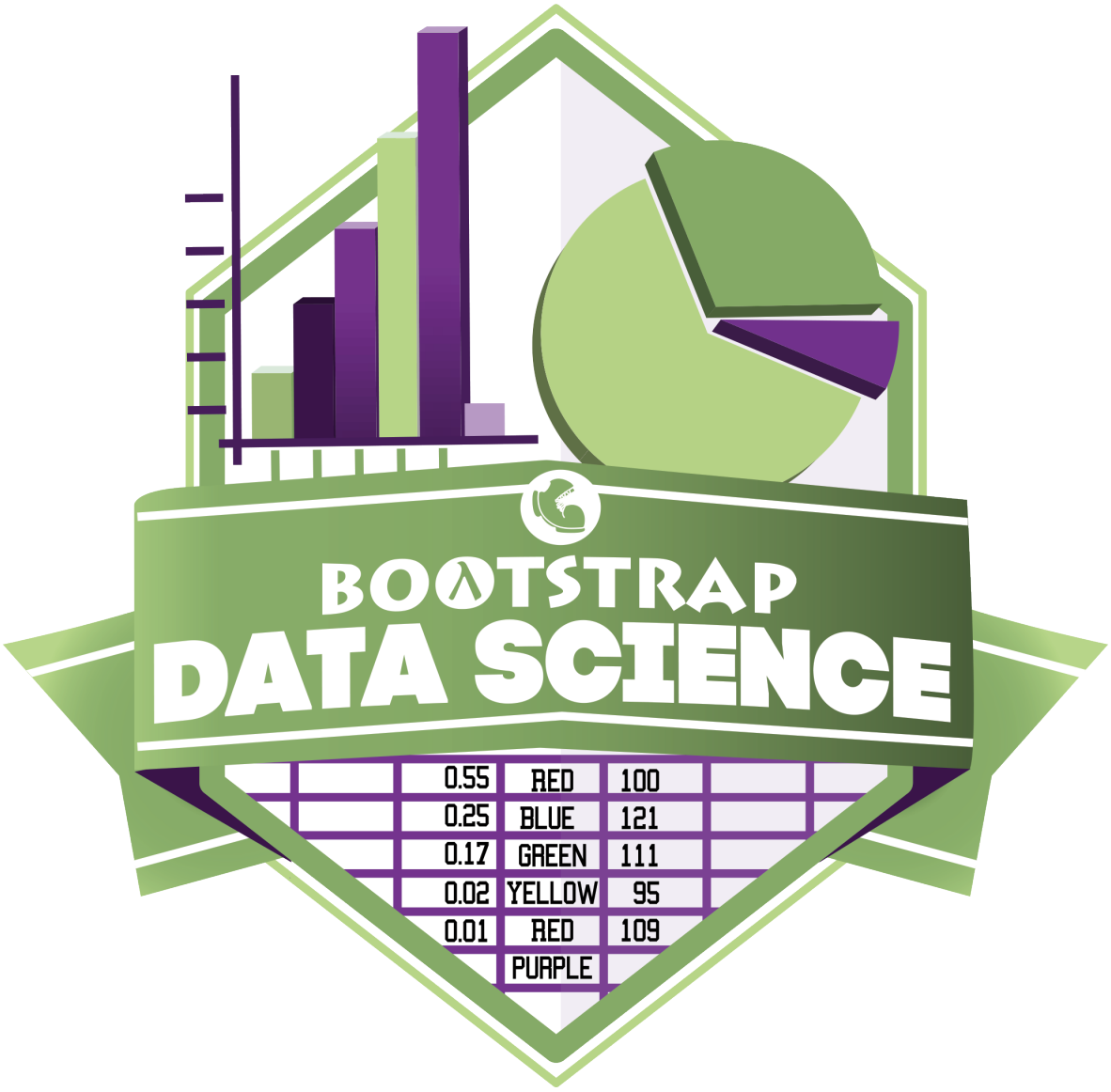


BOOTSTRAP
www.bootstrapworld.org

Workbook v1.1

Brought to you by the Bootstrap team:

- Emmanuel Schanzer
- Kathi Fisler
- Shriram Krishnamurthi
- Ed Campos
- Emma Youndtsmith
- Sam Dooman

Unit 1

- Many important questions ("what's the best restaurant in town?", "is this law good for citizens?", etc.) are answered with data. Data Scientists try and answer these questions, by writing *programs that ask questions of data.*

- Data of all types can be organized into **Tables**

- Every Table has a **header row**, and some number of **data rows**

- **Quantitative data** is data - usually numeric - that measures *quantity*, such as a person's height, a score on test, a measure of distance, etc. A list of quantitative data can be ordered from smallest to largest.

- **Categorical data** is data that specifies *categories*, such as eye color, country of origin, etc. A list of categorical data has no notion of "smallest" or "largest", and cannot be ordered.

- **Programming languages** involves different *datatypes*, such as Numbers, Strings, Booleans and Images.

- **Operators** (like +, -, *, <, etc.) are written between values. For example: `4 + 2`

- **Functions** (like triangle, star, string-repeat, etc.) are written first, followed by a list of **arguments** in parentheses. For example: `star(50, "solid", "red")`

- **Examples** help programmers reason about their code. Every example contains two expressions, and the example "passes" if both expressions evaluate to the same thing. For example: `4 + 2 is 6`, or `"cat" == "dog" is false`

Numbers and Strings

Make sure you've loaded the Unit 1 Starter File, and clicked "Run".

1. Try typing `42` into the Interactions Area and hitting "Enter". What happens?

2. Try typing in other Numbers. What happens if you try a decimal like `0.5`? A fraction like `1/3`? Try really big Numbers, and really small ones.

3. String values are always in quotes. Try typing your name (in quotes!). What happens when you hit "Enter"?

4. Try typing your name with the opening quote, but *without* the closing quote. What happens? Now try typing it without *any* quotes.

5. Is `42` the same as `"42"`? Why or why not? Write your answer below:

They are different data types: 42 (without quotes) is a Number, and "42" (with quotes) is a string.

Operators

6. Just like in math, Pyret has *operators* like `+` and `-`. Try typing in `4 + 2`, and then `4+2` (without the spaces). What can you conclude from this? Write your answer below:

Operators (like +) need whitespace separating them from their operands.

7. Try typing in `4 + 2 + 6`, `4 + 2 * 6`, and `4 + (2 * 6)`. What can you conclude from this? Write your answer below:

You can use the same operator multiple times without parentheses, but you need parentheses to group order of operations if using different operators (like + and *) together.

8. Try typing in `4 + "cat"`, and then `"dog" + "cat"`. What can you conclude from this? Write your answer below:

The + operator can only be used with Numbers, not Strings.

Booleans

Boolean expressions are yes-or-no questions, and will always evaluate to either `true` ("yes") or `false` ("no"). What will each of the expressions below evaluate to? Write down the result in the blanks provided, and type them into Pyret if you're not sure.

`3 <= 4`	True	`"a" > "b"`	False
`3 == 2`	False	`"a" <> "b"`	True
`2 <> 4`	True	`"a" == "b"`	False
`3 <> 3`	True	`"a" <> "a"`	False

Boolean Operators

Pyret also has operators that work on *Booleans*. For each expression below, *write down your guess* about what it will evaluate to. Then type them in and see if you were right!

`(3 <= 4) and (3 == 2)` False

`("a" == "b") and (3 <> 4)` False

`(3 <= 4) or (3 == 2)` True

`("a" == "b") or (3 <> 4)` True

1. How many different Number values are there in Pyret? Infinite
2. How many different String values are there in Pyret? Infinite
3. How many different Boolean values are there in Pyret? Two

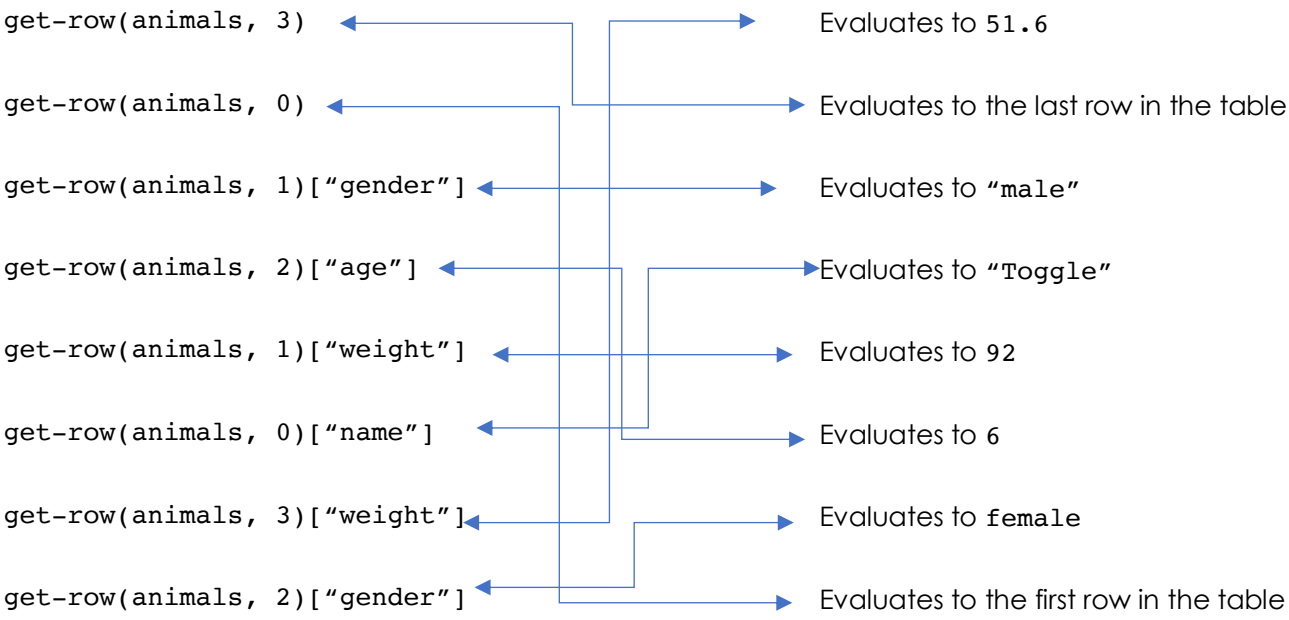
Playing with Tables

The table below represents four animals at the shelter:

animals

name	gender	age	weight
"Toggle"	"female"	3	48
"Fritz"	"male"	4	92
"Nori"	"female"	6	35.3
"Maple"	"female"	3	51.6

1) Match each Pyret expression (left) to the description of what it does (right).



