INFSCI 2710
Database Management

Final Project Report

By
Sai Supreetha Varanasi (sav52@pitt.edu)
Vaishnavi Deshpande (vad28@pitt.edu)

Fall 2017
School of Computing and Information
University of Pittsburgh

sav52@pitt.edu
vad28@pitt.edu

Table of Contents

Database Schema Design 3
1. **ER Diagram of snowflake schema of Animal Shelter database: 3**
2. **DDL scripts for the fact table and dimension tables: 3**
   a. Staging table: 3
   b. Dimension tables: 4
   c. Fact table: 5

Extraction Transformation and Loading 6
1. **Extracting from Staging table: 6**
   a. Finding the distinct values from staging table for filling dimension tables: 6
   b. Extraction for further normalizing the AgeuponOutcome field: 6
2. **Transforming table structure: 7**
3. **Loading into fact and dimension tables: 8**
   a. Inserting into dimension tables: 8
   b. Inserting into the main fact table: 9

Data Exploration 10
1. **Elementary insights from the table: 10**
   a. Number of animals with no name: 10
   b. Number of animals by age measure: 11
2. **Advanced insights from the dataset: 12**
   a. OutcomeType grouped by AnimalType: 12
   b. OutcomeType grouped by Name attribute and AnimalType: 13
   c. OutcomeType grouped by Breed attribute and AnimalType = Cat: 15
   d. OutcomeType grouped by Breed attribute and AnimalType = Dog: 17

Conclusion 18
1. **Cat vs Dogs Outcome comparison 18**
2. **Animal's name and Outcome analysis 19**
3. **Outcome of different Cat breeds 19**
4. **Outcome of different Dog breeds 19**

