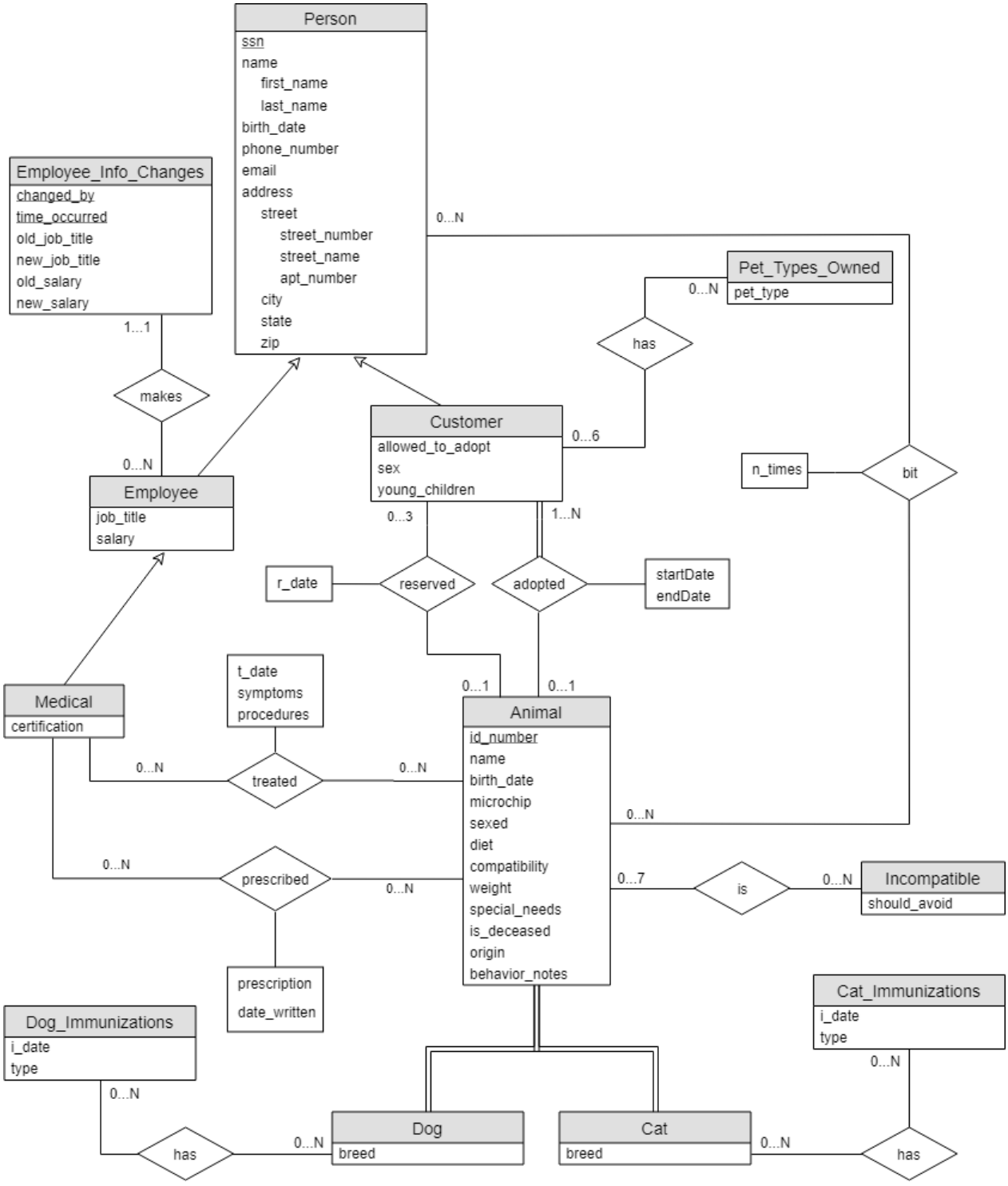
```sql
--Query 19
--How many animals have been returned to the shelter?


SELECT COUNT(end_date) AS numReturned
FROM adopted;


--Query 20
--List the employees that are not volunteers and make less than $8 per
hour.


SELECT first_name, last_name
FROM person JOIN employee
WHERE person.human_id = employee.human_id
       AND salary < 8
       AND job_title != "volunteer";
```