2) Fill in the blanks (left) with the Pyret code that will produce the value (right).

a. *get-row(animals, 3)["name"]*	"Maple"
b. get-row(animals, 1)["gender"]	male
c. get-row(animals, 1)["age"]	4
d. get-row(animals, 0)["weight"]	48
e. get-row(animals, 2)["name"]	"Nori"

Writing Examples

1. In the examples block below, **put an "X" next to the examples that will *fail*.** Remember: examples only pass if the left- and right-hand expressions evaluate to the same thing!

```
examples:
  1 + 2 + 9                      is 19      ✖
  num-sqrt(16)                   is 2 + 2
  3 > 99                         is true    ✖
  square(10, "solid", "red")     is rectangle(10, 10, "solid", "red")
end
```

2. In the examples block below, **fill in the blank on the right-hand side so the example will *pass*.**

```
examples:
  string-repeat("yeah! ", 3)      is "yeah! yeah! yeah! "
  string-contains("Maya", "May")  is true
  "apples" <> "oranges"           is true
end
```

3. The examples block below refers to the `shapes` table on the right, using row-accessors and the `get-row` function. For each example, **fill in the blank so the example will *pass*.**

name	corners	is-round
"triangle"	3	false
"circle"	0	true
"ellipse"	0	true
"square"	4	false

```
examples:
  get-row(shapes, 3)["name"]      is "square"
  get-row(shapes, 0)["corners"]   is 3
  get-row(shapes, 2)["is-round"]  is true
end
```

Unit 2

Answering Questions from Data can take many forms. Here are a few types of questions, each requiring a different kind of analysis:

- **Lookup Questions** can be answered just by finding the right row and column a table. (e.g. – "How old is Toggle?")

- **Comparison Questions** can be answered by comparing a single row or column to all the rest of the table. (e.g. – "What is the heaviest animal at the shelter?")

- **Pattern Questions** require looking for trends across multiple rows or columns. (e.g. – "Do cats tend to be adopted sooner than dogs?")

Threats to Validity can undermine a conclusion, even if the analysis was done correctly. Some examples of threats are:

- **Sample size** – averaging the age of only three animals won't tell us anything reliable about the age of animals at the shelter!

- **Selection bias** – identifying the favorite food of the rabbits won't tell us anything reliable about what all the animals eat.

- **Sample error** – surveying dogs when they are puppies won't tell us anything reliable about overall dog behavior, since their behavior changes as they age.

- **Confounding variables** – if they person surveying the animals has a piece of bacon in their pocket, they will incorrectly find that all dogs are friendly!

The Animals Dataset

1. This dataset is Animals from an animal shelter

2. Four of my columns are….
   (choose four columns, and for each one fill out the name, datatype, and whether it contains Qualitative or Categorical data in the table below)

Name	species	age	fixed	legs
Datatype	String	Number	Boolean	Number
Quantitative or Categorical?	Categorical	Quantitative	Categorical	Quantitative

3. Three questions I have about my dataset:

   1. What is the average age of the animals at this shelter?

   2. Do dogs or cats tend to get adopted faster?

   3. Is there a connection between the age of an animal and how quickly it gets adopted?

The Design Recipe

Define a function called `is-fixed`, which tells us whether or not an animal is fixed

is-fixed :: *(animal :: Row)* → *Boolean*
name domain range

 # Consumes an animal, and produces the value in the fixed column

examples:

is-fixed (*sasha*) **is** *true*

is-fixed (felix) **is** felix["fixed"]

end

fun is-fixed (animal) **:** animal["fixed"]

end

Define a function called `gender`, which consumes a Row of the animals table tells us the gender of that animal

gender :: (animal :: Row) → String
name domain range

Consumes an animal, and produces the value in the gender column

examples:

gender (snowcone) **is** snowcone["gender"]

gender (toggle) **is** toggle["gender"]

end

fun gender (animal) **:** animal["gender"]

end

Define a function called `is-cat`, which consumes a Row of the animals table and produces `true` if it's a cat.

is-cat	::	*(animal :: Row)*	→	*Boolean*
name		domain		range

Consumes an animal, and return true if the species is "cat"

examples:

is-cat (sasha) **is** sasha["species"] == "cat"

is-cat (snuggles) **is** snuggles["species"] == "cat"

end

fun is-cat (animal) **:** animal["species"] == "cat"

end

Define a function called `is-young`, which consumes a Row of the animals table and produces `true` if it's an animal that is less than two years old.

is-young	::	(animal :: Row)	→	Boolean
name		domain		range

Consumes an animal, returns true if the animal is less than 2 years old

examples:

is-young (wade) **is** wade["age"] < 2

is-young (sheba) **is** sheba["age"] < 2

end

fun is-young (animal) **:** animal["age"] < 2

end

Define a function called `nametag`, prints out each animal's name in big red letters.

```
nametag :: (animal :: Row) → Image
name       domain            range
# Consumes an animal, and produces an image of their name in big, red letters
examples:
    nametag ( sasha ) is text(sasha["name"], 50, "red")
    nametag ( felix ) is text(felix["name"], 50, "red")
end
fun nametag ( animal ) : text(animal["name"], 50, "red")
end
```

Define a function called `is-kitten`, which consumes a Row of the animals table and produces true if it's a cat younger than two years old.

```
is-kitten :: (animal :: Row) → Boolean
name         domain             range
# Consumes an animal, returns true if it's a cat less than two years old
examples:
    is-kitten ( snuggles ) is (snuggles["species"] == "cat") and
                              (snuggles["age"] < 2)
    is-kitten ( wade ) is (wade["species"] == "cat") and
                          (wade["age"] < 2)
end
fun is-kitten ( animal ) : (animal["species"] == "cat") and
                           (animal["age"] < 2)
end
```

My Dataset

1. My dataset is [specific to each student]

2. Four of my columns are….
   (choose four columns, and for each one fill out the name, datatype, and whether it contains Qualitative or Categorical data in the table below)

Name				
Datatype				
Quantitative or Categorical?				

3. Three questions I have about my dataset:

   a.

   b.

   c.

Unit 3

- **Methods** are special functions that are attached to pieces of data. We use them to manipulate Tables.

- They are different from functions in several ways:
  1. Their names can't be used alone: they can only be used as part of data, separated by a dot. (For example, `animals.order-by)`
  2. Their contracts are different: they include the type of the data as part of their names. (eg, `<table>.order-by :: (column :: String) → Table`)
  3. They have a "secret" argument, which is the data they are attached to

- We will use three **Table Methods** to manipulate our datasets:
  1. <Table>.order-by – order the rows of a table based on a column
  2. <Table>.filter – create a **subset** of the data, with only certain rows
  3. <Table>.build-column – use the columns of a table to compute a new one

Reviewing Functions

1. **One of the examples for the last function is broken!** Fix this example in the Definitions Area.
2. How many *values* are defined in this file? 4
3. How many *functions* are defined in this file? 7
4. What is the *name* of the last function? is-old-dog
5. What is the *Domain* of the last function? Row
6. What is the *Range* of the last function? Boolean
7. What is the variable name that the last function uses? animal
8. Which function will tell us if an animal is a kitten? is-kitten
9. Which function will print out "<name> the <species>"? sentence
10. Which function will tell us if an animal is a dog older than 10? is-old-dog
11. Which function will tell us if an animal has been fixed? is-fixed
12. Which function will draw a nametag for an animal? nametag

Plans for the `Animals` Dataset

What are two ways you might want to *order* the `animals` dataset?

1) Order by weight

2) Order by age

What are two subsets into which you might *filter the `animals` dataset*?

1) Filter animals heavier than 20 pounds

2) Filter animals that have been fixed

What are two new columns you might want to *build* from the `animals` dataset?

1) Add a column for time in the shelter by months

2) Add a column for whether or not each animal is a kitten

Methods

Methods are a lot like functions, but they differ in three important ways:

- They can only be called as **part of a value**, using the **dot-accessor**. For example: **`animals.`**`row-n(2)`
- Their Contracts are different, because they contain a **Type** as part of their name. For example: **`<Table>.`**`row-n :: (index :: Number) -> Row`
- They have a "secret argument", which is the value they are attached to. In the examples above, the `row-n` method consumes only a `Number` as part of its Domain, but it *also* consumes the `Table` to which it is attached.

Here is the Contract for a method, which consumes the name of a food and produces True if the person likes that food:

`<Person>``.likes :: (`*`food :: String`*`) → Boolean`

1. What Type of data is the method *attached to?* Table
2. What is the name of this method? likes
3. How many things are in its Domain? 1
4. What is the name of the argument in its Domain? food
5. What is the Type of the argument in its Domain? String
6. What Type of data will this method will produce? Boolean
7. Below are 3 expressions. Based on the contract above, circle the correct one.

   `emma.likes("pizza")` `likes("pizza")` `likes(emma, pizza)`

8. On the line below, write your own expression that uses this method, replacing `emma` and `"pizza"` with your own name and a food *you* like.

amy.likes("chocolate")