References: 19

sav52@pitt.edu
vad28@pitt.edu

Database Schema Design

1. ER Diagram of snowflake schema of Animal Shelter database:

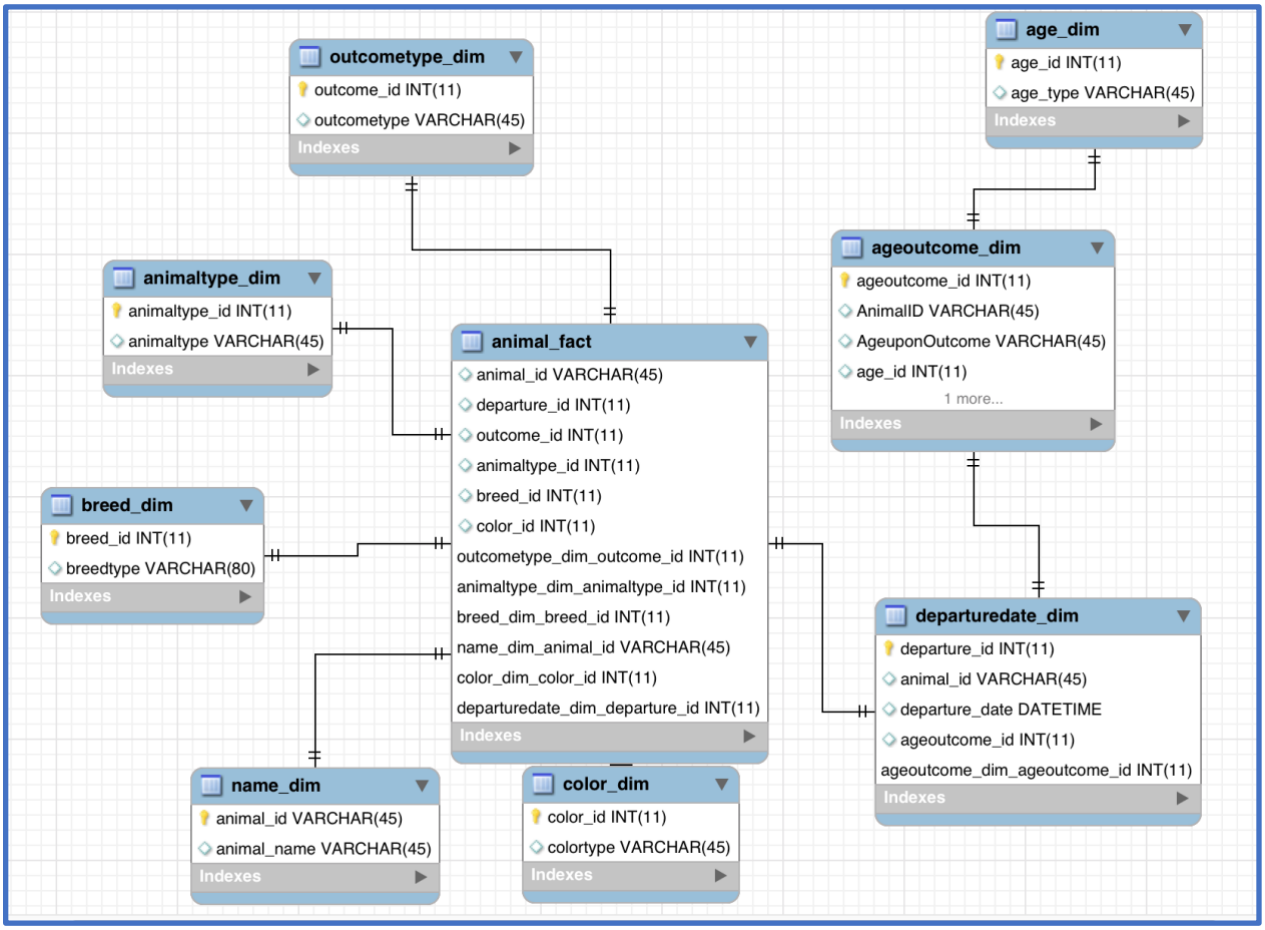


Fig 1: ER Diagram of snowflake schema

2. DDL scripts for the fact table and dimension tables:

```
CREATE SCHEMA `animal` ;
```

a. Staging table:

```
CREATE TABLE `animal`.`animal_staging`(
```

```
`AnimalID` VARCHAR(45) NULL,
`Name` VARCHAR(45) NULL,
`DepartureDateTime` VARCHAR(45) NULL,
`OutcomeType` VARCHAR(45) NULL,
`AnimalType` VARCHAR(45) NULL,
`AgeUponOutcome` VARCHAR(45) NULL,
`Breed` VARCHAR(45) NULL,
`Color` VARCHAR(45) NULL);
```

b. Dimension tables:

```
CREATE TABLE `animal`.`outcometype_dim` (
  `outcome_id` INT NOT NULL AUTO_INCREMENT,
  `outcometype` VARCHAR(45) NULL,
  PRIMARY KEY (`outcome_id`));

CREATE TABLE `animal`.`animaltype_dim` (
  `animaltype_id` INT NOT NULL AUTO_INCREMENT,
  `animaltype` VARCHAR(45) NULL,
  PRIMARY KEY (`animaltype_id`));

CREATE TABLE `animal`.`breed_dim` (
  `breed_id` INT NOT NULL AUTO_INCREMENT,
  `breedtype` VARCHAR(80) NULL,
  PRIMARY KEY (`breed_id`));


CREATE TABLE `animal`.`color_dim` (
  `color_id` INT NOT NULL AUTO_INCREMENT,
  `colortype` VARCHAR(45) NULL,
  PRIMARY KEY (`color_id`));
```

```
CREATE TABLE `animal`.`departuredate_dim`(
  `animal_id` VARCHAR(45) NULL,
  `departure_date` DATETIME NULL);

CREATE TABLE `animal`.`name_dim` (
  `animal_id` VARCHAR(45) NULL,
  `animal_name` VARCHAR(45) NULL);

CREATE TABLE `animal`.`age_dim` (
  `age_id` INT NOT NULL AUTO_INCREMENT,
  `age_type` VARCHAR(45) NULL,
  PRIMARY KEY (`age_id`));


CREATE TABLE `animal`.`ageoutcome_dim` (
  `ageoutcome_id` INT NOT NULL AUTO_INCREMENT,
  `AnimalID` VARCHAR(45) NULL,
  `AgeuponOutcome` VARCHAR(45) NULL,
  `age_id` INT NULL,
  PRIMARY KEY (`ageoutcome_id`));
```

