Problem Solving and Abstraction (CMPU 101)

Tom Ellman

Lecture 7

Reminder

- We won't cover everything in class!
- Follow along with the assigned readings.
- Active Reading:
  - Keep Pyret open and try examples.
  - Take notes.
- In lab and on assignments you'll be expected to try things that may only be in the readings – or may be new altogether.
- Lab and homework are additional opportunities for learning!

Where are we?

- We can enter tabular data directly in Pyret
- We can load it from an external source like a Google spreadsheet.
- We can filter tables to get particular rows.
- We can compute values for each row to add as a new column.
- We can order tables by the values in a particular column.
- We can visualize tabular data with plots.

Is your data reliable? (Probably Not)

- Good data scientists never trust a dataset without first making sure that the values make sense.

- Visualizations and plots can help data scientists identify data they might have missed that still needs to be cleaned/normalized.

Invalid Data

A	B	C	D
name	**eligible**	**height**	**weight**
Allie	FALSE	64	130
GIGI	TRUE	68	150
Elan	TRUE	72	185
Lavon	0	62	130
NUNU	1	70	170

Wrong Data Type

```
sportbook1 =  #Load Fails
  load-table: name, eligible, height, weight
    source:
      load-spreadsheet(dd-ssid).sheet-by-name("sportbook", true)
  end
```

There were worksheet importing errors.

All items in every column must have the same type. We expected to find a Bool at cell B5, but we instead found this Number: 0.

All items in every column must have the same type. We expected to find a Bool at cell B6, but we instead found this Number: 1.

To make the data end up in the format we want, we'll use sanitizers, which convert data from an external source into a specific Pyret data type.

Built-in Sanitizers:

- string-sanitizer
  - Replaces missing values with ""
  - Converts non-string data to strings, e.g., 3 to "3"
- num-sanitizer
  - Replaces missing values with 0
  - Converts numeric strings to numbers, e.g., "3" to 3

Sanitizers are just functions, so you can write your own!

```
sportbook2 =
  load-table: name, eligible, height, weight
    source:
      load-spreadsheet(dd-ssid).sheet-by-name("sportbook", true)
    sanitize name using string-sanitizer
    sanitize eligible using bool-sanitizer
    sanitize height using num-sanitizer
    sanitize weight using num-sanitizer
  end
```

name	eligible	height	weight
"Allie"	false	64	130
"GIGI"	true	68	150
"Elan"	true	72	185
"Lavon"	false	62	130
"NUNU"	true	70	170

Notice that the problematic Boolean values (0/1) are now corrected.

Missing Data

A	B	C	D
name	**NRO**	**asmt1**	**asmt2**
Allie	FALSE	85	90
		75	60
ELAN	TRUE	95	63
Lavon	FALSE		88
NUNU	TRUE	70	0

```
gradebook1a =
  load-table: name, NRO, asmt1, asmt2
    source:
      load-spreadsheet(dd-ssid).sheet-by-name("gradebook", true)
  end
```

name	NRO	asmt1	asmt2
some("Allie")	some(false)	some(85)	90
none	none	some(75)	60
some("ELAN")	some(true)	some(95)	63
some("Lavon")	some(false)	none	88
some("NUNU")	some(true)	some(70)	0

Option Data Type

none	Missing Data
some(“Allie”)	Present Data

If one cell in a column is missing, the entire column is converted to option type.

Working with Option Values

```
fun string-worker(s :: Option) -> String:
  cases(Option) s:
    | some(a) => a
    | none => "Anonymous"
  end
end

fun bool-worker(s :: Option) -> Boolean:
  cases(Option) s:
    | some(a) => a
    | none => false
  end
end

fun num-worker(s :: Option) -> Number:
  cases(Option) s:
    | some(a) => a
    | none => 0
  end
end
```

```
gradebook1b =  #Load Fails
  load-table: name, NRO, asmt1, asmt2
    source:
      load-spreadsheet(gb-ssid).sheet-by-name("gradebook", true)
      sanitize name using string-sanitizer
      sanitize NRO using bool-sanitizer
      sanitize asmt1 using num-sanitizer
      sanitize asmt2 using num-sanitizer
  end
```

```
"Cannot sanitize the empty cell at column NRO, row 1 as a boolea
```