Playing with Methods

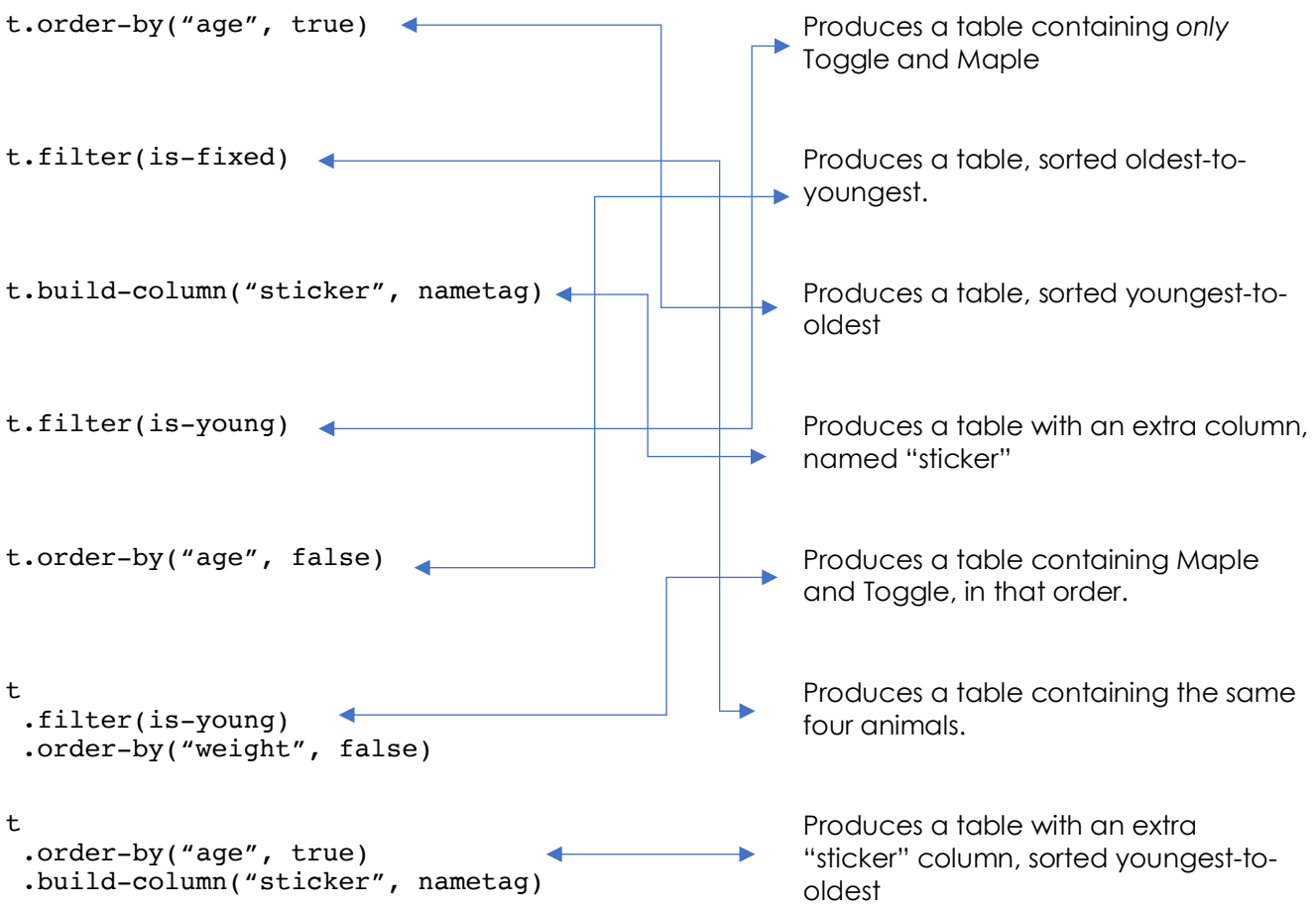
You have the following functions defined below (read them carefully!):

```
fun is-fixed(animal): animal[“fixed”]                    end
fun is-young(animal): animal[“age”] < 4                  end
fun nametag(animal):  text(animal["name"], 20, "red") end
```

The table **t** below represents four animals at the shelter:

name	gender	age	fixed	weight
“Toggle”	“female”	3	true	48
“Fritz”	“male”	4	true	92
“Nori”	“female”	6	true	35.3
“Maple”	“female”	3	true	51.6

Match each Pyret expression (left) to the description of what it does (right).



Unit 4

- Functions can contain value definitions
- We use **Table Plans** to help us use table methods correctly, without making mistakes

Review

1. In the Interactions Area, use table methods to sort your table by one column. **Try sorting your table in both ascending and descending order.**

2. If a researcher is looking at a dataset of students, they might want to divide the data into separate populations of boys and girls. A veterinarian might want to look at only the cats at a shelter. **Copy one of your "filtering" answers from Page 18 below**, to define the filtering criteria you want to use.

Filter animals heavier than 20 pounds

3. In the space below, **use the Design Recipe to write a function that checks if a row in your dataset fits that criteria.** Whatever criteria you choose, it should be true for some rows and false for others. **Type this function into the Definitions Area.**

```
is-large :: (animal :: Row) → Boolean
  name          domain          range
# Consumes an animal and produces true if its weight is greater than 20 lbs
examples:
  is-large(sample1) is sample1["weight"] > 20
  is-large(sample2) is sample2["weight"] > 20
end
fun is-large(animal):
  animal["weight"] > 20
end
```

4. **Use the function to filter your dataset.**

5. Instead of using the function you wrote to *filter* your dataset, **use another table method to build a new column** that shows whether or not each row meets the criteria.

Table Plan

On Kitten Day, the shelter prints up a list of all the cats in their database that are less than 2 years old, and makes nametags for them. They need a function that will help them out! Define a function called `get-kittens-tags`, which takes in the dataset and produces the correct table.

Contract and Purpose

get-kittens-tags **::** *(animals :: Table)* → *Table*

 # Consume a table of animals, and produce a table containing kittens with nametags, sorted by name

Example Tables

Make a Start Table and a result based on that table.

`animals-table` → `get-kittens-tags(animals-table)`

name	species	age	fixed	legs	weight	adopt
Sasha	cat	1	FALSE	4	6.5	4
Toggle	dog	3	TRUE	4	48	3
Buddy	lizard	2	FALSE	4	0.3	12
Wade	cat	1	FALSE	4	3.2	4
Mittens	cat	2	TRUE	4	7.4	5

name	species	age	fixed	legs	weight	adopt	tag
Sasha	cat	1	FALSE	4	6.5	4	Sasha
Wade	cat	1	FALSE	4	3.2	4	Wade

Define the function

Use the relevant methods (circle your helper functions!), then produce a result with the new table.

```
fun get-kittens-tags ( animals ) :
  t = animals                                          Define the table
  .build-column( nametag )                             Are there more columns?
  .filter( is-kitten )                                 Are there fewer rows?
  .order-by( "name", true )                            Are the rows ordered?
  t                                                    Produce the result
end
```

Table Plan

The first weekend of every month, the shelter holds a "meet the dogs" picnic, to encourage families to adopt their dogs. Write a function called `get-dogs-by-age`, that takes their database and produces a table of all the dogs in the shelter, sorted from youngest to oldest.

Contract and Purpose

get-dogs-by-age **::** *(animals :: Table)* → *Table*

 # Consume a table of animals, and produce a table containing only the dogs, sorted by age

Examples

Make a Start Table and a result based on that table.

`animals-table` → `get-dog-by-age(animals-table)`

name	species	age	fixed	legs	weight	adopt
Snowcone	cat	2	TRUE	4	6.1	5
Wade	cat	1	FALSE	4	3.2	4
Hercules	cat	3	FALSE	4	13.4	7
Toggle	dog	3	TRUE	4	48	3
Fritz	dog	4	TRUE	4	92	6

name	species	age	fixed	legs	weight	adopt
Toggle	dog	3	TRUE	4	48	3
Fritz	dog	4	TRUE	4	92	6

Define the function

Use the relevant methods (circle your helper functions!), then produce a result with the new table.

```
fun get-dogs-by-age ( animals ) :
  t = animals                                          Define the table
  .build-column(                                )      Are there more columns?
  .filter(              is-dog                  )      Are there fewer rows?
  .order-by(          "age", true               )      Are the rows ordered?
                                                       Produce the result
end
```

Table Plan

It's important for animals to stay healthy, especially when they get older. The veterinarians at the shelter want to put some of the dogs on a diet! They need a regular report of all the older dogs, sorted from heaviest-to-lightest. Define a function `old-dogs-diet`, which does just that!

Contract and Purpose

old-dogs-diet :: (animals :: Table) → Table

Consumes a table of animals, and produces a table with only old dogs, sorted heaviest to lightest

Examples

Make a Start Table and a result based on that table.

`animals-table` → `old-dogs-diet(animals-table)`

name	species	age	fixed	legs	weight	adopt
Snowcone	cat	2	TRUE	4	6.1	5
Lucky	dog	3	TRUE	3	45.4	9
Mr. PB	dog	10	FALSE	4	161	6
Boo-boo	dog	11	TRUE	4	123	24
Snuggles	tarantula	2	FALSE	8	0.1	1

name	species	age	fixed	legs	weight	adopt
Mr. PB	dog	10	FALSE	4	161	6
Boo-boo	dog	11	TRUE	4	123	24

Define the function

Use the relevant methods (circle your helper functions!), then produce a result with the new table.

```
fun old-dogs-diet ( animals ) :
  t = animals                                          Define the table
  .build-column(                                  )    Are there more columns?
  .filter(                is-old-dog              )    Are there fewer rows?
  .order-by(              "weight", false         )    Are the rows ordered?
                                                       Produce the result
end
```

Table Plan

The shelter is tracking birth-years for all the animals who've been fixed. They need a function that takes in their database and returns a table that contains the birth-year for each one. Define `get-fixed-birth` that will do this for them.

Contract and Purpose

get-fixed-birth :: (animals :: Table) → Table

Consumes a table of animals, produces a new table of animals who have been fixed, with a new column for birth year

Examples

Make a Start Table and a result based on that table.

`animals-table` → `get-fixed-by-legs(animals-table)`

name	species	age	fixed	legs	weight	adopt
Snowcone	cat	2	TRUE	4	6.1	5
Lucky	dog	3	TRUE	3	45.4	9
Hercules	cat	3	FALSE	4	13.4	7
Toggle	dog	3	TRUE	4	48	3
Snuggles	tarantula	2	FALSE	8	0.1	1

name	species	age	fixed	legs	weight	adopt	year
Snowcone	cat	2	TRUE	4	6.1	5	2015
Lucky	dog	3	TRUE	3	45.4	9	2014
Toggle	dog	3	TRUE	4	48	3	2014

Define the function

Use the relevant methods (circle your helper functions!), then produce a result with the new table.

```
fun get-fixed-birth ( animals ) :
  t = animals                                          Define the table
  .build-column( birth-year )                          Are there more columns?
  .filter( is-fixed )                                  Are there fewer rows?
  .order-by( )                                         Are the rows ordered?
  _____                                                Produce the result
end
```

My Dataset

What are two ways you might want to *order* this dataset?

1) [specific to each student]

2)

What are two subsets into which you might *filter this dataset*?

1)

2)

What are two new columns you might want to *build* from this dataset?

1)

2)

Unit 5

- **Bar charts** show the *absolute* quantity of each row in a dataset. The larger the quantity, the longer the bar. Bar charts provide a visual representation of values in a dataset.

- **Pie charts** show the *relative* quantity of each row in a dataset. The greater the percentage, the larger the pie slice. Pie charts provide a visual representation of proportions in a dataset.

- **Choosing a Sample Table** is important when coming up with small examples for Table Plans. A good sample table has:
  1. At least all the relevant columns
  2. Enough rows to accurately represent the dataset
  3. Rows that are randomly-ordered

Statements about Columns

Use the Table below to help you answer the questions.