ER Diagram:


Person
ssn
name
first_name
last_name
birth_date
phone_number
email
address
street
street_number
street_name
apt_number
city
state
zip
Employee_Info_Changes
changed_by
time_occurred
old_job_title
new_job_title
old_salary
new_salary
Pet_Types_Owned
pet_type
has
makes
Customer
allowed_to_adopt
sex
young_children
n_times
bit
Employee
job_title
salary
r_date
reserved
adopted
startDate
endDate
t_date
symptoms
procedures
Medical
certification
Animal
id_number
name
birth_date
microchip
sexed
diet
compatibility
weight
special_needs
is_deceased
origin
behavior_notes
treated
prescribed
is
Incompatible
should_avoid
prescription
date_written
Dog_Immunizations
i_date
type
Cat_Immunizations
i_date
type
Dog
breed
Cat
breed
has
has
0...N
1...1
0...6
1...N
0...3
0...1
0...7

Relational Model:

Entity	Attribute	Domain
Person	human_id (primary key)	A bigint(20) in the range 0 - (2^{64}-1) Must be real.
	first_name	A variable length UTF-8 character string. Maximum length is 32 characters.
	last_name	A variable length UTF-8 character string. Maximum length is 32 characters.
	birth_date	MySQL DATE type.
	phone_number	10-digit positive numeric value for area code and number. Null allowed.
	email	A variable length UTF-8 string containing characters before and after an @ symbol. Maximum length is 64 characters. Null allowed.
	street_number	A positive mediumint(8) value in the range 0-16777215.
	street_name	A variable length UTF-8 character string. Maximum length is 16 letters.
	apt_number	A positive mediumint(8) value in the range 0-16777215. Null allowed.
	city	A variable length UTF-8 character string. Maximum length is 24 letters.
	state	2-character string for state code in upper case letters (A-Z).
	zip	A positive 5-digit numeric value for zip code.
Customer	allowed_to_adopt	A positive tinyint(1) value, where options are 0, or 1 representing whether the customer is allowed to adopt or barred for whatever reason.

	has_young_children	A positive tinyint(1) value, where options are 0, or 1 representing whether the customer's household contains any children between the ages of 0 and 10.
	sex	One of three options, "Female" , "Male" or "Please ask"
Pet_Types_Owned	pet_types_owned	Zero, one, or several of a set of pet types.
Employee	ssn	9-digit positive numeric value for a social security number.
	job_title	One of a set of possible job titles.
	salary	Numeric currency value with 2 decimal places and a range of 0.00-999999.99
Employee_Info_Changes	changed_by	Name of person who made the change. A variable length UTF-8 character string, numbers and emojis not allowed. Maximum length is 32 characters.
	time_occured	MySQL TIMESTAMP type.
	old_job_title	One of a set of job titles
	new_job_title	One of a set of job titles
	old_salary	Numeric currency value with 2 decimal places and a range of 0.00-999999.99
	new_salary	Numeric currency value with 2 decimal places and a range of 0.00-999999.99
Medical	certification	One of a set of possible values to represent the highest level of certification achieved by the medical staff member.
Animal	id_number	A positive mediumint(8) value in the range 0-16777215.

