Dataset

We will use Traffic Stops from Cincinnati City.

Data description: This dataset captures all subjects of traffic stops involving motor vehicles. Time of incident, officer assignment, race/sex of stop subject, and outcome of the stop ("Action taken") are also included in this data. Individual traffic stops may populate multiple data rows to account for multiple subjects and multiple outcomes: "incident number" is the unique identifier for every one (1) traffic stop.

Filename: "Traffic_Crash_Reports__CPD__Aug2018.csv"

Step 1: Write Julia code to load this data into memory.

In [1]:

```
Pkg.add("CSV")
using CSV
data = CSV.read("Traffic_Crash_Reports__CPD__Aug2018.csv", delim=",", missingstring
```

Out[1]:

	ADDRESS_X	LATITUDE_X	LONGITUDE_X	AGE	COMMUNITY_COUNCIL_NEIGHBORI
1	47XX READING RD	39.1721	84.4678	18-25	m
2	29XX MONTANA AV	39.1489	84.5963	31-40	m
3	39XX W LIBERTY ST	39.118	-84.578	26-30	WEST PRICE
4	29XX WASSON RD	39.1436	84.4347	41-50	m
5	S I75 AT 1-8 MM	39.1148	-84.5319	26-30	WES
6	56XX RIDGE AV	39.1751	-84.427	UNDER 18	PLEASANT F
7	26XX FIRTREE CT	39.1744	-84.5459	26-30	NORTH
8	50XX PADDOCK RD	39.1778	-84.4777	61-70	BONI
9	40XX HAMILTON AV	39.1583	-84.54	18-25	NORTH
10	29XX VERNON PL	39.1344	-84.4998	51-60	AVON
11	13XX W NORTH BEND RD	39.2017	-84.5415	UNKNOWN	COLLEGE
12	4XX E MARTIN LUTHER KING DR	39.1351	84.4994	51-60	m

	ADDRESS_X	LATITUDE_X	LONGITUDE_X	AGE	COMMUNITY_COUNCIL_NEIGHBORI
13	39XX BEEKMAN ST	39.1582	84.5495	51-60	m
14	S I75 AT 1-2 MM	39.1061	-84.5301	41-50	QUEENS
15	7XX W KING DR	39.1403	84.5316	18-25	m
16	6XX CLEMMER AV	39.1288	-84.5277	UNKNOWN	
17	1XX W MCMILLAN ST	39.1271	-84.5169	18-25	CUF - HEI
18	54XX BAHAMA TE	39.1899	84.568	UNKNOWN	m
19	N I471 AT 5.4	39.1035	84.4978	18-25	m
20	N I75 AT 2-5 MM	39.126	-84.5342	26-30	CAMP WASHIN
21	32XX GLENWAY AV	39.1125	84.563	31-40	m
22	S I71 AT 2-6 MM	39.119	-84.4993	18-25	WALNUT
23	36XX GLENWAY AV	39.1133	84.5722	41-50	m
24	3XX FOREST AV	39.146	-84.5001	51-60	AVON
25	13XX TENNESSEE AV	39.1679	84.4705	26-30	m
26	1XX W 3RD ST	39.0991	84.5145	61-70	m

	ADDRESS_X	LATITUDE_X	LONGITUDE_X	AGE	COMMUNITY_COUNCIL_NEIGHBORI
27	49XX GLENWAY AV	39.1209	84.601	31-40	m
28	1XX CRAFT ST	39.1784	-84.5115	31-40	WINTON
29	2XX GOODMAN AV	39.1366	-84.5032	31-40	CORRY
30	37XX GLENWAY AV	39.113	84.5746	UNKNOWN	m
⋮	⋮	⋮	⋮	⋮	

```
INFO: Pkg operations are not possible on JuliaBox. Please use the "Pa
ckages" menu at the top of the main screen.
```

Step 2: What is the size of the data?

In [2]:

```
using DataFrames;
size(data)
```

Out[2]:

```
(2567, 25)
```

Step 3: Create a new Dataframe by selecting the columns AGE, CRASHSEVERITYID, DAYOFWEEK, GENDER, INJURIES, LIGHTCONDITIONSPRIMARY, LOCALREPORTNO, MANNEROFCRASH, ROADSURFACE, WEATHER, ZIP

From here on wards work with the new Dataframe.

