Lecture 2: Introduction to Julia

Dataset

We will use Traffic Crash Reports data from Cincinnati City.

Data description: Traffic Crash Reports are records in the event of a CPD response to a traffic crash. The source of this data is the City of Cincinnati Police Department. The column names for this data are self explantory.

Filename: "Traffic_Crash_Reports__CPD__Aug2018.csv" *Make sure this file in the same directory as the ipynb file*

Setup: Use Julia 0.6.4 kernel. Install the packages CSV, Gadfly, Cairo and Fontconfig.

In [1]:
```
# using Pkg
# Pkg.add("CSV")
# Pkg.add("Gadfly")
# Pkg.add("Cairo")
# Pkg.add("Fontconfig")
# Pkg.add("RDatasets")
# Pkg.add("DataFrames")
# Pkg.add("GLM")
```

In [2]:
```
# Pkg.add("CSV",VersionNumber("0.2.5"));
# Pkg.add("Gadfly",VersionNumber("0.8.0"));
# Pkg.add("Cairo",VersionNumber("0.5.6"));
# Pkg.add("Fontconfig",VersionNumber("0.1.1"))
# Pkg.add("RDatasets",VersionNumber("0.4.0"))
```

Use the packages...

In [3]:
```
using CSV, DataFrames, Gadfly, Cairo, Fontconfig, RDatasets;
```

Questions

Write Julia code to answer the following questions:

Q 1: Load this data (Traffic_Crash_Reports__CPD__Aug2018.csv) into memory.

In [4]:
```
data = DataFrame(CSV.File("Traffic_Crash_Reports__CPD__Aug2018.csv", delim="
```

In [5]:
```
typeof(data)
```

```
DataFrame
```

In [6]:
```
# data = CSV.read(
#     "Traffic_Crash_Reports__CPD__Aug2018.csv",
#     DataFrame;
#     delim=",",
#     # missingstring="N/A",
#     );
# typeof(data)
```

In [7]:
```
first(data, 5)
```

5×25 DataFrame

Row	ADDRESS_X	LATITUDE_X	LONGITUDE_X	AGE	COMMUNITY_COUNCIL_NE
	String31	Float64?	Float64?	String15	String31
1	47XX READING RD	39.1721	84.4678	18-25	N/A
2	29XX MONTANA AV	39.1489	84.5963	31-40	N/A
3	39XX W LIBERTY ST	39.118	-84.578	26-30	WEST PRICE HILL
4	29XX WASSON RD	39.1436	84.4347	41-50	N/A
5	S I75 AT 1-8 MM	39.1148	-84.5319	26-30	WEST END

Q 2: What is the size of the dataset? How many data points and how many attributes?

In [8]:
```
size(data)
```

```
(2567, 25)
```

=> There are 2567 data points (number of rows) and 25 attributes (number of cols)

Q 3: Create a new Dataframe 'new_data' by selecting the columns AGE, CRASHSEVERITY, DAYOFWEEK, GENDER, INJURIES, LIGHTCONDITIONSPRIMARY, LOCALREPORTNO, MANNEROFCRASH, ROADSURFACE, WEATHER, and ZIP

*Use the new_data Dataframe for **Q4** and **Q5**.*

In [9]:
```
names(data)
```

```
25-element Vector{String}:
 "ADDRESS_X"
 "LATITUDE_X"
 "LONGITUDE_X"
 "AGE"
 "COMMUNITY_COUNCIL_NEIGHBORHOOD"
 "CPD_NEIGHBORHOOD"
 "CRASHDATE"
 "CRASHLOCATION"
 "CRASHSEVERITY"
 "CRASHSEVERITYID"
 ⋮
 "LOCALREPORTNO"
 "MANNEROFCRASH"
 "ROADCONDITIONSPRIMARY"
 "ROADCONTOUR"
 "ROADSURFACE"
 "SNA_NEIGHBORHOOD"
 "TYPEOFPERSON"
 "WEATHER"
 "ZIP"
```

In [10]:
```
new_data = data[:, ["AGE", "CRASHSEVERITY", "DAYOFWEEK", "GENDER", "INJURIES
                    "LIGHTCONDITIONSPRIMARY", "LOCALREPORTNO", "MANNEROFCRAS
                    "ROADSURFACE", "WEATHER", "ZIP"]];
```

Just to make sure new_data has the same rows as data and 11 cols

In [11]:
```
size(new_data)
```

```
(2567, 11)
```

Q 4: Using describe() function, list the different element types in the new data frame. Also list the columns in which there are missing values.