	name	A variable length UTF-8 character string that corresponds to the animal's name. Maximum length 32 characters.
	birth_date	MySQL DATE type.
	date_sheltered	MySQL DATE type.
	gender	One of three options, "Female" , "Male" or "Please ask"
	microchip_id	A 10-digit number representing the ID number of the microchip implanted in this animal, if such a microchip exists. If no such microchip exists, this value is null.
	sexed	A positive tinyint(1) value, where options are 0, or 1 to denote whether or not this animal has been spayed/neutered.
	diet	A variable length UTF-8 character string describing any special dietary restrictions the animal has. Maximum length 256 characters.
	weight	Numeric value in the range 0.0-999.9. To be measured in pounds.
	special_needs	A variable length UTF-8 character string describing any special/unusual accommodations the animal requires. Maximum length 256 characters. Null allowed.
	is_deceased	A positive tinyint(1) value, where options are 0, or 1 to denote whether this animal has passed away.
	origin	A variable length UTF-8 character string describing the location of where the animal was found. Maximum length 256 characters. Null allowed.
	behavior_notes	A variable length UTF-8 character

		string describing any behavioral quirks the animal has. Maximum length 256 characters. Null allowed.
Cat	breed	One or two of a set of possible breed names
Dog	breed	One or two of a set of possible breed names
Cat Immunizations	i_date	MySQL DATE type.
	type	Zero, one, or several from a set of possible immunizations. Not included as a “treatment” because an animal may be immunized prior to admission.
Dog Immunizations	i_date	MySQL DATE type.
	type	Zero, one, or several from a set of possible immunizations. Not included as a “treatment” because an animal may be immunized prior to admission.
Incompatible	should_avoid	Zero, one, or several from a set of possible compatible groups, e.g. small_children, cats, dogs.

Relationship	**Attribute**	**Domain**
bit	n_times	A tinyint(4) value ranging from 1-256, number of times an animal bit a specific person (whole numbers only)
reserved	r_date	MySQL DATE type.
adopted	start_date	MySQL DATE type.
	end_date	MySQL DATE type.
treated	t_date	MySQL DATE type.
	symptoms	A variable length UTF-8 character

		string describing any behavioral quirks the animal has. Maximum length 256 characters.
	procedures	A variable length UTF-8 character string describing/naming any surgeries or operations to be performed. Maximum length 256 characters.
prescribed	prescription	A variable length UTF-8 character string describing the medication, activity, or treatment prescribed. Maximum length 256 characters.
	date_written	MySQL DATE type.

Functionality of Relationship Diagram:

Relationship	**Functionality**	**Justification**
bit	many-to-many	One animal can bite many people, and one person can be bitten by many animals.
reserved	many-to-one	One person can reserve up to three animals, and one animal can be reserved by a person.
adopted	many-to-one	One person can adopt many animals, and each animal can only be adopted by one person at a time.
treated	many-to-many	A medical employee can treat many animals, and an animal can be treated by many medical employees.
prescribed	many-to-many	A medical employee may write many prescriptions for an animal, and an animal may be prescribed things by many medical employees.

Additional Integrity/Consistency Constraints:

The value of an attribute cannot be null unless otherwise specified.

All ranges listed are inclusive.

A person’s name, and address must be their legal name, and mailing address.

An employee’s ssn must be their legal social security number.

An attribute with a boolean domain is stored as an unsigned tinyint(1).

If r_date is older than 6 months, that reservation should be deleted.

All dates should be in the present or in the past.

Logical Design:

Person (human_id, first_name, last_name, birth_date, phone_number, email, street_number, street_name, apt_number, city, state, zip)

Candidate Keys: human_id

Primary Key: human_id

Functional Dependencies satisfied: human_id → first_name, last_name, birth_date, phone_number, email, street_number, street_name, apt_number, city, state, zip

Normal Form: 5NF

Customer (human_id, allowed_to_adopt, young_children, pet_types_owned)

Candidate Keys: human_id

Primary Key: human_id

Functional Dependencies satisfied: human_id → allowed_to_adopt, young_children, pet_types_owned

Normal Form: 5NF

Employee (human_id, job_title, salary)

Candidate Keys: human_id

Primary Key: human_id

Functional Dependencies satisfied: human_id → ssn, job_title, salary

Normal Form: 5NF

Medical (human_id, certification)