(Show program evaluation trace...)

```
gradebook2a =
  load-table: name, NRO, asmt1, asmt2
    source:
      load-spreadsheet(dd-ssid).sheet-by-name("gradebook", true)
      sanitize name using option-sanitizer(string-sanitizer)
      sanitize NRO using option-sanitizer(bool-sanitizer)
      sanitize asmt1 using option-sanitizer(num-sanitizer)
      sanitize asmt2 using option-sanitizer(num-sanitizer)
  end
```

Combining option sanitizer with other sanitizers. If the type sanitizer succeeds, use it's value, otherwise return none.

name	NRO	asmt1	asmt2
some("Allie")	some(false)	some(85)	some(90)
none	none	some(75)	some(60)
some("ELAN")	some(true)	some(95)	some(63)
some("Lavon")	some(false)	none	some(88)
some("NUNU")	some(true)	some(70)	some(0)

There are Many Publicly Available Data Sets

- There are a staggering number of publicly available data sets that we can load from a spreadsheet.

- Take a look at the archives of the Data is Plural newsletter: data-is-plural.com

We can use Pyret to explore these data sets and transform them so they’re easier for us to use.

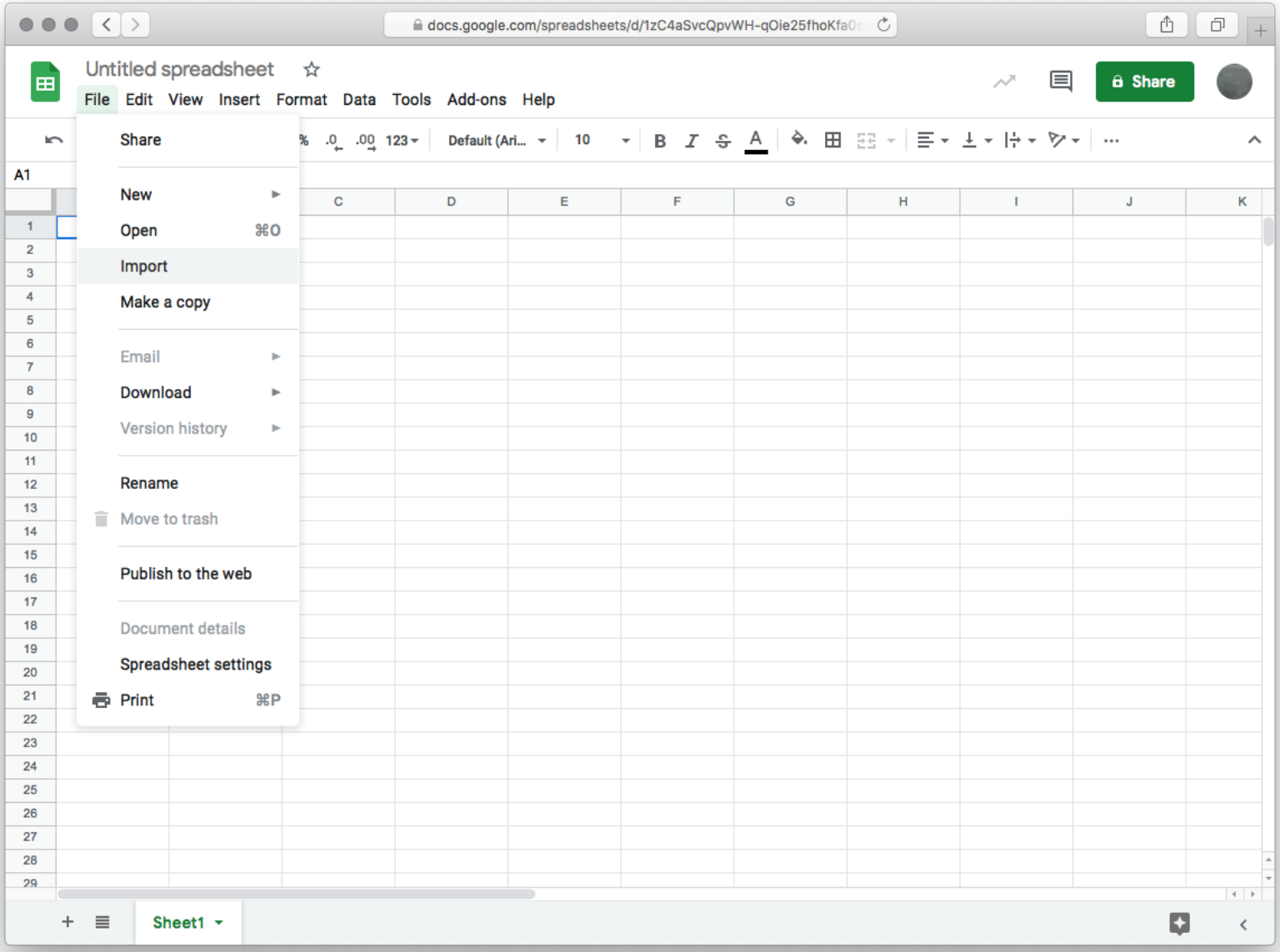
The London Fire Brigade responds to hundreds of requests to rescue animals each year.