In [12]:
```
describe(new_data)
```

11×7 DataFrame

Row	variable	mean	min	median	max	n
	Symbol	Union...	Any	Union...	Any	Ir
1	AGE		18-25		UNKNOWN	
2	CRASHSEVERITY		1 - FATAL INJURY		3 - PROPERTY DAMAGE ONLY (PDO)	
3	DAYOFWEEK		FRI		WED	
4	GENDER		F - FEMALE		M - MALE	
5	INJURIES		1 - NO INJURY / NONE REPORTED		5 - FATAL	
6	LIGHTCONDITIONSPRIMARY		1 - DAYLIGHT		9 - UNKNOWN	
7	LOCALREPORTNO		185010686		LCP18080700027	
8	MANNEROFCRASH		1 - NOT COLLISION BETWEEN TWO MOTOR VEHICLES IN TRANSPORT		9 - UNKNOWN	
9	ROADSURFACE		1 - CONCRETE		6 - OTHER	
10	WEATHER		1 - CLEAR		9 - OTHER/UNKNOWN	
11	ZIP	45217.2	45202	45216.0	45251	

=> Gender, Injuries, and Zip columns have missing values since they have non-zero values at nmissing attribute

Q 5: Create a new dataframe 'newdata_nomissing' by removing the rows in the missing values from the new_data Dataframe. How many rows have been removed in this process?

In [13]:
```
new_data_nomissing = dropmissing(new_data);
```

In [14]:
```
size(new_data)
```

```
(2567, 11)
```

In [15]:
```
size(new_data_nomissing)
```

```
(2250, 11)
```

In [16]:
```
size(new_data)[1] - size(new_data_nomissing)[1]
```

```
317
```

=> 317 rows have been removed

Q 6: Generate a list of the different types of crashes in this data.

In [17]:
```
names(new_data_nomissing)
```

```
11-element Vector{String}:
 "AGE"
 "CRASHSEVERITY"
 "DAYOFWEEK"
 "GENDER"
 "INJURIES"
 "LIGHTCONDITIONSPRIMARY"
 "LOCALREPORTNO"
 "MANNEROFCRASH"
 "ROADSURFACE"
 "WEATHER"
 "ZIP"
```

In [18]:
```
unique(new_data_nomissing.CRASHSEVERITY)
```

```
3-element Vector{String31}:
 "3 - PROPERTY DAMAGE ONLY (PDO)"
 "2 - INJURY"
 "1 - FATAL INJURY"
```

Q 7: Generate a list of the different types of WEATHER conditions in this data.

In [19]:
```
unique(new_data_nomissing.WEATHER)
```

```
5-element Vector{String31}:
 "1 - CLEAR"
 "2 - CLOUDY"
 "4 - RAIN"
 "9 - OTHER/UNKNOWN"
 "3 - FOG, SMOG, SMOKE"
```

Q 8: Determine the number of crashes happened in each of these weather conditions using by() function.

In [20]:
```
by(new_data_nomissing, "WEATHER", nrow)
```

```
UndefVarError: by not defined



Stacktrace:

 [1] top-level scope

   @ ~/Main/Class/Probalistic Model/module_0/jl_notebook_cell_df34fa98e69747
e1a8f8a730347b8e2f_X52sZmlsZQ==.jl:1
```

by() seems to be deprecated, I'll use combine() instead

In [21]:
```
combine(groupby(new_data_nomissing, "WEATHER"), nrow => "Number of crashes")
```

5×2 DataFrame

Row	WEATHER	Number of crashes
	String31	Int64
1	1 - CLEAR	1519
2	2 - CLOUDY	402
3	4 - RAIN	321
4	9 - OTHER/UNKNOWN	7
5	3 - FOG, SMOG, SMOKE	1

Q 9: Generate a list of the different light conditions in this data.

In [22]:
```
unique(new_data_nomissing.LIGHTCONDITIONSPRIMARY)
```

```
7-element Vector{String}:
 "1 - DAYLIGHT"
 "5 - DARK - ROADWAY NOT LIGHTED"
 "9 - UNKNOWN"
 "4 - DARK - LIGHTED ROADWAY"
 "6 - DARK - UNKNOWN ROADWAY LIGHTING"
 "2 - DAWN"
 "3 - DUSK"
```

Q 10: Determine the number of crashes happened in each combination of weather and light conditions using by() function. State which combination of weather and ligt conditions result in most number of crashes.