c. Fact table:

```
CREATE TABLE `animal`.`animal_fact` (
  `animal_id` VARCHAR(45) NULL,
  `departure_id` INT NULL,
  `outcome_id` INT NULL,
  `animaltype_id` INT NULL,
  `breed_id` INT NULL,
  `color_id` INT NULL);
```

sav52@pitt.edu
vad28@pitt.edu

Extraction Transformation and Loading

- We have to forst convert the mail dataset file, the csv file in the form a new csv file with utf-8 encoding. Otherwise, many of the data is not imported.
- Then using the MySQL workbench Table import wizard, load the data into the above created animal_staging table.
- Once the data is successfully imported, we can explore the data further (like shown below in the next section) to decide how to load data into remaining tables.
- Loading process is made easier by the fact that the result of these explorations are basically what is inserted into the other tables.

1. Extracting from Staging table:

We have created the dimension tables in the previous section, we now need to extract data from the staging table to populate these dimension tables.

a. Finding the distinct values from staging table for filling dimension tables:

```
select count(distinct OutcomeType)from animal_staging;
```

- ANS: 5

```
select count(distinct AnimalType)from animal_staging;
```

- ANS: 2

```
select count(distinct Breed)from animal_staging;
```

- ANS: 1380

```
select count(distinct Color)from animal_staging;
```

- ANS: 366

b. Extraction for further normalizing the AgeuponOutcome field:

```
select count(*) from animal_staging
```

```
where AgeuponOutcome like "%year%";
```

- ANS: 14843

```
select count(*) from animal_staging
where AgeuponOutcome like "%month%";
```

- ANS: 9620

```
select count(*) from animal_staging
where AgeuponOutcome like "%week%";
```

- ANS: 1850

```
select count(*) from animal_staging
where AgeuponOutcome like "%day%";
```

- ANS: 398

```
select count(*) from animal_staging
where AgeuponOutcome = "";
```

- ANS: 18

2. Transforming table structure:

It is often the case that initial table created will need to be altered as we explore the dataset more and usually before any data is loaded.

```
ALTER TABLE `animal`.`departuredate_dim`
ADD COLUMN `dep_age_weeks` INT NULL AFTER `departure_date`;

ALTER TABLE `animal`.`departuredate_dim`
```

```
CHANGE COLUMN `dep_age_weeks` `age_id` INT(11) NULL DEFAULT
NULL ,
ADD COLUMN `departure_id` INT NOT NULL AUTO_INCREMENT
FIRST,
ADD PRIMARY KEY (`departure_id`);

ALTER TABLE `animal`.`departuredate_dim`
CHANGE COLUMN `age_id` `ageoutcome_id` INT(11) NULL DEFAULT
NULL ;
```

Notice that we are transforming the the DepartureDateTime from string format to Date format using special format specifiers before loading them into the dimension table.

```
insert into departuredate_dim
select AnimalID, STR_TO_DATE(DepartureDateTime,'%c/%e/%y %k:%i')
from animal_staging;
```

3. Loading into fact and dimension tables:

a. Inserting into dimension tables:

We load the data that we obtained from Step 1 in this section into its respective dimension tables using the insert command. Notice that we have used the Auto Increment feature for creating the primary keys for these dimensions.

```
insert into outcometype_dim (outcometype)
select distinct OutcomeType from animal_staging;

insert into animaltype_dim (animaltype)
select distinct AnimalType from animal_staging;

insert into color_dim (colortype)
```

```
select distinct Color from animal_staging;


insert into breed_dim (breedtype)
select distinct Breed from animal_staging;
```