Since 2009 they've kept a record of these events:

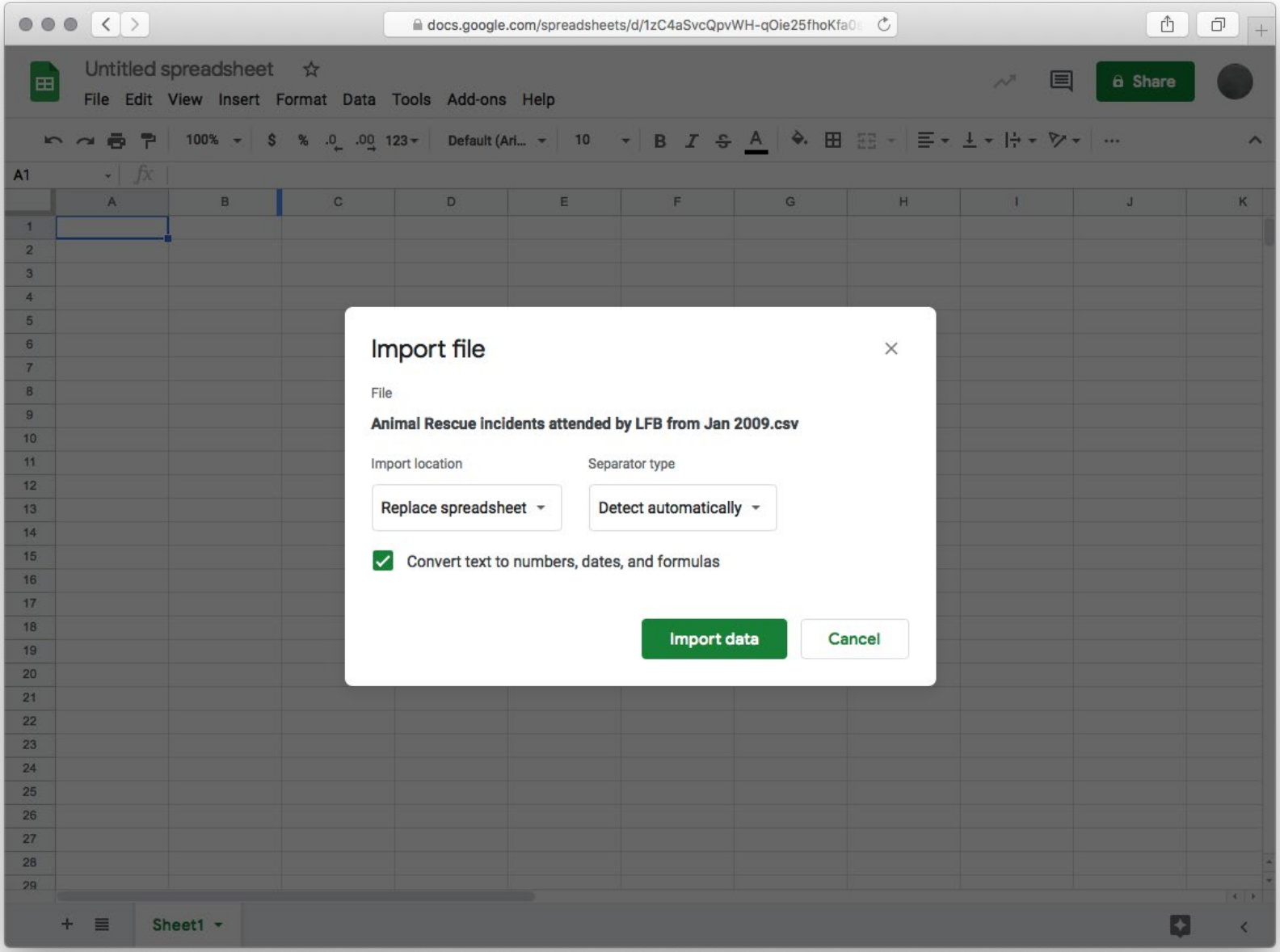
data.london.gov.uk/dataset/animal-rescue-incidents-attended-by-lfb