In [23]:
```
combine(groupby(new_data_nomissing, ["WEATHER", "LIGHTCONDITIONSPRIMARY"]),
```

22×3 DataFrame

Row	WEATHER	LIGHTCONDITIONSPRIMARY	Number of crashes
	String31	String	Int64
1	1 - CLEAR	1 - DAYLIGHT	1200
2	1 - CLEAR	5 - DARK – ROADWAY NOT LIGHTED	9
3	1 - CLEAR	9 - UNKNOWN	1
4	1 - CLEAR	4 - DARK - LIGHTED ROADWAY	259
5	1 - CLEAR	3 - DUSK	37
6	1 - CLEAR	6 - DARK – UNKNOWN ROADWAY LIGHTING	4
7	1 - CLEAR	2 - DAWN	9
8	2 - CLOUDY	1 - DAYLIGHT	347
9	2 - CLOUDY	5 - DARK – ROADWAY NOT LIGHTED	2
10	2 - CLOUDY	9 - UNKNOWN	1
11	2 - CLOUDY	4 - DARK - LIGHTED ROADWAY	37
12	2 - CLOUDY	3 - DUSK	9
13	2 - CLOUDY	2 - DAWN	6
14	4 - RAIN	1 - DAYLIGHT	245
15	4 - RAIN	5 - DARK – ROADWAY NOT LIGHTED	3
16	4 - RAIN	4 - DARK - LIGHTED ROADWAY	62
17	4 - RAIN	3 - DUSK	3
18	4 - RAIN	2 - DAWN	8
19	9 - OTHER/UNKNOWN	1 - DAYLIGHT	1
20	9 - OTHER/UNKNOWN	9 - UNKNOWN	2
21	9 - OTHER/UNKNOWN	4 - DARK - LIGHTED ROADWAY	4
22	3 - FOG, SMOG, SMOKE	4 - DARK - LIGHTED ROADWAY	1

=> Combination Clear and Daylight results in most number of crashes (1200)

Q 11: How many ZIP codes are covered in this data.

In [24]:
```
zip_codes = unique(new_data_nomissing.ZIP)
```

```
33-element Vector{Int64}:
 45237
 45211
 45205
 45208
 45214
 45213
 45223
 45219
 45229
 45220
     ⋮
 45203
 45221
 45251
 45228
 45233
 45230
 45212
 45231
 45215
```

=> 33 ZIP Codes

For the following questions that involve generating plots, you may use the white_panel theme.

In [25]:
```
white_panel = Theme(
    panel_fill=colorant"white",
    default_color=colorant"blue",
    major_label_font_size=16pt,
    minor_label_font_size=12pt,
    major_label_color=colorant"black",
    minor_label_color=colorant"black"
);
```

Q 12: Plot a bar graph showing the number of accidents in each of the ZIP codes

Dataframe storing number of accidents in each of the ZIP codes

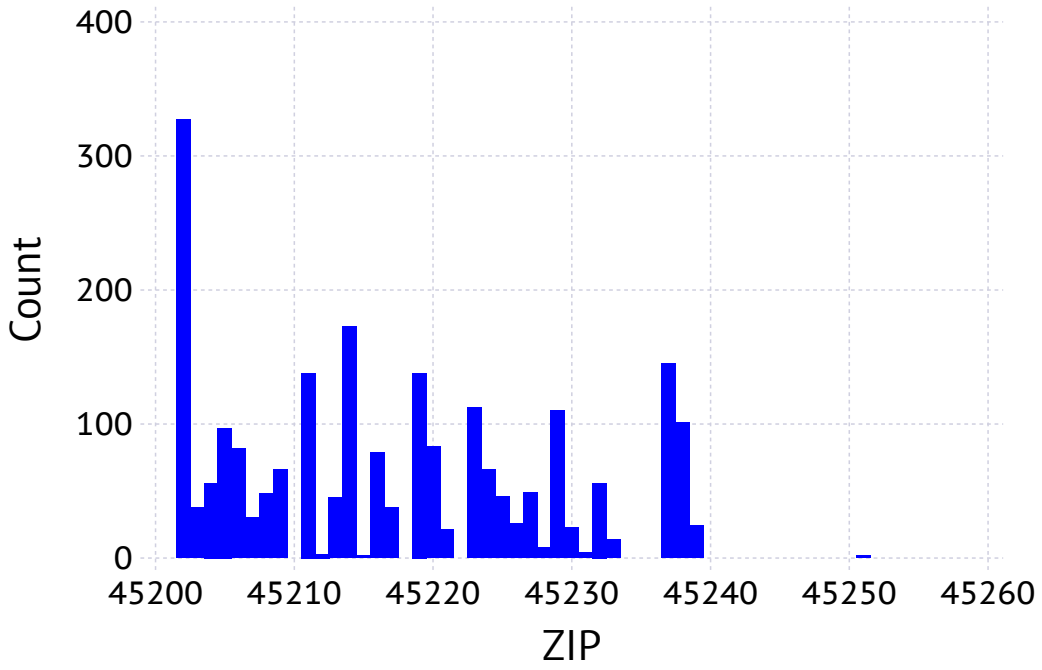
In [26]:
```
zip_accidents = combine(groupby(new_data_nomissing, :ZIP), nrow => "Count")
```