The name dimension holds the actual AnimalIDs and names, if present. Notice that there is no separate primary key being created as AnimalID itself acts as the primary key.

```
insert into name_dim
select AnimalID, Name from animal_staging ;
```

The age dimension is created to further normalize the departuredate dimension table. It takes the as the primary key.

```
insert into age_dim(age_type) values('year'),('month'),('week'),('day'),('');


insert into ageoutcome_dim(AnimalID, AgeuponOutcome)
select AnimalID,AgeuponOutcome from animal_staging;
```

b. Inserting into the main fact table:

The animal fact table will refer to all the primary keys of the dimension tables while preserving the original staging table structure.

```
insert into animal_fact(animal_id,departure_id)
select animal_id,departure_id from departuredate_dim;


insert into animal_fact(outcome_id)
```

```
select outcome_id from animal_staging, outcometype_dim
where animal_staging.OutcomeType = outcometype_dim.outcometype;

insert into animal_fact(animaltype_id)
select animaltype_id from animal_staging, animaltype_dim
where animal_staging.AnimalType = animaltype_dim.animaltype;

insert into animal_fact(breed_id)
select breed_id from animal_staging, breed_dim
where animal_staging.Breed = breed_dim.breedtype;

insert into animal_fact(color_id)
select color_id from animal_staging, color_dim
where animal_staging.Color = color_dim.colortype;
```

Data Exploration

1. Elementary insights from the table:

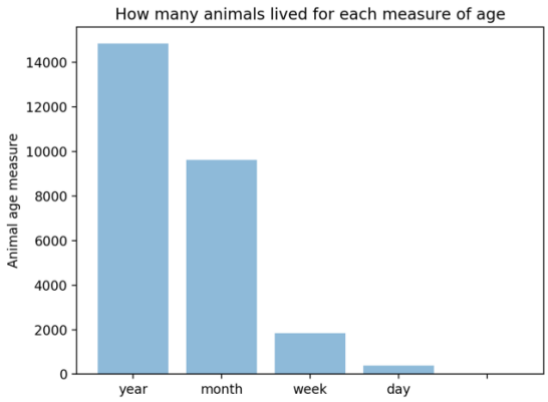
a. Number of animals with no name:

```
select count(*) from animal_staging
where Name = "";
```

- ANS: 7691

sav52@pitt.edu
vad28@pitt.edu

b. Number of animals by age measure:




```
import csv
import mysql.connector
import matplotlib.pyplot as plt
plt.rcdefaults()
import numpy as np
mydb = mysql.connector.connect(host="localhost",
                               user="root",
                               password="INFSCI2710",
                               database="animal")
rowcount=0
print('connected to data base')
cursor = mydb.cursor()

cursor.execute('''Select * from age_dim;''')

age_types = cursor.fetchall()
print(age_types)
agetype_dict = [{i[1]:i[0]} for i in age_types]
print(agetype_dict)

agetype_list = [list(i.keys())[0] for i in agetype_dict]

y_pos = np.arange(len(agetype_list))
# values gotten from mysql workbench queries
performance = [14843,9620,1850,398,18]

plt.bar(y_pos, performance, align='center',alpha=0.5)
plt.xticks(y_pos,agetype_list)
plt.ylabel('Animal age measure')
plt.title('How many animals lived for each measure of age')
plt.show()

#commit all insert transactions here
mydb.commit()
#close the connection to the database.
cursor.close()
print("Done")
```

sav52@pitt.edu
vad28@pitt.edu

2. Advanced insights from the dataset:

Analysis using R language is done as creating visualization plots is easier for us in R than in python. The following libraries are used to plot the charts:

```
library(ggplot2)
library(dplyr)
library(tidyr)
library(plotly)
library(scales)
```

a. OutcomeType grouped by AnimalType:

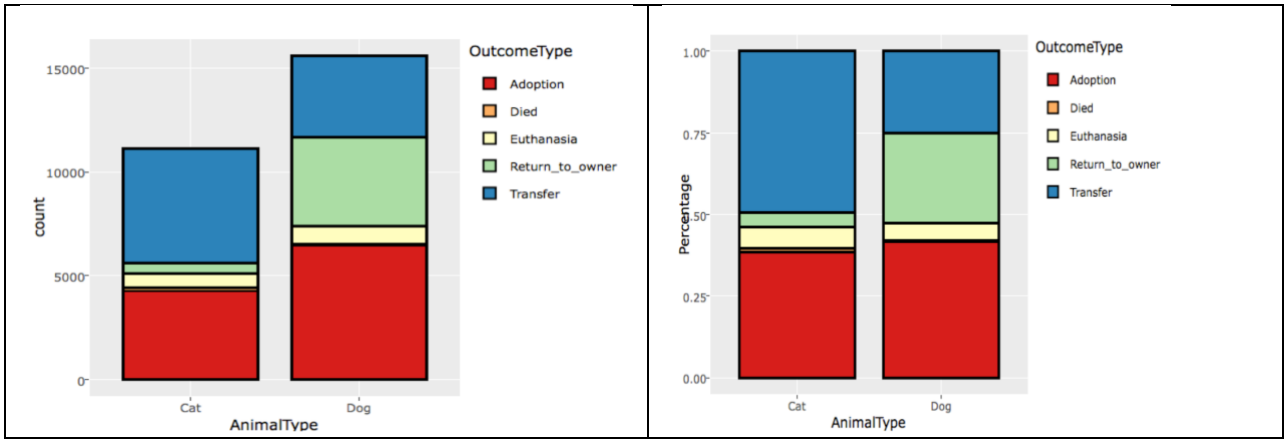
```
create view dep_outcome as
select OutcomeType, count(AnimalType), AnimalType
from animal_staging
group by AnimalType, OutcomeType;

select * from dep_outcome;
```

OutcomeType	count(AnimalType)	AnimalType
Adoption	4272	Cat
Died	147	Cat
Euthanasia	710	Cat
Return_to_owner	500	Cat
Transfer	5505	Cat
Adoption	6497	Dog
Died	50	Dog
Euthanasia	845	Dog
Return_to_owner	4286	Dog
Transfer	3917	Dog

This below plot is obtained by using the R code snippet below to get comparison of outcomes in terms of percentages and to visualize the results:

```
bar_plot <- function(data, xvar, group,
                     position = "stack",
                     color = "black",
                     palette = "Spectral") {
  g <- ggplot(data, aes_string(xvar, fill = group)) +
```

```
    geom_bar(position = position, color = color) +
    scale_fill_brewer(palette = palette)
  if(position == "fill") {
    g <- g + scale_y_continuous(labels=percent) +
ylab("Percentage")
  }
  return(g)
}
outcomeCount <- animal_train %>%
  group_by(OutcomeType, OutcomeSubtype) %>% count()

Sankey <- gvisSankey(outcomeCount, from="OutcomeType",
to="OutcomeSubtype", weight="n")
plot(Sankey)
```



b. OutcomeType grouped by Name attribute and AnimalType:

Cats' outcomes when they don't have names:

```
select OutcomeType, count(OutcomeType)
from animal_staging
where AnimalType="cat" and Name = ""
group by OutcomeType;
```

OutcomeType	*count(OutcomeType)*
Adoption	646
Died	92
Euthanasia	554
Return_to_owner	44

Transfer	3683

Cats' outcomes when they have names:

```
select OutcomeType, count(OutcomeType)
from animal_staging
where AnimalType="cat" and Name != ""
group by OutcomeType;
```

OutcomeType	*count(OutcomeType)*
Adoption	3626
Died	55
Euthanasia	156
Return_to_owner	456
Transfer	1822

Dogs' outcomes when they don't have names:

```
select OutcomeType, count(OutcomeType)
from animal_staging
where AnimalType="dog" and Name = ""
group by OutcomeType;
```

OutcomeType	*count(OutcomeType)*
Adoption	1032
Died	28
Euthanasia	261
Return_to_owner	109
Transfer	1242