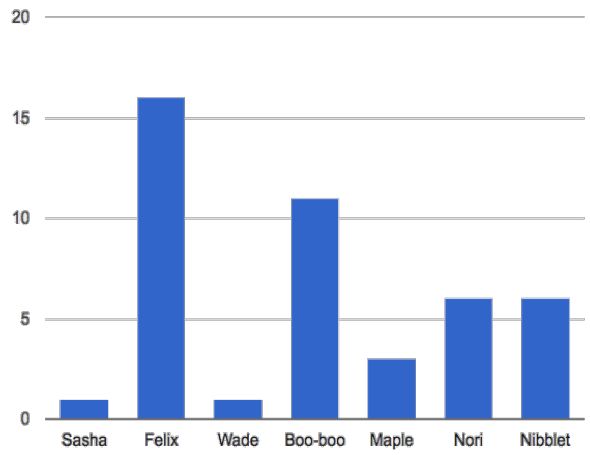
name	species	age	pounds
Sasha	cat	1	6.5
Felix	cat	16	9.2
Wade	cat	1	3.2
Boo-boo	dog	11	123
Maple	dog	3	51.6
Nori	dog	6	35.3
Nibblet	rabbit	6	4.3

1. Which animal(s) is/are the heaviest? Boo-boo
2. Which animal(s) is/are the youngest? Sasha, Wade
3. How much of the *total weight* comes from Maple? 22%
4. How much of the *combined age* comes from Nori? 13%
5. Would these questions be harder to answer if the table had 100 rows? If so, why?

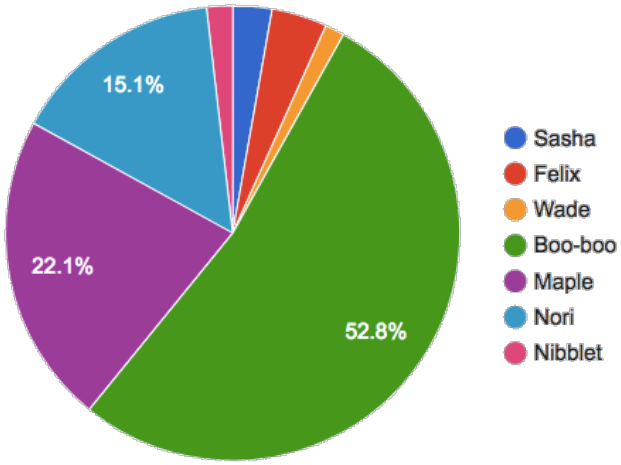
Much harder if you were estimating, because it is harder to calculate a large number of entries without using software.

Visualizing Quantity

In the table below, there are two observations drawn from the following charts. Add two more.



Animals Ages (yrs)



Animals Weights (lbs)

Based on a _____ chart of _____	**I notice that _____**
Based on a **bar chart** of 7 animals' ages	Felix is by far the oldest
Based on a **pie chart** of 7 animals' weights	Boo-boo weighs more than the other six animals combined!
Based on a **bar chart** of 7 animals' ages	Wade and Sasha are the youngest animals
Based on a **pie chart** of 7 animals' weights	Maple is as large as the five smallest animals

Table Plan

Dogs are generally a lot bigger heavier than cats, so the shelter wants to look at a chart of *only* the dogs to determine who needs more exercise time. Define a function `pie-dog-weight`, which will make a pie chart showing the relative weights of all the dogs in the shelter.

Contract and Purpose

pie-dog-weight :: (animals :: Table) → Image

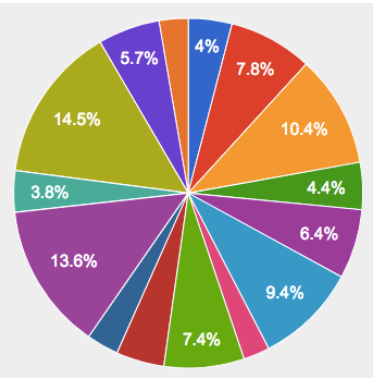
Consumes a table of animals, produces a pie chart of only the dogs and their weights

Examples

Make a Start Table and a result based on that table.

`animals-table` → `pie-dog-weight(animals-table)`

name	...	pounds
Snowcone	...	6.1
Lucky	...	45.4
Hercules	...	13.4
Toggle	...	48
Snuggles	...	0.1



Define the function

Use the relevant methods (circle your helper functions!), then produce a result with the new table.

```
fun pie-dog-weight ( animals ) :
    t = animals                                          Define the table
  .build-column(                                )        Are there more columns?
  .filter(          is-dog                      )        Are there fewer rows?
  .order-by(                                    )        Are the rows ordered?
    pie-chart(t, "name", "pounds")                       Produce the result
end
```

Bad Sample Tables!

For each word problem, a Sample Table must have (1) all the columns that matter, (2) a representative sample of the rows, and be in (3) random order. For each problem below, check the boxes to determine if the Sample Table meets those criteria.

1. The shelter wants to know the median age of all the cats

name	species	age	fixed	legs	pounds	weeks
Sasha	cat	1	FALSE	4	6.5	3
Mittens	cat	2	TRUE	4	7.4	5
Sunfower	cat	5	TRUE	4	8.1	10

- ✓ Relevant columns
- ✓ Representative sample of rows
- ✓ Random order

2. The shelter wants a pie chart showing all the dogs' weight

name	species	age
Fritz	dog	4
Wade	cat	2
Nibblet	rabbit	6
Daisy	dog	5

- ☐ Relevant columns
- ✓ Representative sample of rows
- ✓ Random order

3. Sort all the animals alphabetically by name

name	species	age	fixed	legs	pounds	weeks
Ada	dog	2	TRUE	4	32	3
Bo	dog	4	TRUE	4	76.1	10
Boo-boo	dog	11	TRUE	4	123	10

- ✓ Relevant columns
- ☐ Representative sample of rows
- ☐ Random order

4. Make a bar chart for all the fixed animals

name	species	age	fixed	legs	pounds	weeks
Sasha	cat	1	FALSE	4	6.5	3

- ☐ Relevant columns
- ☐ Representative sample of rows
- ☐ Random order

Table Plan

Define a function `bar-kitten-adoption`, which takes in a Table of animals and creates a bar chart showing how many weeks it took for each kitten to be adopted

Contract and Purpose

bar-kitten-adoption :: (animals :: Table) → Image

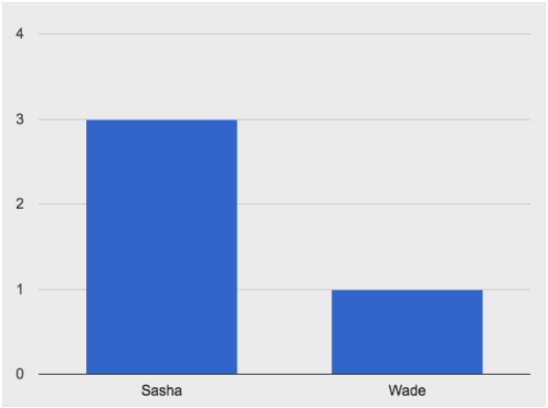
Consumes a table of animals, produces a bar chart showing the weeks it took for each kitten to be adopted

Examples

Make a Start Table and a result based on that table.

animals-table → bar-kitten-adoption(animals-table)

name	species	age	fixed	pounds	weeks
Sasha	cat	1	FALSE	6.5	3
Wade	cat	1	FALSE	7.4	1
Sunfower	cat	5	TRUE	8.1	10
Boo-boo	dog	11	TRUE	123	10



Define the function

Use the relevant methods (circle your helper functions!), then produce a result with the new table.

```
fun bar-kitten-adoption ( animals ) :
    t = animals                                          Define the table
    .build-column(                                )
    .filter(          is-kitten                   )      Are there more columns?
    .order-by(                                    )      Are there fewer rows?
                                                         Are the rows ordered?
        bar-chart(t, "name", "weeks")                    Produce the result
end
```

Table Plan

Contract and Purpose

_____ **::** _____ → _____

_____

_____

Examples

Make a Start Table and a result based on that table.

_____ → _____

Define the function

Use the relevant methods (circle your helper functions!), then produce a result with the new table.

fun _____ (_____) **:**

t = _____

_____

_____

_____

_____

end

Define the table

Are there more columns?

Are there fewer rows?

Are the rows ordered?

Produce the result

Table Plan

Contract and Purpose

_____ **::** _____ → _____

_____

_____

Examples

Make a Start Table and a result based on that table.

_____ → _____

Define the function

Use the relevant methods (circle your helper functions!), then produce a result with the new table.

fun _____ (_____) **:**

t = _____

_____

_____

_____

_____

end

Define the table

Are there more columns?

Are there fewer rows?

Are the rows ordered?

Produce the result

Visualizing My Dataset

What quantity charts did you make, and what do you notice? Fill in the table below.

Based on a _____ chart of _____	**I notice that _____**

Unit 6

- There are three ways to measure the "center" of a dataset, to talk about a whole column of data using just one number:

    1. The **mean** of a dataset is the average of all the numbers

    2. The **median** of a dataset is a value that is smaller than half the dataset, and larger than the other half

    3. The **modes** of a dataset are the numbers that appear the most often.

- Data Scientists can also measure the "variation" of a dataset using a **five number summary:**

    1. The **minimum** – the smallest value in the dataset

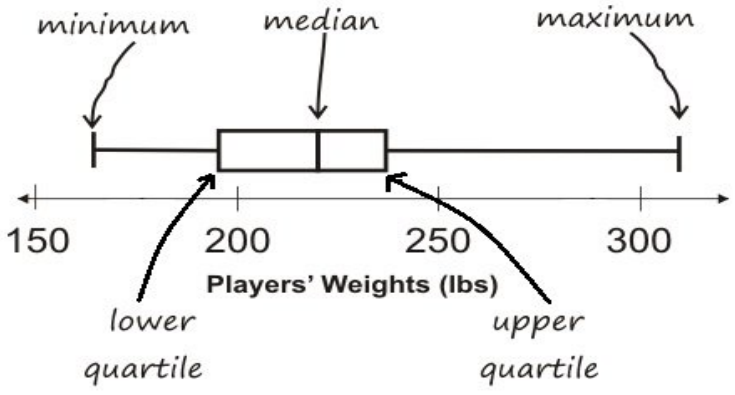
    2. The **first, or "lower" quartile (Q1)** – the median value that separates the first quarter of the values in the dataset from the second quarter

    3. The **second quartile (Q2)** – the median value which separates the entire dataset into "top" and "bottom" halves.

    4. The **third, or "upper" quartile (Q3)** – the median value that separates the third quarter of the values in the dataset from the fourth quarter

    5. The **maximum** – the largest value in the dataset

- The **five number summary** can be used to draw a **box-and-whisker plot.**



Summarizing Columns in `Animals`

The column I choose to measure is **weeks**

Measures of Center

The three measures for this column are:

Mean (Average)	Median	Mode(s)
6.0689	4	1

Based on the differences between mean and median, I conclude :

On average, animals stay at the shelter for about 6 weeks, but half of all the animals were adopted after 4 weeks or less.