In [3]:

```
# New dataframe
data1 = data[:,[:AGE, :CRASHSEVERITYID, :DAYOFWEEK, :GENDER, :INJURIES, :LIGHTCONDI
head(data1)
```

Out[3]:

	AGE	CRASHSEVERITYID	DAYOFWEEK	GENDER	INJURIES	LIGHTCONDITIONSPRIMA
1	18-25	3	FRI	M - MALE	1 - NO INJURY / NONE REPORTED	1 - DAYLIG
2	31-40	2	THU	M - MALE	1 - NO INJURY / NONE REPORTED	1 - DAYLIG
3	26-30	2	FRI	M - MALE	2 - POSSIBLE	5 - DARK – ROADWAY N LIGHT
4	41-50	3	MON	M - MALE	1 - NO INJURY / NONE REPORTED	1 - DAYLIG
5	26-30	3	FRI	M - MALE	1 - NO INJURY / NONE REPORTED	1 - DAYLIG
6	UNDER 18	2	WED	F - FEMALE	1 - NO INJURY / NONE REPORTED	1 - DAYLIG

Step 4: Using showcols, list the different element types in the new data frame. Also list the columns in which there are missing values.

In [4]:

```
using DataFrames;
showcols(data1)
#TODO list of columns
```

Out[4]:

	variable	eltype	nmissing	first
1	AGE	CategoricalArrays.CategoricalString{UInt32}	0	18-25
2	CRASHSEVERITYID	Int64	0	3
3	DAYOFWEEK	CategoricalArrays.CategoricalString{UInt32}	0	FR
4	GENDER	CategoricalArrays.CategoricalString{UInt32}	299	M - MALE
5	INJURIES	CategoricalArrays.CategoricalString{UInt32}	10	1 - NO INJURY / NONE REPORTED
6	LIGHTCONDITIONSPRIMARY	CategoricalArrays.CategoricalString{UInt32}	0	1 - DAYLIGHT
7	LOCALREPORTNO	CategoricalArrays.CategoricalString{UInt32}	0	185011500
8	MANNEROFCRASH	CategoricalArrays.CategoricalString{UInt32}	0	2 - REAR-END
9	ROADSURFACE	CategoricalArrays.CategoricalString{UInt32}	0	1 - CONCRETE
10	WEATHER	CategoricalArrays.CategoricalString{UInt32}	0	1 - CLEAR
11	ZIP	Int64	18	45237

Step 5: Remove the rows in the missing values from the Dataframe. Comment on the number of rows that have been removed in this process.

In [5]:

```
# w_missing_data is dataframe cleaned dataframe without any missing values
w_missing_data = dropmissing(data1)
size(w_missing_data)
print(size(data1)[1]- size(w_missing_data)[1], " number of rows were removed")
```

Step 6: List the unique entries in the CRASHSEVERITY column

In [6]:

```
unique(data[:CRASHSEVERITY])
```

Out[6]:

```
3-element Array{Union{Missings.Missing, String},1}:
 "3 - PROPERTY DAMAGE ONLY (PDO)"
 "2 - INJURY"
 "1 - FATAL INJURY"

317 number of rows were removed
```

Step 7: Find out the different types of crashes in this data.

In [7]:

```
number_of_crashes = by(data, :CRASHSEVERITY, nrow)
```

Out[7]:

	CRASHSEVERITY	x1
1	3 - PROPERTY DAMAGE ONLY (PDO)	1916
2	2 - INJURY	646
3	1 - FATAL INJURY	5

Step 8: Find out the different types of WEATHERCONDITIONS in this data.

In [8]:

```
unique(w_missing_data[:WEATHER])
```

Out[8]:

```
5-element Array{Union{Missings.Missing, String},1}:
 "1 - CLEAR"
 "2 - CLOUDY"
 "4 - RAIN"
 "9 - OTHER/UNKNOWN"
 "3 - FOG, SMOG, SMOKE"
```

Step 9: Determine the number of crashes happened in each of these weather conditions using by() function.