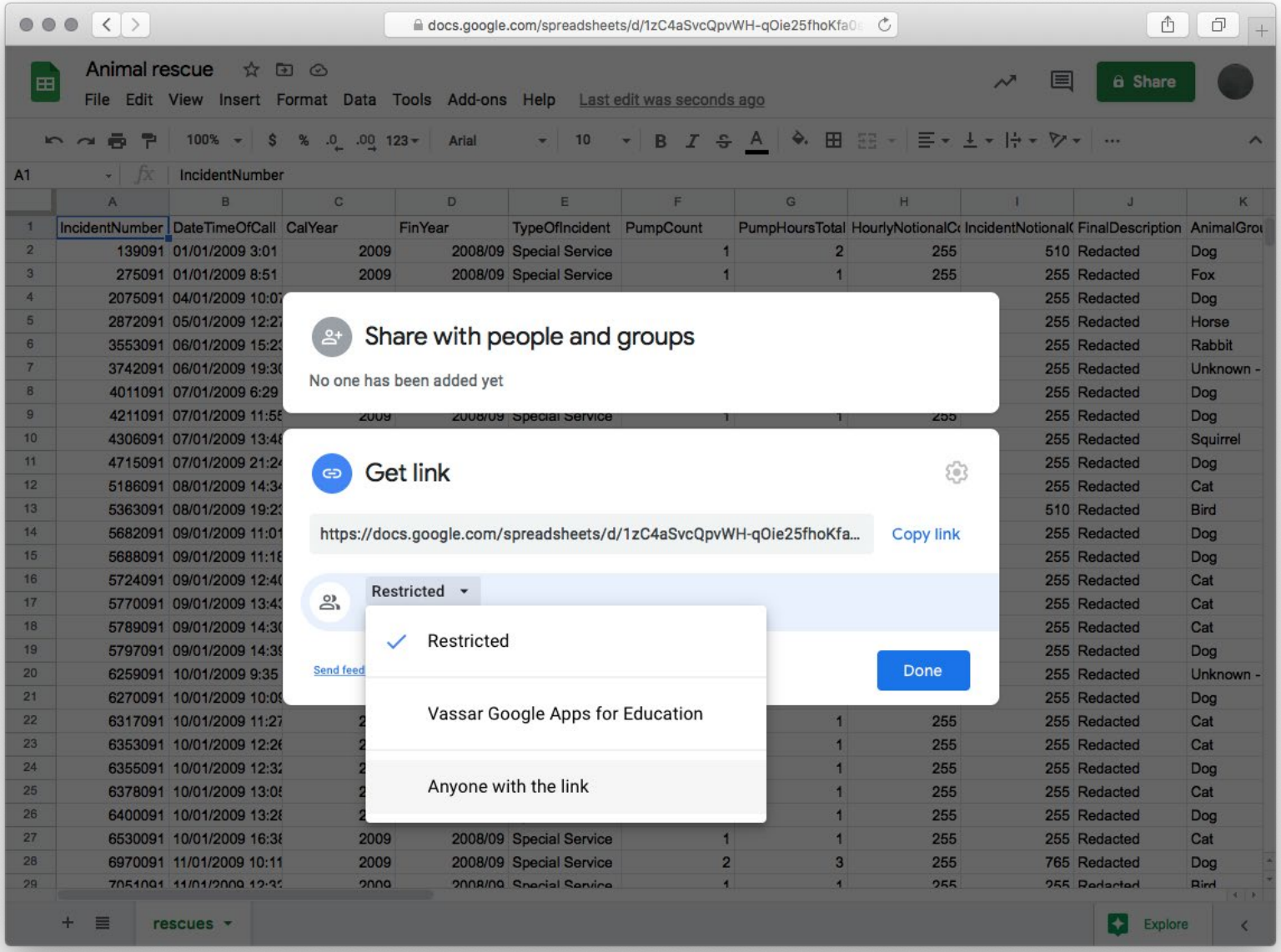
This data is available as CSV – a plain-text file where each cell of the spreadsheet is separated by commas.

To load it into Pyret, we can first upload it to a Google spreadsheet.