Dogs' outcomes when they have names:

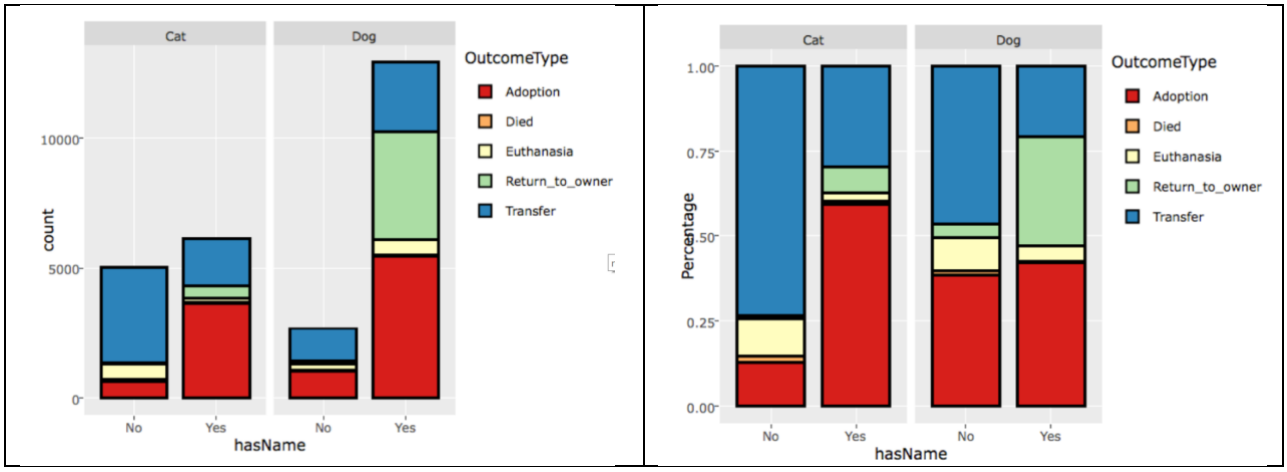
```
select OutcomeType, count(OutcomeType)
from animal_staging
where AnimalType="dog" and Name != ""
group by OutcomeType;
```

sav52@pitt.edu
vad28@pitt.edu

OutcomeType	*count(OutcomeType)*
Adoption	5465
Died	22
Euthanasia	584
Return_to_owner	4177
Transfer	2675

This below plot is obtained by using the R code snippet below to get comparison of outcomes in terms of percentages and to visualize the results:

```
animal_train <- animal_train %>%
  mutate(hasName = ifelse(is.na(Name), "No", "Yes"))

ggplotly(bar_plot(animal_train, "hasName", "OutcomeType") +
           facet_grid(. ~ AnimalType))

ggplotly(bar_plot(animal_train, "hasName", "OutcomeType",
position = "fill") +
           facet_grid(. ~ AnimalType))
```



c. OutcomeType grouped by Breed attribute and AnimalType = Cat:

```
create view cat_breed as
select OutcomeType, count(Breed) as breed_count, Breed
from animal_staging
```