In [9]:

```
number_of_crashes = by(w_missing_data, :WEATHER, nrow)
```

Out[9]:

	WEATHER	x1
1	1 - CLEAR	1519
2	2 - CLOUDY	402
3	4 - RAIN	321
4	9 - OTHER/UNKNOWN	7
5	3 - FOG, SMOG, SMOKE	1

Step 10: Find out the different light conditions in this data.

In [10]:

```
unique(data[:LIGHTCONDITIONSPRIMARY])
```

Out[10]:

```
8-element Array{Union{Missings.Missing, String},1}:
 "1 - DAYLIGHT"
 "5 - DARK – ROADWAY NOT LIGHTED"
 "9 - UNKNOWN"
 "4 - DARK - LIGHTED ROADWAY"
 "3 - DUSK"
 "6 - DARK – UNKNOWN ROADWAY LIGHTING"
 "8 - OTHER"
 "2 - DAWN"
```

Step 11: Determine the number of crashes happened in each of these weather conditions using by() function. What is the condition in which most crashes are reported?

In [11]:

```
number_of_crashes = by(w_missing_data, :LIGHTCONDITIONSPRIMARY, nrow)
```

Out[11]:

	LIGHTCONDITIONSPRIMARY	x1
1	1 - DAYLIGHT	1793
2	5 - DARK – ROADWAY NOT LIGHTED	14
3	9 - UNKNOWN	4
4	4 - DARK - LIGHTED ROADWAY	363
5	6 - DARK – UNKNOWN ROADWAY LIGHTING	4
6	2 - DAWN	23
7	3 - DUSK	49

In [12]:

```
print("Most crashes were reported in Daylight condition i.e. 1793 accidents")
```

In [13]:

```
number_of_crashes = by(w_missing_data, :WEATHER, nrow)
```

Out[13]:

	WEATHER	x1
1	1 - CLEAR	1519
2	2 - CLOUDY	402
3	4 - RAIN	321
4	9 - OTHER/UNKNOWN	7
5	3 - FOG, SMOG, SMOKE	1

In [14]:

```
print("Most crashes were reported in Clear condition i.e. 1519 accidents")
```

Step 12: Determine the number of crashes happened in each combination of weather and light conditions using by() function. State your observations.

In [15]:

```
a=unique(data[:WEATHER])
size(a)
b=unique(data[:LIGHTCONDITIONSPRIMARY])
size(b)
by(data, collect(Base.product(a ,b)), nrow)
```

```
ArgumentError: idx[1] has type Array{Tuple{Union{Missings.Missing, St
ring},Union{Missings.Missing, String}},2}; DataFrame only supports in
dexing columns with integers, symbols or boolean vectors

Stacktrace:
 [1] getindex(::DataFrames.Index, ::Array{Array{Tuple{Union{Missings.
Missing, String},Union{Missings.Missing, String}},2},1}) at /home/jru
n/.julia/v0.6/DataFrames/src/other/index.jl:172
 [2] getindex(::DataFrames.DataFrame, ::Array{Array{Tuple{Union{Missi
ngs.Missing, String},Union{Missings.Missing, String}},2},1}) at /hom
e/jrun/.julia/v0.6/DataFrames/src/dataframe/dataframe.jl:264
 [3] #groupby#89(::Bool, ::Bool, ::Function, ::DataFrames.DataFrame,
 ::Array{Array{Tuple{Union{Missings.Missing, String},Union{Missings.M
issing, String}},2},1}) at /home/jrun/.julia/v0.6/DataFrames/src/grou
peddataframe/grouping.jl:82
 [4] (::DataFrames.#kw##groupby)(::Array{Any,1}, ::DataFrames.#groupb
y, ::DataFrames.DataFrame, ::Array{Array{Tuple{Union{Missings.Missin
g, String},Union{Missings.Missing, String}},2},1}) at ./<missing>:0
 (repeats 2 times)
 [5] #by#101 at /home/jrun/.julia/v0.6/DataFrames/src/groupeddatafram
e/grouping.jl:304 [inlined]
 [6] by(::DataFrames.DataFrame, ::Array{Tuple{Union{Missings.Missing,
 String},Union{Missings.Missing, String}},2}, ::Function) at /home/jr
un/.julia/v0.6/DataFrames/src/groupeddataframe/grouping.jl:304
 [7] include_string(::String, ::String) at ./loading.jl:522


Most crashes were reported in Daylight condition i.e. 1793 accidentsM
ost crashes were reported in Clear condition i.e. 1519 accidents
```

Step 12: How many ZIP codes are covered in this data.