Candidate Keys: human_id

Primary Key: human_id

Functional Dependencies satisfied: human_id → certification

Normal Form: 5NF

Animal (id_number, name, birth_date, date_sheltered, gender, microchip, sexed, diet, weight, special_needs, is_deceased, origin, behavior_notes)

Candidate Keys: id_number

Primary Key: id_number

Functional Dependencies satisfied: id_number → name, birth_date, date_sheltered, gender, microchip, sexed, diet, weight, special_needs, is_deceased, origin, behavior_notes

Normal Form: 5NF

Cat (id_number, breed)

Candidate Keys: id_number

Primary Key: id_number

Functional Dependencies satisfied: id_number → breed

Normal Form: 5NF

Dog (id_number, breed)

Candidate Keys: id_number

Primary Key: id_number

Functional Dependencies satisfied: id_number → breed

Normal Form: 5NF

Pet_Types_Owned(human_id, pet_types_owned)

Candidate Keys: human_id

Primary Key: human_id

Functional Dependencies satisfied: human_id → pet_types_owned

Normal Form: 5NF

Employee_Info_Changes(changed_by, time_occurred, human_id, old_job_title, new_job_title, old_salary, new_salary)

Candidate Keys: changed_by, time_occurred

Primary Key: changed_by, time_occurred

Functional Dependencies satisfied: changed_by, time_occurred, → human_id, old_job_title, new_job_title, old_salary, new_salary

Normal Form: 5NF

Cat_Immunizations(id_number, i_date, type)

Candidate Keys: id_number

Primary Key: id_number

Functional Dependencies satisfied: id_number → i_date, type

Normal Form: 5NF

Dog_Immunizations(id_number, i_date, type)

Candidate Keys: id_number

Primary Key: id_number

Functional Dependencies satisfied: id_number →i_date, type

Normal Form: 5NF

Incompatible(id_number, should_avoid)

Candidate Keys: id_number

Primary Key: id_number

Functional Dependencies: id_number → should_avoid

Normal Form: 5NF

bit (id_number, human_id, n_times)

Candidate Keys: id_number, human_id

Primary Key: id_number, human_id

Functional Dependencies satisfied: id_number, human_id → n_times

Normal Form: 5NF

reserved (human_id, id_number, r_date)

Candidate Keys: human_id, id_number

Primary Key: human_id, id_number

Functional Dependencies satisfied: human_id, id_number → r_date

Normal Form: 5NF

adopts (human_id, id_number, startDate, endDate)

Candidate Keys: human_id, id_number

Primary Key: human_id, id_number

Functional Dependencies satisfied: human_id, id_number → startDate, endDate

Normal Form: 5NF

treated (human_id, id_number, t_date, symptoms, procedures)

Candidate Keys: human_id, id_number

Primary Key: human_id, id_number

Functional Dependencies satisfied: human_id, id_number → t_date, symptoms, procedures

Normal Form: 5NF

prescribed (human_id, id_number, prescription, date_written)

Candidate Keys: human_id, id_number

Primary Key: human_id, id_number

Functional Dependencies satisfied: human_id, id_number → prescription, date_written

Normal Form: 5NF

How we decomposed to BCNF/4NF

We started by converting each entity and each relation to a schema. At that point, all of the schemas already satisfied 1NF, 2NF, 3NF, BCNF, 4NF, and 5NF, so we didn't have to decompose anything. Part of the reason it was so easy is that we assumed that each person would only list one address, one phone number, and one email address. In other situations it might be important to collect multiple of each if people offer them, but we think it's sufficient in this case to collect only the primary contact information for each person.

We briefly thought that ZIP codes could always be determined by the address (and thus ZIP codes were functionally determined by addresses, so the schema was actually in 1NF unless it was decomposed) but it turns out that ZIP codes are a little more complicated than that, so the dependency doesn't hold true.