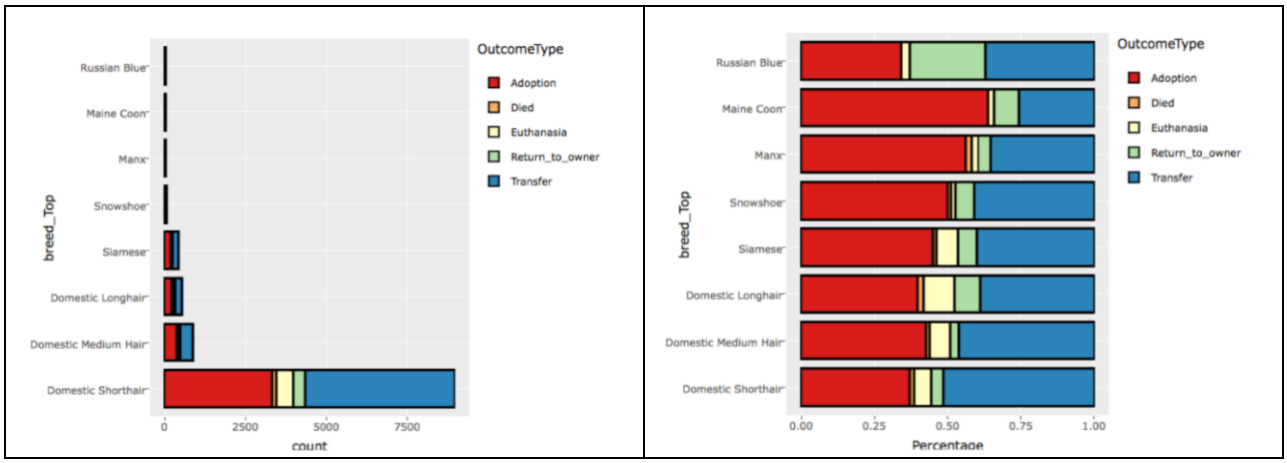
```
where AnimalTYpe="cat"
group by Breed, OutcomeType;


select * from cat_breed order by breed_count desc limit 8;
```

OutcomeType	breed_count	Breed
Transfer	4538	Domestic Shorthair Mix
Adoption	3273	Domestic Shorthair Mix
Euthanasia	535	Domestic Shorthair Mix
Transfer	383	Domestic Medium Hair Mix
Adoption	357	Domestic Medium Hair Mix
Return_to_owner	352	Domestic Shorthair Mix
Transfer	205	Domestic Longhair Mix
Adoption	199	Domestic Longhair Mix

This below plot is obtained by using the R code snippet below to get comparison of outcomes in terms of percentages and to visualize the results:

```
animal_train <- animal_train %>%
  mutate(breed_Top = gsub("/.*", "", gsub(" Mix", "", Breed)))

animalTopBreed <- animal_train %>%
  group_by(AnimalType, breed_Top) %>%
  summarise(Total = n()) %>%
  top_n(8, Total)

breedByFreq <-
animalTopBreed$breed_Top[order(animalTopBreed$Total, decreasing =
TRUE)]

ggplotly(bar_plot(animal_train %>%
                    filter(AnimalType == "Cat", breed_Top %in%
breedByFreq) %>%
                    mutate(breed_Top = factor(breed_Top, levels =
breedByFreq)),
                  "breed_Top", "OutcomeType") + coord_flip())

ggplotly(bar_plot(animal_train %>%
```

```
                    filter(AnimalType == "Cat", breed_Top %in%
breedByFreq) %>%
                    mutate(breed_Top = factor(breed_Top, levels =
breedByFreq)),
                  "breed_Top", "OutcomeType", position = "fill")
+ coord_flip())
```



d. OutcomeType grouped by Breed attribute and AnimalType = Dog:

```
create view dog_breed as
select OutcomeType, count(Breed) as breed_count, Breed
from animal_staging
where AnimalType = "dog"
group by Breed, OutcomeType;
```