33×2 DataFrame *8 rows omitted*

Row	ZIP	Count
	Int64	Int64
1	45202	327
2	45203	38
3	45204	56
4	45205	97
5	45206	82
6	45207	30
7	45208	48
8	45209	66
9	45211	138
10	45212	3
11	45213	45
12	45214	173
13	45215	2
⋮	⋮	⋮
22	45226	26
23	45227	49
24	45228	8
25	45229	110
26	45230	23
27	45231	4
28	45232	56
29	45233	14
30	45237	145
31	45238	101
32	45239	24
33	45251	2

In [27]:
```
myplot = Gadfly.plot(zip_accidents,
            x = :ZIP,
            y = :Count,
            Geom.bar,
```

```
            white_panel);
draw(PNG("./figs/q12.png", 8inch, 8inch), myplot)
myplot
```

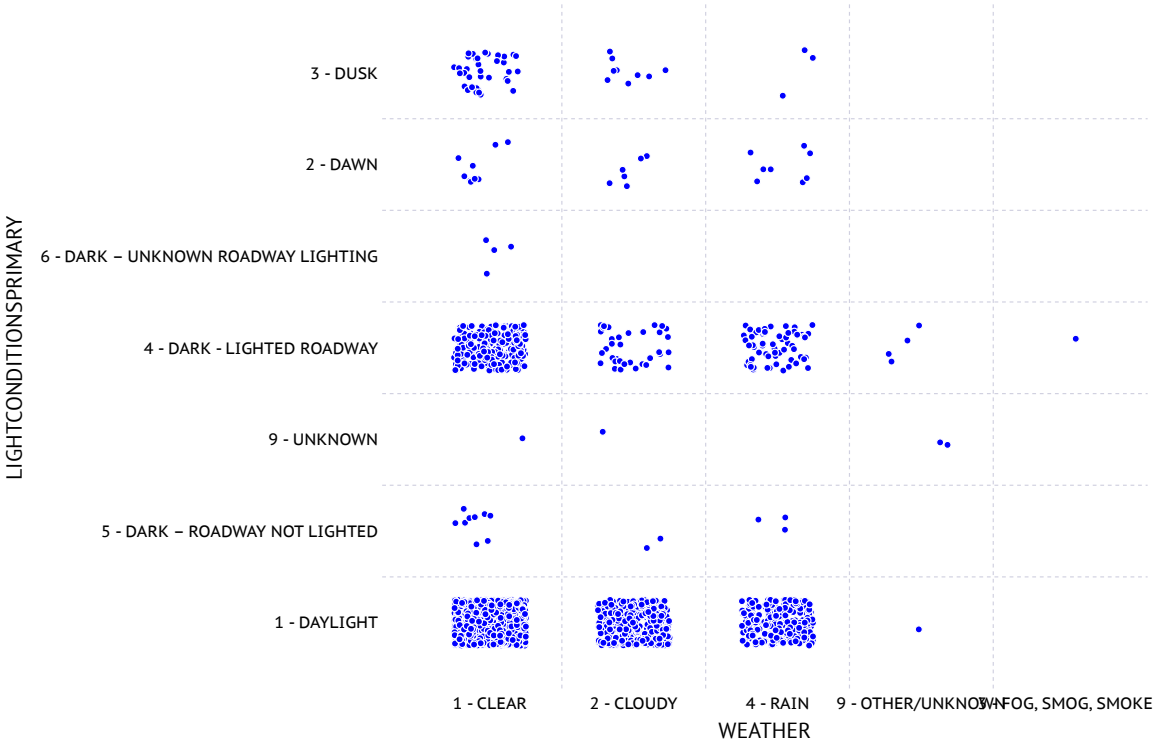


Q 13: Generate a scatter plot between weather and light conditions. State which combinations of weather and light conditions appear to have significantly higher number of crashes. Please use set_default_plot_size(12inch, 8inch) function to adjust the figure size as needed for visibility.