Measures of Variation

My five-number summary is:

Minimum	Q1	Q2 (Median)	Q3	Maximum
1	2.5	4	8	30

A box plot can be drawn from this summary on the number line below:

From this summary and box-plot, I conclude:

The vast majority of animals are adopted before 8 weeks in the shelter, but there are a number of outliers (such as the maximum of 30)

Table Plan

The shelter wants a summary of the variation in ages among the dogs. Write a function called `variation-dog-age` that will take in a table of animals and produce a box-plot that shows this variation.

Contract and Purpose

variation-dog-age :: (animals :: Table) → Image

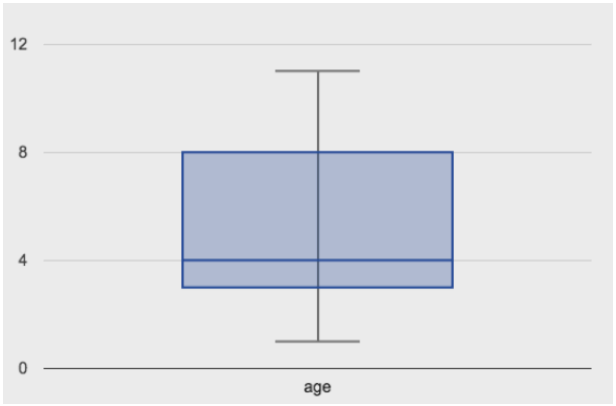
Consumes a table of animals, produces a box plot showing the variation of ages among only the dogs

Examples

Make a Start Table and a result based on that table.

`animals-table` → `variation-dog-age(animals-table)`

name	species	age	fixed	legs	weight	weeks
Snowcone	cat	2	TRUE	4	6.1	5
Lucky	dog	3	TRUE	3	45.4	9
Hercules	cat	3	FALSE	4	13.4	7
Toggle	dog	3	TRUE	4	48	3
Snuggles	tarantula	2	FALSE	8	0.1	1



Define the function

Use the relevant methods (circle your helper functions!), then produce a result with the new table.

```
fun variation-dog-age ( animals ) :
  t = animals                                          Define the table
    .build-column(                                )    Are there more columns?
    .filter(          is-dog                      )    Are there fewer rows?
    .order-by(                                    )    Are the rows ordered?
      box-plot(t, "age")                               Produce the result
end
```

Interpreting Variation

Consider the following list dataset, representing the annual income of ten people:

`$65k, $12k, $14k, $280k, $15k, $22k, $45k, $34k, $45k, $175k`

1. In the space below, rewrite this dataset in **sorted order**.

`$12k, $14k, $15k, $22k, $34k, $45k, $45k, $65k, $175k, $280k`

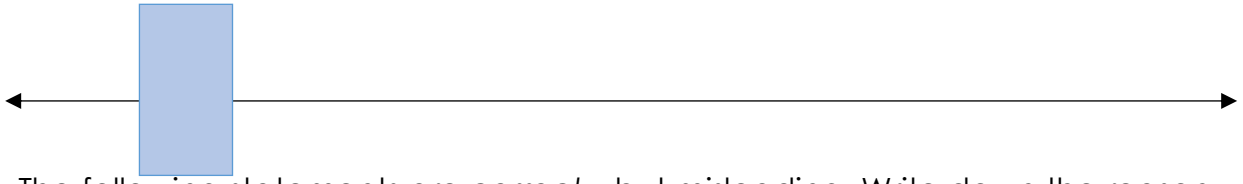
2. In the table below, compute the **measures of center** for this dataset.

Mean (Average)	Median	Mode(s)
70,700	39,500	45,000

3. In the table below, compute the **five number summary** of this dataset.

Minimum	Q1	Q2 (Median)	Q3	Maximum
12,000	15,000	39,500	65,000	280,000

4. On the number line below, draw a **box plot** for this dataset.



5. The following statements are *correct*…but misleading. Write down the reason why.

Statement	Why it's misleading
"They're rich! The average person makes more than $70k dollars!"	While the mean is close to $70k, there are some very high earning outliers pushing the average up
"It's a middle-income list: the most common salary is $45k/yr!"	Looking at the full dataset, more than half of the entries are people making less than $45k, making the mode misleading
"This group is really diverse, with people making as little as 12k and as much as $280k!"	While the spread of incomes is large, the vast majority are still making less than $65k, with very high earning outliers.

Summarizing a Column in My Dataset

The column I choose to measure is [Specific to each student]

Measures of Center

The three measures for this column are:

Mean (Average)	Median	Mode(s)

Based on the differences between mean and median, I conclude :

Measures of Variation

My five-number summary is:

Minimum	Q1	Q2 (Median)	Q3	Maximum

A box plot can be drawn from this summary on the number line below:

From this summary and box-plot, I conclude:

Unit 7

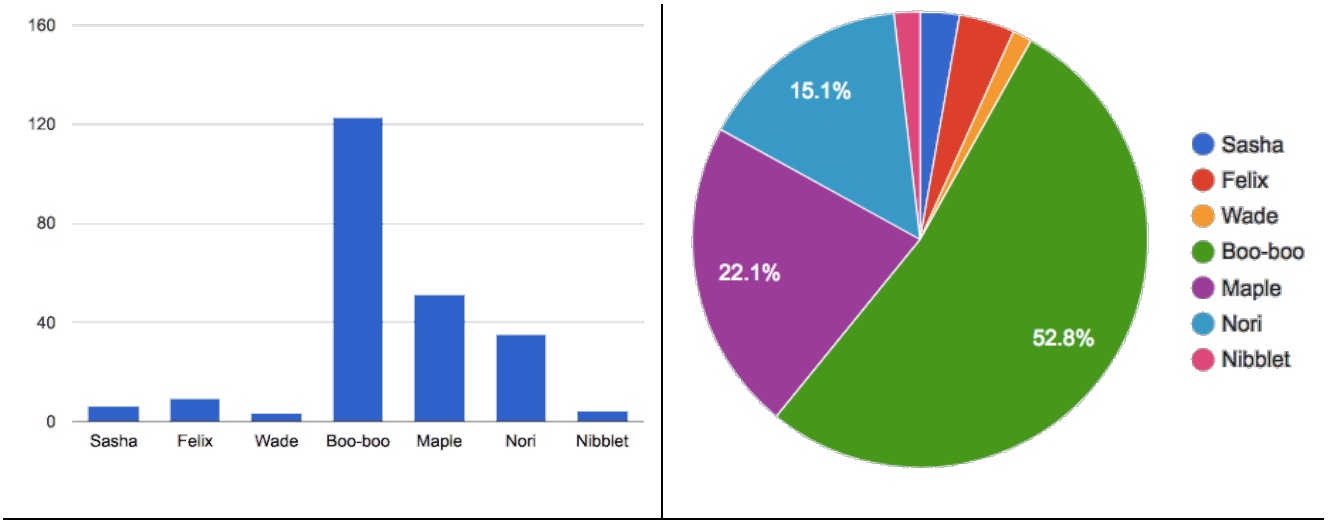
- **Frequency Bar charts** show the number of rows belonging to a given category. The more rows in each category, the longer the bar. Frequency bar charts provide a visual representation of the frequency of values in a **categorical** column. Since categorical data cannot be ordered, there is no strict ordering of bars in a frequency bar chart.

- **Histograms** show the number of rows that fall within certain ranges, or "bins" of a dataset. The more rows that that fall within a particular "bin", the longer the bar. Histograms provide a visual representation of the frequency of values in a **quantitative** column. Quantitative data can be ordered, so the bars of a histogram are always sorted.

- When dealing with histograms, it's important to select a good **bin size**. If the bins are too small or too large, it is difficult to see the distribution in the dataset.

Visualizing Quantity (Review)

Use the charts below to help you answer the questions.



Animals Weights (lbs)

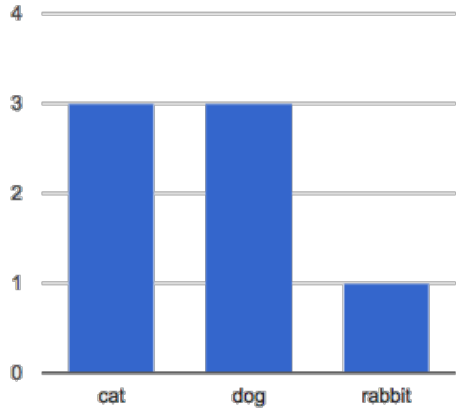
1. Which animal is the heaviest? Boo-boo
2. Which animal is the lightest? Wade
3. How much of the *total weight* comes from Maple? 22.1%
4. How much of the *total weight* comes from Nori? 15.1%
5. Which chart did you use for questions 1 and 2? Bar chart
6. Which chart did you use for questions 3 and 4? Pie chart
7. Why are some questions easier to answer with one kind of chart or another?

Bar charts are better for finding the amount of something quickly, whereas pie charts are better for seeing the percentage of something relative to the total for other elements in a table.

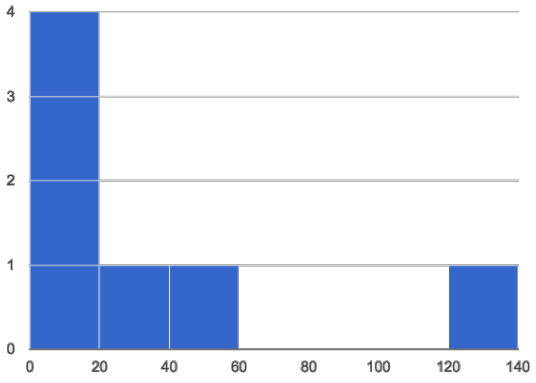
Visualizing Frequency

name	species	age	pounds
"Sasha"	"cat"	1	6.5
"Boo-boo"	"dog"	11	123
"Felix"	"cat"	16	9.2
"Nori"	"dog"	6	35.3
"Wade"	"cat"	1	3.2
"Nibblet"	"rabbit"	6	4.3
"Maple"	"dog"	3	51.6

1. How many cats are there? 3
2. How many dogs are there? 3
3. How many animals are between 3-6 years old? 3
4. How many weigh between 0-5 pounds? 2
5. Are there more animals weighing 0-5 than 6-10 pounds? Yes
6. The charts below are based on the Sample Table above. What is each one measuring? Write down your guess underneath each one.



Amount of each species



Frequency of animal weights

Table Plan

Define a function `freq-bar-gender`, which takes in a Table of animals and creates a frequency bar chart showing how many animals are male v. female.