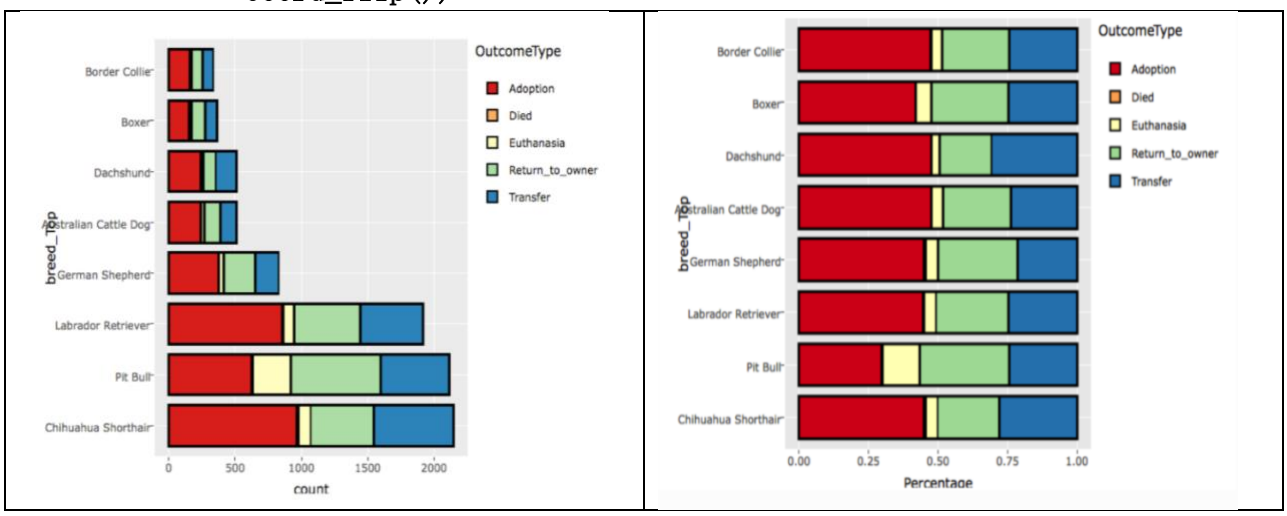
```
select * from dog_breed order by breed_count desc limit 8;
```

OutcomeType	breed_count	Breed
Adoption	767	Chihuahua Shorthair Mix
Return_to_owner	598	Pit Bull Mix
Adoption	590	Labrador Retriever Mix
Adoption	568	Pit Bull Mix
Transfer	501	Chihuahua Shorthair Mix
Transfer	478	Pit Bull Mix
Return_to_owner	407	Chihuahua Shorthair Mix

Return_to_owner	369	Labrador Retriever Mix

This below plot is obtained by using the R code snippet below to get comparison of outcomes in terms of percentages and to visualize the results:

```
ggplotly(bar_plot(animal_train %>%
                    filter(AnimalType == "Dog", breed_Top %in%
breedByFreq) %>%
                    mutate(breed_Top = factor(breed_Top, levels =
breedByFreq)),
                  "breed_Top", "OutcomeType") + coord_flip())

ggplotly(bar_plot(animal_train %>%
                    filter(AnimalType == "Dog", breed_Top %in%
breedByFreq) %>%
                    mutate(breed_Top = factor(breed_Top, levels =
breedByFreq)),
                  "breed_Top", "OutcomeType", position = "fill")
+ coord_flip())
```



Conclusion

We have limited out analysis to just the study of the OutcomeType and how it is impacted by attributes like Name, Breed and AnimalType. Below are the conclusions from each of the plots obtained above.

1. Cat vs Dogs Outcome comparison

- Both cats and dogs were adopted equally.

- Dogs are much more likely to be returned to their owners than cats.
- Cats are transferred between shelters more often than dogs.
- Very few animals died or got euthanized overall but in that cats euthanized more.

2. Animal's name and Outcome analysis

- There are 7691 animals with no names.
- Cats with names were adopted more
- Adoption for dogs don't seem to depend on whether they have name or not.

3. Outcome of different Cat breeds

- Domestic shorthair cats are more in number in the animal shelter.
- Marine Coon is adopted more of all the breeds.
- Domestic Longhair euthanized more.

4. Outcome of different Dog breeds

- Chihuahua shorthair, Pitbull and Labrador Retriever dogs are more in number.
- There is no breed preference for adoption of dogs as all barely have same adoption %.
- Pitbull seems to be euthanized more, adopted less and returned to owner more.

References:

- https://www.kaggle.com/apapiu/visualizing-breeds-and-ages-by-outcome
- https://github.com/iamchuan/ShelterAnimals/blob/master/ShelterAnimals.R
- https://www.kaggle.com/andraszsom/age-gender-breed-and-name-vs-outcome
- https://mark-borg.github.io/blog/2016/shelter-animal-competition/