Use the Edit—Option menu item.


docs.google.com/spreadsheets/d/1zC4aSvcQpvWH-qOie25fhoKfa0
Untitled spreadsheet
File Edit View Insert Format Data Tools Add-ons Help
Share
Import file
File
Animal Rescue incidents attended by LFB from Jan 2009.csv
Import location
Replace spreadsheet
Separator type
Detect automatically
Convert text to numbers, dates, and formulas
Import data
Cancel
Sheet1


docs.google.com/spreadsheets/d/1zC4aSvcQpvWH-qOie25fhoKfa
Animal rescue
File Edit View Insert Format Data Tools Add-ons Help Last edit was seconds ago
Share
A1 IncidentNumber
Share with people and groups
No one has been added yet
Get link
https://docs.google.com/spreadsheets/d/1zC4aSvcQpvWH-qOie25fhoKfa...
Copy link
Restricted
Restricted
Vassar Google Apps for Education
Anyone with the link
Done
Send feed
rescues
Explore

Too many columns!

We could copy-and-paste the names to have our column names in Pyret, but, instead, let’s trim columns we don’t care about first.

```
# UK Pet Rescue
rss-ssid = "1JWfZkiVirEskNwaLuszuJJ8tkjCwZMEqeNyp_jdLawI"

rescue-data1 =
  load-table: DateTimeOfCall, CalYear, AnimalGroupParent, ward, borough
    source:
      load-spreadsheet(rss-ssid).sheet-by-name("Animal Rescue LFB", true)
  end
```

	ward	borough
	some("Crystal Palace & Upper Norwood")	some("Croydon")
	some("Woodside")	some("Croydon")
	some("Carshalton Central")	some("Sutton")
	some("Harefield")	some("Hillingdon")
	some("Gooshays")	some("Havering")
	some("Alibon")	some("Barking and Dagenham")

Some ward and borough data is missing.

```
rescue-data2 =
  load-table: DateTimeOfCall, CalYear, AnimalGroupParent, ward, borough
    source:
      load-spreadsheet(rss-ssid).sheet-by-name("Animal Rescue LFB", true)
    sanitize DateTimeOfCall using string-sanitizer
    sanitize CalYear using num-sanitizer
    sanitize AnimalGroupParent using string-sanitizer
    sanitize borough using string-sanitizer
    sanitize ward using string-sanitizer
  end
```

String sanitizer converts missing values to empty string. Thus all values are present and option type not needed.

ward	borough
"Crystal Palace & Upper Norwood"	"Croydon"
"Woodside"	"Croydon"
"Carshalton Central"	"Sutton"
"Harefield"	"Hillingdon"
"Gooshays"	"Havering"
"Alibon"	"Barking and Dagenham"

	CalYear	AnimalGroupParent
	2009	"Dog"
	2009	"Fox"
	2009	"Dog"
	2009	"Horse"
	2009	"Rabbit"

Num sanitizer converts “2009” string to 2009 number.

```
fun just-time(date-time :: String) -> String:
  str = string-substring(date-time, 11, 13)
  if string-contains(str,":") :
    string-substring(str,0,1)
  else:
    str
  end
where:
  just-time("01/01/2009 03:01") is "03"
  just-time("06/01/2009 15:23") is "15"
end

rescue-data3 = transform-column(rescue-data2, "DateTimeOfCall", just-time)
```

The transform-column function is used to clean up or otherwise alter the data in a single column of a table.
It returns a new table by applying its function argument to each value of the given column.

DateTimeOfCall	CalYear
"3"	2009
"8"	2009
"10"	2009
"12"	2009
"15"	2009

```
freq-bar-chart(rescue-data3, "AnimalGroupParent")
freq-bar-chart(rescue-data3, "CalYear")
freq-bar-chart(rescue-data-2020, "borough")
```

Reality is more complicated than imagination!

- Unlike data sets we create for exposition, real data sets often have:
- Missing values
- Values of wrong data type
- Same data expressed different ways

  7/4/1987 vs. 4/7/1987

  July 4, 1987 vs. 4 July 1987 …
- Differing levels of precision:
  - E.g., Tue vs. Tue @ Noon.
  - 1987 vs. 4 July 1987

Checking Email Addresses

```
# Checking Email Addresses.

email1 = "thellman@vassar.edu"
email2 = "thellmanvassar.edu"
email3 = "@vassar.edu"
email4 = "thellman@vassar"

fun emailp(em :: String ) -> Boolean:
  string-contains(em,"@")
  and
  not(string-index-of(em,"@") == 0)
  and
  (string-index-of(em,".edu") == (string-length(em) - 4))
where:
  emailp(email1) is true
  emailp(email2) is false
  emailp(email3) is false
  emailp(email4) is false
end
```

Checking URLs, Times and Dates