Contract and Purpose

freq-bar-gender :: (animals :: Table) → Image

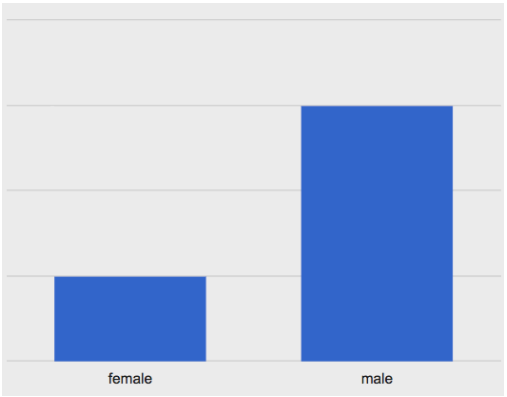
Consumes a table of animals and produces a frequency bar chart of their genders

Examples

Make a Start Table and a result based on that table.

animals-table → freq-bar-gender(animals-table)

name	species	age	gender
Fritz	dog	4	male
Wade	cat	2	male
Nibblet	rabbit	6	male
Daisy	dog	5	female



Define the function

Use the relevant methods (circle your helper functions!), then produce a result with the new table.

```
fun freq-bar-gender ( animals ) :
  t = animals
  
  
  
  freq-bar-chart(t, "gender")
end
```

Define the table

Are there more columns?

Are there fewer rows?

Are the rows ordered?

Produce the result

Table Plan

Define a function `histogram-adoption`, which takes in a Table of animals and creates a histogram showing how long it took for animals to get adopted

Contract and Purpose

histogram-adoption :: (animals :: Table) → Image

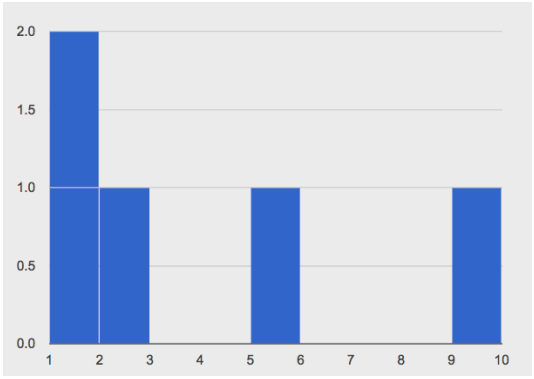
Consumes a table of animals and produces a histogram showing how long it took for animals to get adopted

Examples

Make a Start Table and a result based on that table.

animals-table → histogram-adoption(animals-table)

name	species	age	fixed	legs	weight	weeks
Snowcone	cat	2	TRUE	4	6.1	5
Lucky	dog	3	TRUE	3	45.4	9
Hercules	cat	3	FALSE	4	13.4	7
Toggle	dog	3	TRUE	4	48	3
Snuggles	tarantula	2	FALSE	8	0.1	1



Define the function

Use the relevant methods (circle your helper functions!), then produce a result with the new table.

```
fun histogram-adoption ( animals ) :
    t = animals
    
    
    
    histogram(t, "weeks", 1)
end
```

Define the table

Are there more columns?

Are there fewer rows?

Are the rows ordered?

Produce the result

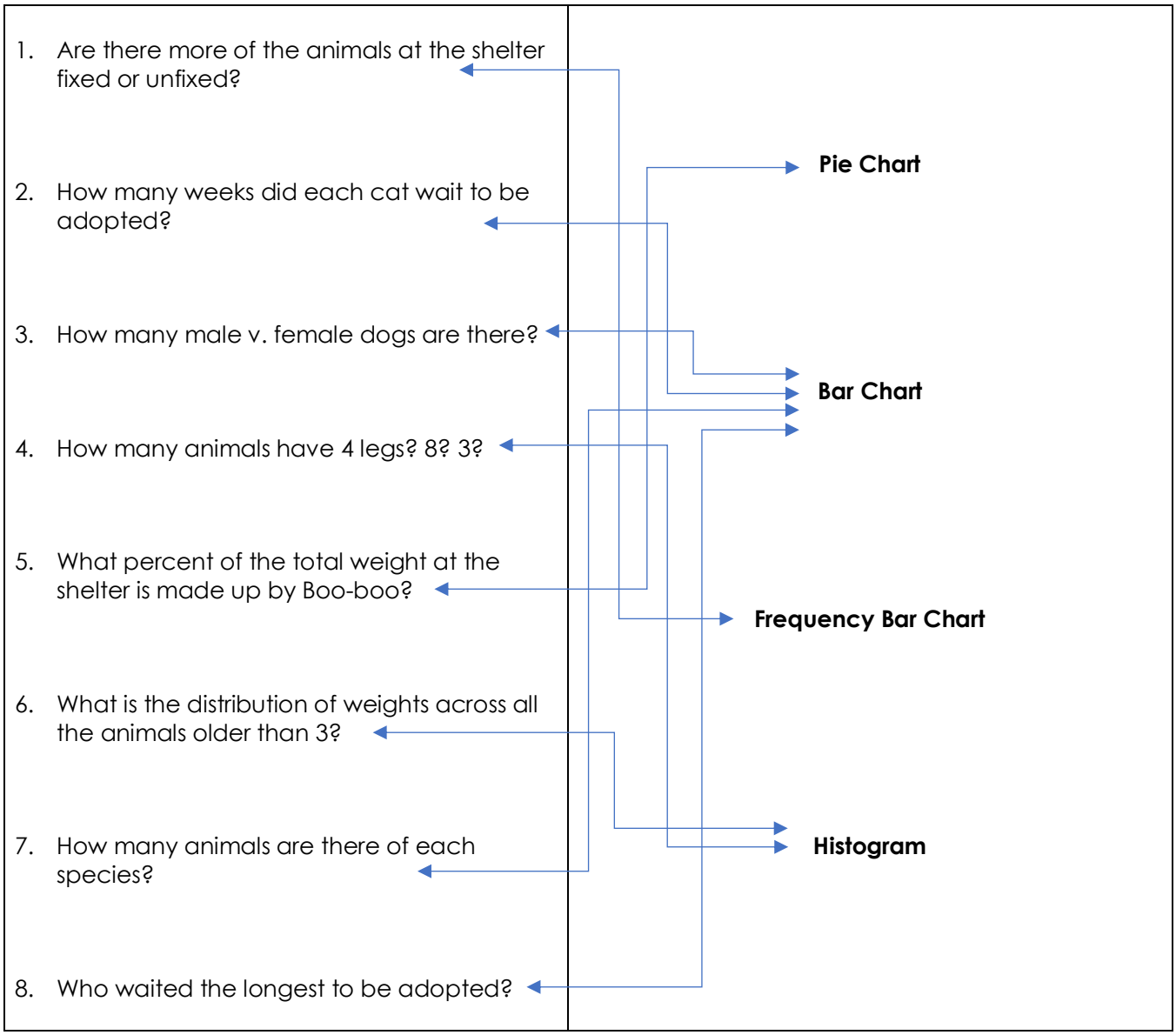
Visualizing My Dataset

What frequency charts did you make, and what do you notice? Fill in the table below.

Based on a _____ chart of _____	**I notice that _____**

Matching Charts to Questions

For each of the questions below, draw a line to the chart that will best answer it. (You may find that more than one question is best answered by the same chart!)



Unit 8

- **Scatter Plots** show the relationship between two quantitative columns. Each row in the dataset is represented by a point, with one column providing the x-value and the other providing the y-value. The resulting "point cloud" makes it possible to look for a relationship between those two columns.

- If the points in a scatter plot appear to follow a pattern, it is possible that a relationship – or **correlation** – exists between those two columns.

- If there is a pattern to the points in a scatter plot, points that are far away from the pattern are called **outliers**.

- We can express this correlation by drawing line through the data cloud, so that the distance between the line and each of the points is as small as possible. This line is called the **line of best fit** – or **predictor function** - and allows us to make predictions based on the dataset.

(Dis)Proving a Claim

"Younger animals are cuter, so they get adopted faster."

Do you agree? If so, why?

I hypothesize…

that younger animals *will* get adopted faster, possibly because they are considered cuter, but there may be other factors causing them to get adopted faster.

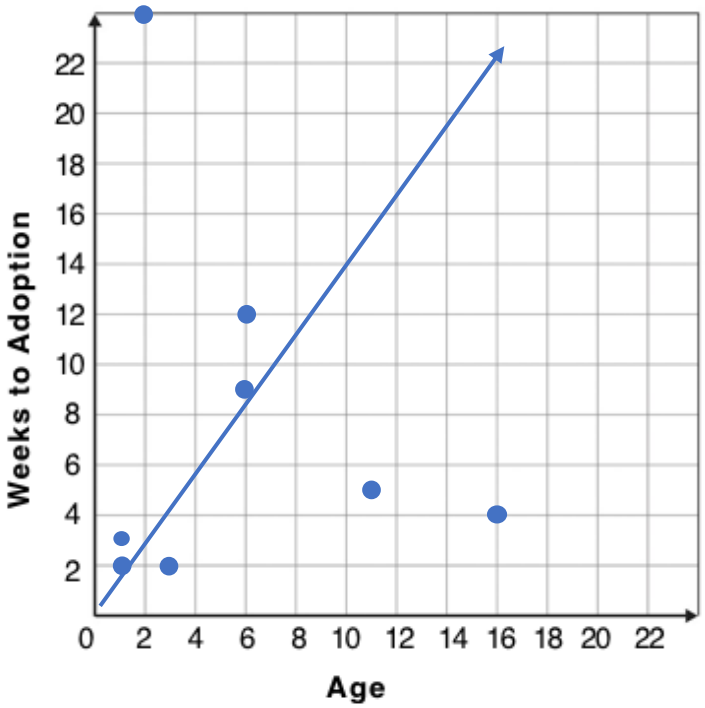
[specific to each student]

What would you look for in the dataset to see if you are right?

I would look at both the ages and number of weeks until adoption for each animal to see if there was a correlation. I would also want to collect more data, such as conduct a survey of adopters.

Creating a Scatter Plot

name	species	age	weeks
"Sasha"	"cat"	1	3
"Boo-boo"	"dog"	11	5
"Felix"	"cat"	16	4
"Buddy"	"lizard"	2	24
"Nori"	"dog"	6	9
"Wade"	"cat"	1	2
"Nibblet"	"rabbit"	6	12
"Maple"	"dog"	3	2



1. **For each row in the Sample Table on the left, add a point to the scatter plot on the right**. The first 3 rows have been completed for you. Use the values from the `age` column for the x-axis, and values from the `weeks` column for the y-axis.