In [28]:
```
set_default_plot_size(12inch, 8inch)

myplot = Gadfly.plot(new_data_nomissing,
    x = "WEATHER",
    y = "LIGHTCONDITIONSPRIMARY",
    Stat.x_jitter(range=0.5, seed=1),
    Stat.y_jitter(range=0.5, seed=2),
    Geom.point,
    white_panel,
);

draw(PNG("./figs/q13.png", 12inch, 8inch), myplot)
myplot
```

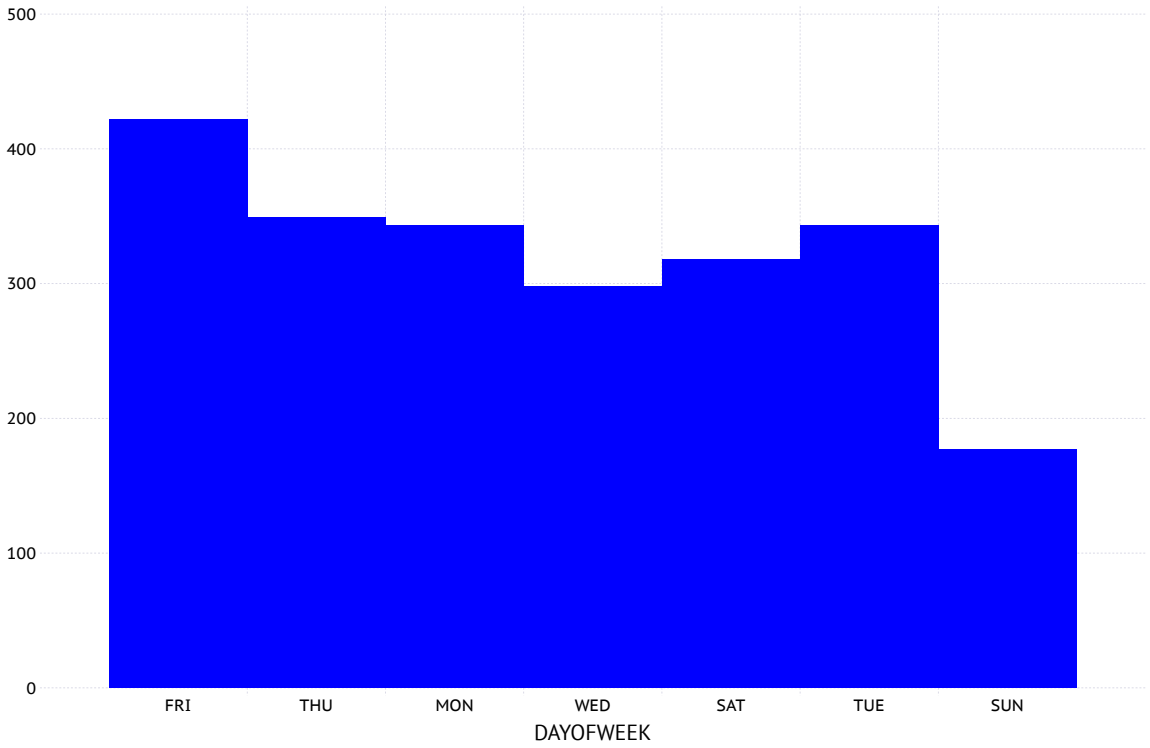


=> Combinations of weather and light conditions having higher number of crashes:

- Clear vs Daylight
- Cloudy vs Daylight
- Rain vs Daylight
- Clear vs Dark - Lighted Roadway

Q 14: Generate a plot to view the number of crashes on different days of the week. On which day of the week do fewer crashes happen? On which day of the week do the highest number of crashes happen?