2. Do you see a pattern? Do the points seem to shift up or down as age increases? **Draw a line on the scatter plot to show this pattern**.

3. Does the line slope upwards or downwards? Slightly upwards

4. Are the points mostly close to the line? A few points are close to the line, but as ages increase the points get much farther apart.

Table Plan

Define a function `dogs-age-weeks`, which takes in a Table of animals and creates a scatter plot of all the dogs, tracking their `age` on the x-axis and the number of `weeks` it took for them to be adopted on the y-axis.

Contract and Purpose

dog-age-weeks :: (animals :: Table) → Image

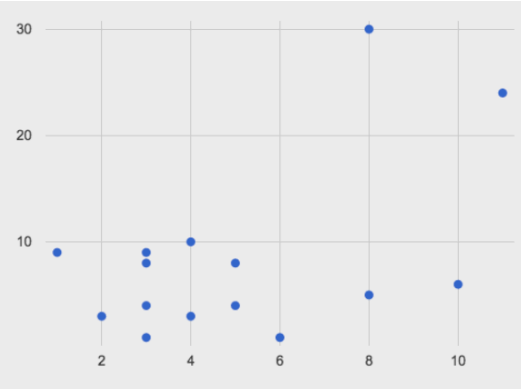
Consumes a table of animals and produces a scatter plot showing the relationship between age and weeks to adoption

Examples

Make a Start Table and a result based on that table.

animals-table → dog-age-weeks(animals-table)

name	species	age	fixed	legs	weight	weeks
Snowcone	cat	2	TRUE	4	6.1	5
Lucky	dog	3	TRUE	3	45.4	9
Hercules	cat	3	FALSE	4	13.4	7
Toggle	dog	3	TRUE	4	48	3
Snuggles	tarantula	2	FALSE	8	0.1	1



Define the function

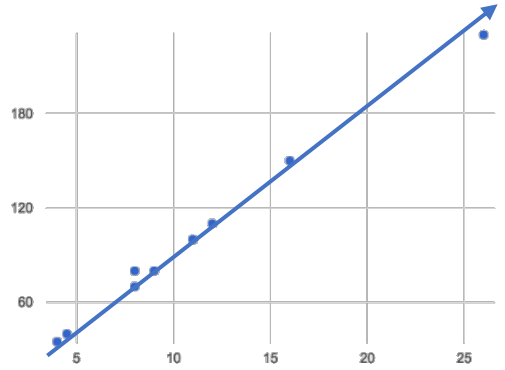
Use the relevant methods (circle your helper functions!), then produce a result with the new table.

```
fun dog-age-weeks (animals):
  t = animals-table                                    Define the table
                                                       Are there more columns?
                                                       Are there fewer rows?
  .filter( is-dog )                                    Are the rows ordered?
  scatter-plot(t, "age", "weeks")                      Produce the result
end
```

Drawing Predictors

For each of the scatter plots below, draw a **predictor line** that fits best.

A

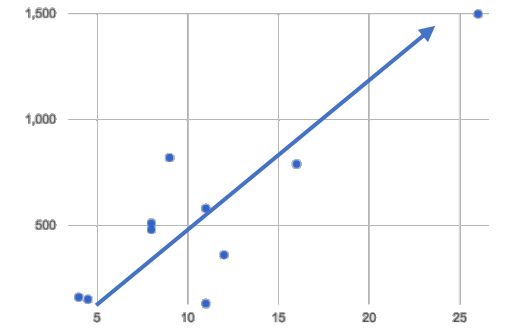


fat (g) v. calories-from-fat in common menu items

Direction: (Positive) Negative None

Strength: (Strong) Weak

B

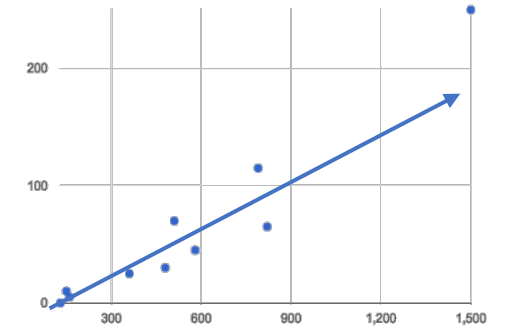


fat (g) v. sodium (g) in common menu items

Direction: (Positive) Negative None

Strength: (Strong) Weak

C

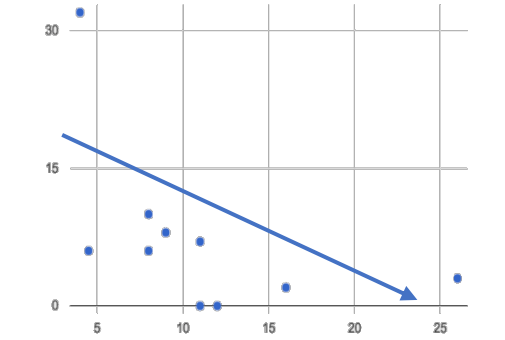


sodium (g) v. cholesterol (mg) in common menu items

Direction: (Positive) Negative None

Strength: (Strong) Weak

D



fat (g) v. sugar (g) in common menu items

Direction: Positive (Negative) None

Strength: Strong (Weak)

Correlations in My Dataset

1) There may be a correlation between _____ and
column

_____. I think it is a _____, _____
column strong / weak positive / negative

correlation, because _____

_____. It would be stronger if I looked

at _____.
a subset or extension of my data

---

1) There may be a correlation between _____ and
column

_____. I think it is a _____, _____
column strong / weak positive / negative

correlation, because _____

_____. It would be stronger if I looked

at _____.
a subset or extension of my data

---

1) There may be a correlation between _____ and
column

_____. I think it is a _____, _____
column strong / weak positive / negative

correlation, because _____

_____. It would be stronger if I looked

at _____.
a subset or extension of my data

Unit 9

- Given a **predictor function** and a scatter plot, we can compute the error by adding the squares of all the distances between the function and each point in the plot. The error is called the **r^2 statistic**, which tells us *how much of the variation in the y-axis can be explained by the x-axis.*

- A **strong correlation** will have a large r^2. A **weak correlation** will have a small r^2.

- A **positive correlation** means the slope of the line of best fit is positive. A **negative correlation** means the slope is negative.

- **Linear Regression** is a way of computing the **line of best fit**, by taking a scatter plot and deriving the slope and y-intercept for a line that has the smallest possible r^2.

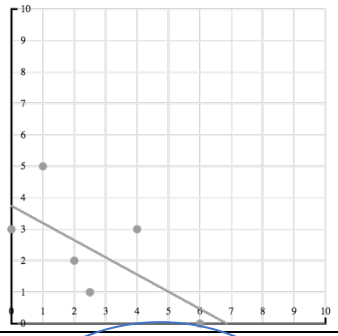
- **Correlation is not causation**! Correlation only suggests that two measures are *related*, but does not tell us if one *causes* the other. For example, hot days are *correlated* with people running their air conditioners, air conditioners do not *cause* hot days!

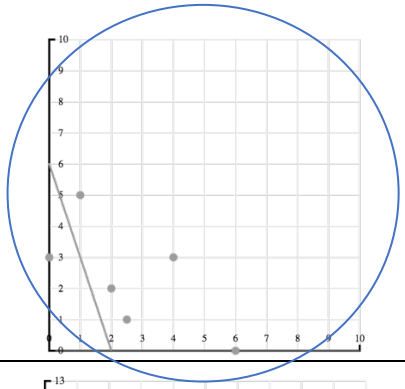
Grading Predictors

Below are the scatter plots for data sets A-D, with two different lines predictor lines drawn on top. For plots A-D:

1. Circle the plot with the line that fits better
2. Give the plot you circled a grade between 0 (no correlation) and 1 (perfect correlation)

A

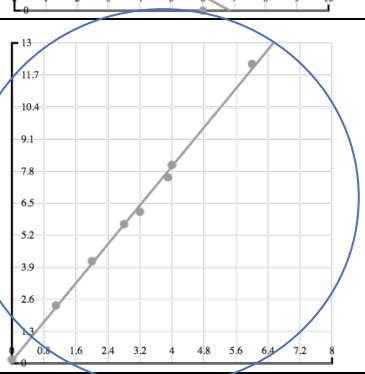


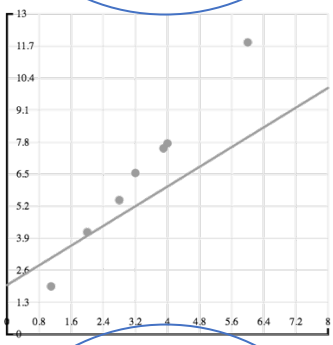


Strength of Correlation:

0.2

B

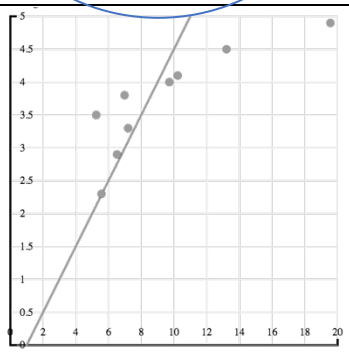


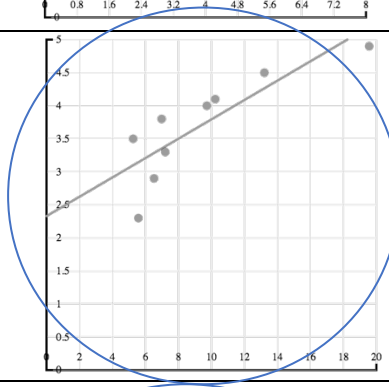


Strength of Correlation:

0.95

C

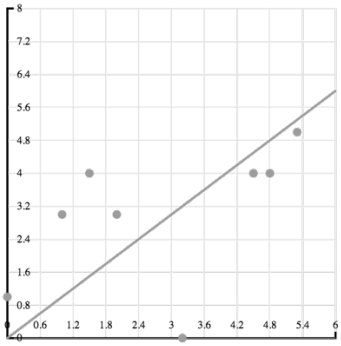


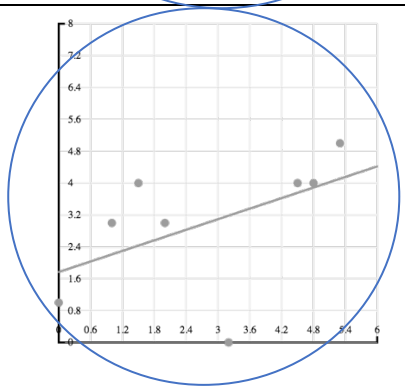


Strength of Correlation:

0.65

D





Strength of Correlation:

0.4

Findings in the `animals` Dataset

I performed a linear regression on dogs at the shelter (dataset or subset), and found a weak (r^2=0.25), positive (a strong/weak (r^2=…), positive/negative) correlation between age of the dogs (in weeks) [x-axis] and number of weeks to be adopted [y-axis]. From this, I conclude that 25% of the variability in adoption time is explained (r^2 % of the variation in [y-axis] is explained by [x-axis]) by the age of the dog.

I performed a linear regression on cats at the shelter (dataset or subset), and found a weak (r^2=0.025), positive (a strong/weak (r^2=…), positive/negative) correlation between weight of the cats (in pounds) [x-axis] and number of weeks to be adopted [y-axis]. From this, I conclude that 2.5% of the variability in adoption time is explained (r^2 % of the variation in [y-axis] is explained by [x-axis]) by the weight of the cat.

I performed a linear regression on fixed animals at the shelter (dataset or subset), and found a weak (r^2=0.025), positive (a strong/weak (r^2=…), positive/negative) correlation between age of the animal [x-axis] and weight of the animal (in pounds) [y-axis]. From this, I conclude that 2.5% of the variability in weight is explained (r^2 % of the variation in [y-axis] is explained by [x-axis]) by the age of the animal.

Correlations in My Dataset

I performed a linear regression on ______________________________, and
dataset or subset

found ______________________________ correlation between
a strong/weak (r^2=…), positive/negative

______________________ and ______________________. From this, I
[x-axis] [y-axis]

conclude that ______________________________
r^2 % of the variation in [y-axis] is explained by [x-axis]

______________________________.

I performed a linear regression on ______________________________, and
dataset or subset

found ______________________________ correlation between
a strong/weak (r^2=…), positive/negative

______________________ and ______________________. From this, I
[x-axis] [y-axis]

conclude that ______________________________
r^2 % of the variation in [y-axis] is explained by [x-axis]

______________________________.

I performed a linear regression on ______________________________, and
dataset or subset

found ______________________________ correlation between
a strong/weak (r^2=…), positive/negative

______________________ and ______________________. From this, I
[x-axis] [y-axis]

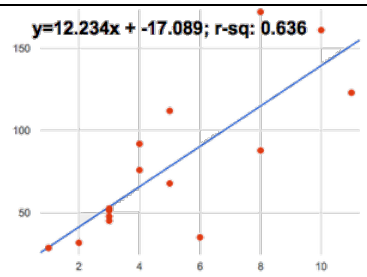
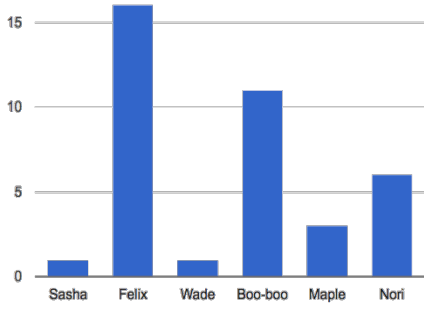
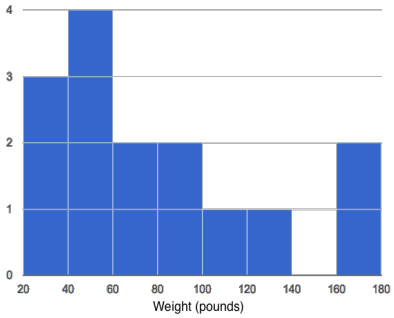
conclude that ______________________________
r^2 % of the variation in [y-axis] is explained by [x-axis]

______________________________.

Unit 10

Fake News!

Every claim below is *wrong*! Your job is to figure out why, by looking at the data.

	Data	Claim	Why it's *wrong*
1	The average player on a basketball team is 6'1".	*"Most of the players on the team are taller than 6'."*	The average is based on all the players, and there may be outliers pushing the average height up-average tells you nothing about the majority of the players.
2	After performing linear regression on census data, a positive correlation (r^2=0.18) was found between people's height and salary.	*"Taller people get paid more."*	Only 18% of the variation in salary is based on height, which is not a large enough r-squared value to say that taller people get paid more.
3		*"According to the predictor function indicated here, the value on the x-axis is will predict the value on the y-axis 63.6% of the time."*	The r-squared value of 0.636 does not mean how often the y-value will be predicted, rather what percent of variation in the y-value is based on the x-value.
4	 Bar Chart of Pet Ages	*"According to this bar chart, Felix makes up a little more than 15% of the total ages of all the animals in the dataset."*	Bar charts are not the most appropriate image for showing the percentage of each measurement based on the total- pie charts should be used for that info. This bar chart shows that Felix is a little more than 15 years old.
5		*"According to this histogram, most animals weigh between 40 and 60 pounds."*	More animals fit into the histogram bin between 40-60 lbs than any other bin, but that doesn't mean that most animals weigh between 40-60 lbs.
6	After performing linear regression, a negative correlation (r^2=0.91) was found between the number of hairs on a person's head and their likelihood of owning a wig.	*"Owning wigs causes people to go bald."*	Though there is a strong correlation between hair and owning a wig, correlation does NOT equal causation.

Blank Recipes, Table Plans, and References

Design Recipes

```
_____________________::_____________________ → _____________
        name                    domain                range
# _____________________________________________________________
examples:

    _____________(_________) is _____________________________

    _____________(_________) is _____________________________
end

fun _____________ (_______) :  _____________________________

end
```

```
_____________________::_____________________ → _____________
        name                    domain                range
# _____________________________________________________________
examples:

    _____________(_________) is _____________________________

    _____________(_________) is _____________________________
end

fun _____________ (_______) :  _____________________________

end
```

Design Recipes

```
_____::_____ → _____
 name   domain   range
# _____
```

examples:

```
    _____(_____) is _____

    _____(_____) is _____
end

fun _____ (_____) : _____

end
```

```
_____::_____ → _____
 name   domain   range
# _____
```

examples:

```
    _____(_____) is _____

    _____(_____) is _____
end

fun _____ (_____) : _____

end
```

Design Recipes

```
_____________________::_____________________ → _____________
        name                    domain                range
# _____________________________________________________________
```

examples:

```
    _____________(_________) is _____________________________

    _____________(_________) is _____________________________
end

fun _____________ (_______) :  _____________________________

end
```

```
_____________________::_____________________ → _____________
        name                    domain                range
# _____________________________________________________________
```

examples:

```
    _____________(_________) is _____________________________

    _____________(_________) is _____________________________
end

fun _____________ (_______) :  _____________________________

end
```

Table Plan

Contract and Purpose

_____ : : _____ → _____

_____

Examples

Make a Start Table and a result based on that table.

_____ → _____

Define the function

Use the relevant methods (circle your helper functions!), then produce a result with the new table.

fun _____ (_____) :

t = _____

_____

_____

_____

_____

end

Define the table

Are there more columns?

Are there fewer rows?

Are the rows ordered?

Produce the result

Table Plan

Contract and Purpose

_____ **::** _____ → _____

_____

Examples

Make a Start Table and a result based on that table.

_____ → _____

Define the function

Use the relevant methods (circle your helper functions!), then produce a result with the new table.

fun _____ (_____) **:**

t = _____

_____

_____

_____

_____

end

Define the table

Are there more columns?

Are there fewer rows?

Are the rows ordered?

Produce the result

Table Plan

Contract and Purpose

_____ **::** _____ → _____

_____

Examples

Make a Start Table and a result based on that table.

_____ → _____

Define the function

Use the relevant methods (circle your helper functions!), then produce a result with the new table.

fun _____ (_____) **:**

t = _____

_____

_____

_____

_____

end

Define the table

Are there more columns?

Are there fewer rows?

Are the rows ordered?

Produce the result

Contracts

Name	Domain		Range
triangle	:: (side :: *Number*, style :: *String*, color :: *String*)	→	*Image*
circle	:: (radius :: *Number*, style :: *String*, color :: *String*)	→	*Image*
star	:: (radius :: *Number*, style :: *String*, color :: *String*)	→	*Image*
rectangle	:: (width :: *Num*, height :: *Num*, style :: *Str*, color :: *Str*)	→	*Image*
ellipse	:: (width :: *Num*, height :: *Num*, style :: *Str*, color :: *Str*)	→	*Image*
square	:: (size :: *Number*, style :: *String*, color :: *String*)	→	*Image*
text	:: (str :: *String*, size :: *Number*, color :: *String*)	→	*Image*
overlay	:: (img1 :: *Image*, img2 :: *Image*)	→	*Image*
rotate	:: (degree :: *Number*, img :: *Image*)	→	*Image*
scale	:: (factor :: *Number*, img :: *Image*)	→	*Image*
string-repeat	:: (text :: *String*, repeat :: *Number*)	→	*String*
string-contains	:: (text :: *String*, search-for :: *String*)	→	*Boolean*
num-sqr	:: (n :: *Number*)	→	*Number*
num-sqrt	:: (n :: *Number*)	→	*Number*
num-min	:: (a :: *Number, b:: Number*)	→	*Number*
num-max	:: (a :: *Number, b:: Number*)	→	*Number*
get-row	:: (t :: *Table, index :: Number*)	→	*Row*

Contracts

Name	Domain		Range
<Table>.row-n	:: (n :: *Number*)	→	*Row*
<Table>.order-by	:: (col :: *String*, *increasing* :: *Boolean*)	→	*Table*
<Table>.filter	:: (test :: *(Row → Boolean)*)	→	*Table*
<Table>.build-column	:: (col :: *String*, *builder* :: *(Row → Value)*)	→	*Table*
mean	:: (t :: *Table*, col :: *String*)	→	*Number*
median	:: (t :: *Table*, col :: *String*)	→	*Number*
modes	:: (t :: *Table*, col :: *String*)	→	*List<Number>*
bar-chart	:: (t :: *Table*, labels :: *String*, values :: *String*)	→	*Image*
pie-chart	:: (t :: *Table*, labels :: *String*, values :: *String*)	→	*Image*
box-plot	:: (t :: *Table*, col:: *String*)	→	*Image*
freq-bar-chart	:: (t :: *Table*, values :: *String*)	→	*Image*
histogram	:: (t :: *Table*, values :: *String*, bin-width :: *Number*)	→	*Image*
scatter-plot	:: (t :: *Table*, xs :: *String*, ys :: *String*)	→	*Image*
labeled-scatter-plot	:: (t :: *Table*, labels :: *String*, xs :: *String*, ys :: *String*)	→	*Image*
lr-plot	:: (t :: *Table*, xs :: *String*, ys :: *String*)	→	*Image*
labeled-lr-plot	:: (t :: *Table*, labels :: *String*, xs :: *String*, ys :: *String*)	→	*Image*