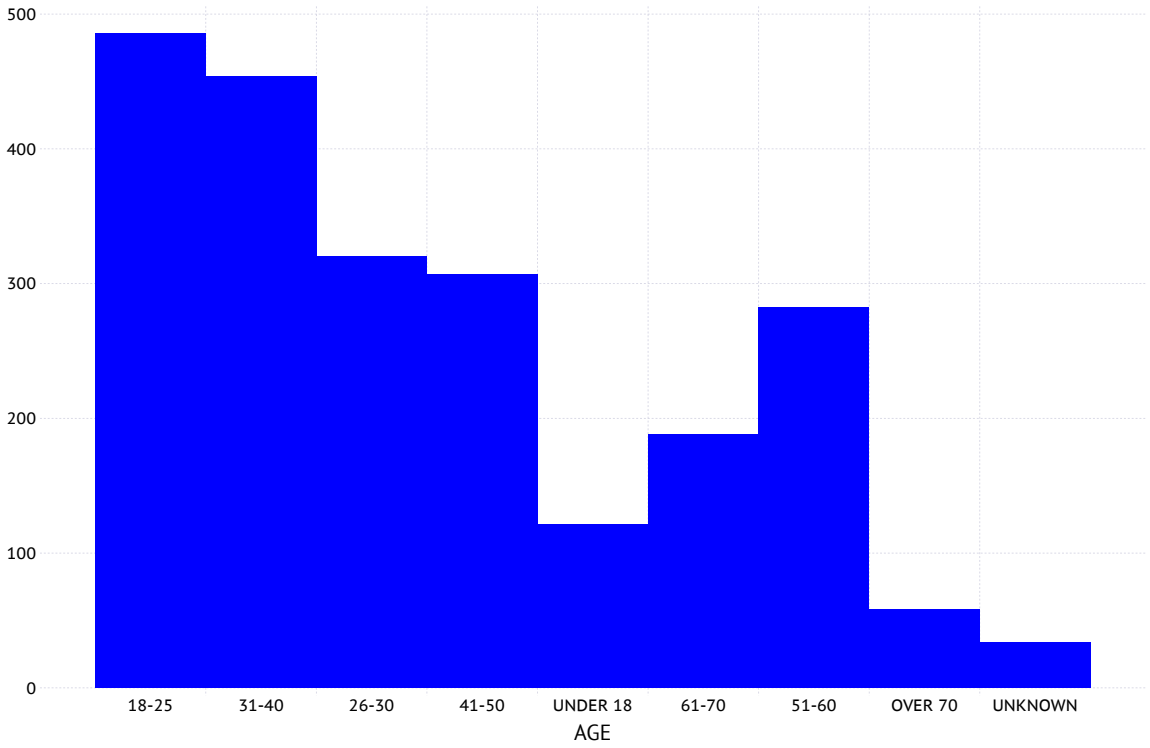
In [29]:
```
myplot = Gadfly.plot(new_data_nomissing,
    x = "DAYOFWEEK",
    Geom.histogram,
    white_panel,
);
draw(PNG("./figs/q14.png", 12inch, 8inch), myplot)
myplot
```



=> Sunday has fewest crashes, and Friday has highest crashes

Q 15: Generate a plot to view the number of crashes reported per age-group. Which age group is involved in the most number of crashes?

In [30]:
```
myplot = Gadfly.plot(new_data_nomissing,
    x = "AGE",
    Geom.histogram,
    white_panel,
);
draw(PNG("./figs/q15.png", 12inch, 8inch), myplot)
myplot
```



=> Age group from 18-25 is involved in the most number of crashes

Q 16: Load the "iris" dataset using the following command.

iris = dataset("datasets", "iris");

This dataset has information about flowers from three plant species.

Do:

1. List attributes in this data
2. Generate a scatter plot between "PetalLength" and "PetalWidth" where each point is colored based on "Sepecies". What observations can you make about the flowers from the three plant species based on this plot.

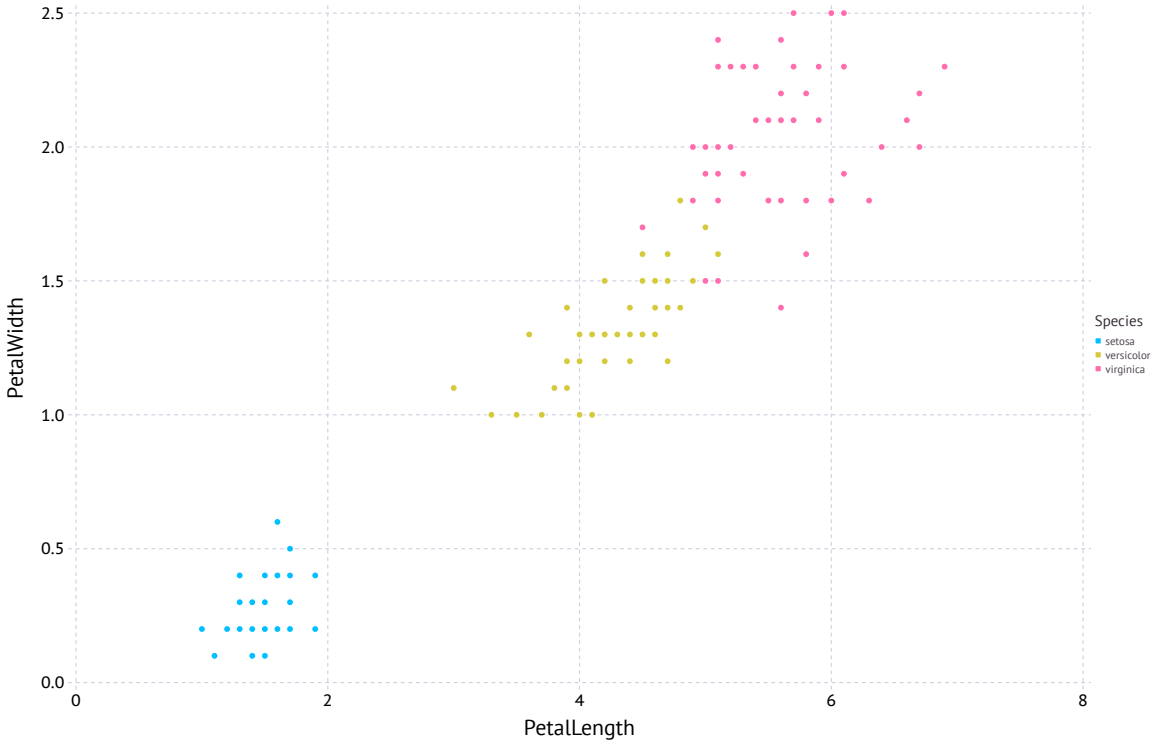
In [31]:
```
iris = dataset("datasets", "iris");
```

Attributes in this data:

In [32]:
```
names(iris)
```

```
5-element Vector{String}:
 "SepalLength"
 "SepalWidth"
 "PetalLength"
 "PetalWidth"
 "Species"
```

In [33]:
```
myplot = Gadfly.plot(iris,
    x = "PetalLength",
    y = "PetalWidth",
    color = "Species",
    Geom.point,
    white_panel,
);
draw(PNG("./figs/q16.png", 6inch, 6inch), myplot)
myplot
```



=> It's clear that setosa has the shortest petal length and width (bottom left cluster), while virginica has the longest (top right cluster). Versicolor lies in the middle range.

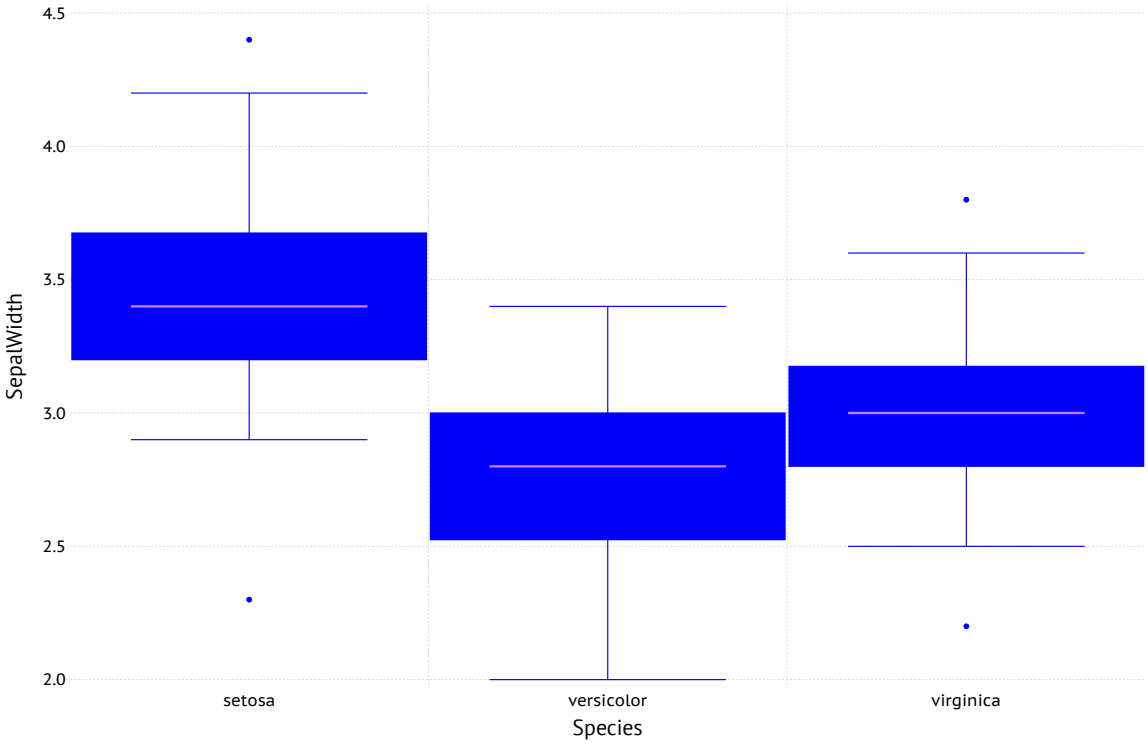
Also, from this plot, it's easy to classify setosa since its cluster is completely separated from the others.

Although there's a little overlap between versicolor and virginica, virginica has higher length and width in overall.

Q 17: Using Iris dataset, generate a box plot to compare the SepalWidth for the three plant species. Flowers from which species has generally longer sepalwidths?

In [34]:
```
myplot = Gadfly.plot(iris,
    x = "Species",
    y = "SepalWidth",
    Geom.boxplot,
    white_panel,
);
```

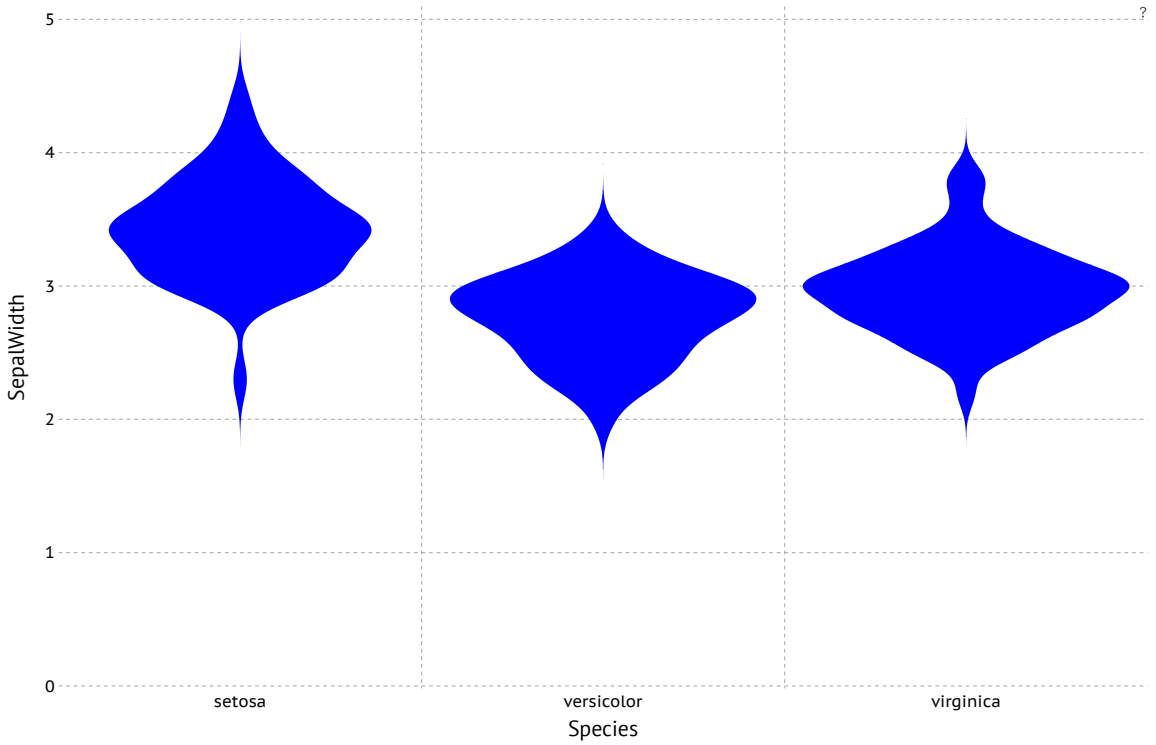
```
draw(PNG("./figs/q17.png", 6inch, 6inch), myplot)
myplot
```



=> Generally, setosa has longer sepal width

Q 18: Using Iris dataset, generate a violin plot for SepalWidth (similar to the box plot above). What new observations can you make from this plot, compared to the box plots you generated in response to **Q 17**.

In [35]:
```
myplot = Gadfly.plot(iris,
    x = "Species",
    y = "SepalWidth",
    Geom.violin,
    white_panel,
);
draw(PNG("./figs/q18.png", 6inch, 6inch), myplot)
myplot
```



=> Setosa has pretty wide distribution with highest peak (at ~3.4) compared to the other two.

Overall, versicolor distribution almost looks like virginica. However, virginica has a little higher peak (3.2 compared to 3.1), and the upper tail/quantile is wider (3.8), showing it has a little more variations at 3.8 width