In [16]:

```
a = size(unique(w_missing_data[:ZIP]))
```

Out[16]:

```
(33,)
```

In [36]:

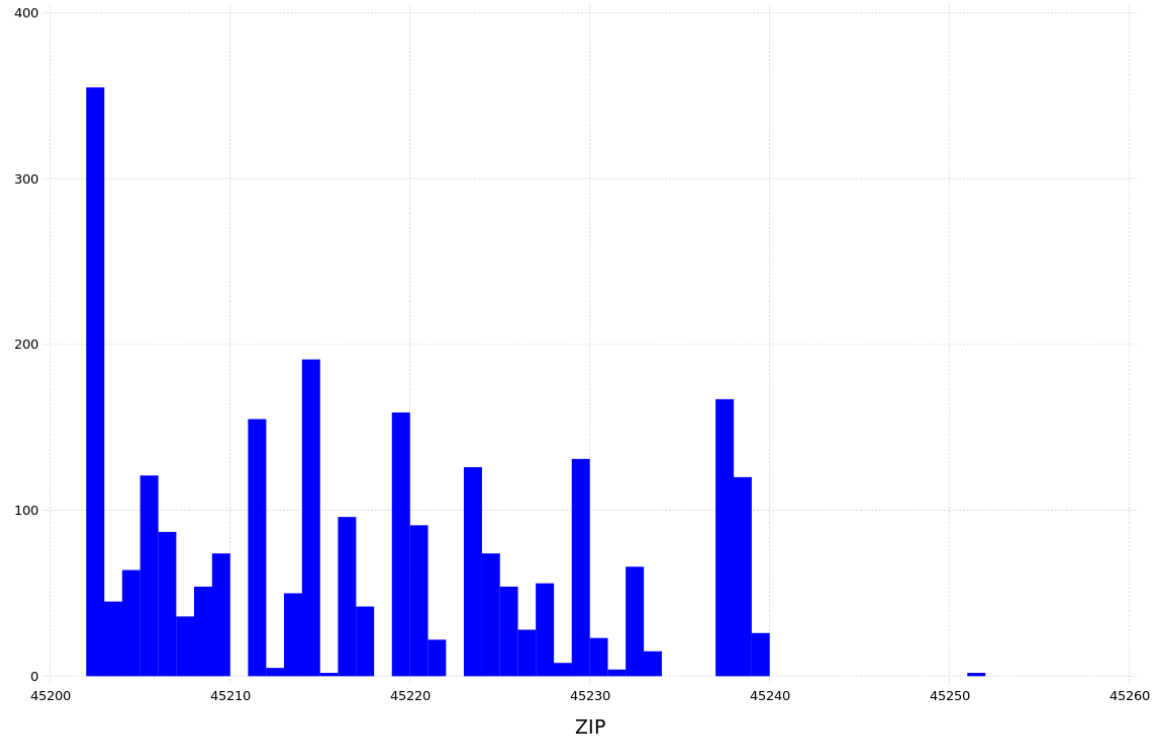
```
print("Observation: There are total 33 zip codes covered in this data.")
```

```
Observation: There are total 33 zip codes covered in this data.
```

Step 13: Plot a bar graph showing the number of accidents in each of the ZIP codes

In [25]:

```
Pkg.add("Gadfly");
Pkg.add("Cairo");

using Gadfly, Cairo;

white_panel = Theme(panel_fill = colorant"white",
                    default_color= colorant"blue",
                    major_label_font_size=14pt,
                    minor_label_font_size=10pt,
                    major_label_color=colorant"black",
                    minor_label_color=colorant"black");

# df = by(data1, :ZIP, nrows)
# myplot = plot(df, x="ZIP", y="x1", Geom.bar, color="ZIP", white_panel)

myplot = plot(data1, x="ZIP", Geom.histogram, white_panel)
draw(PNG(12inch, 8inch), myplot)
```



Out[25]:

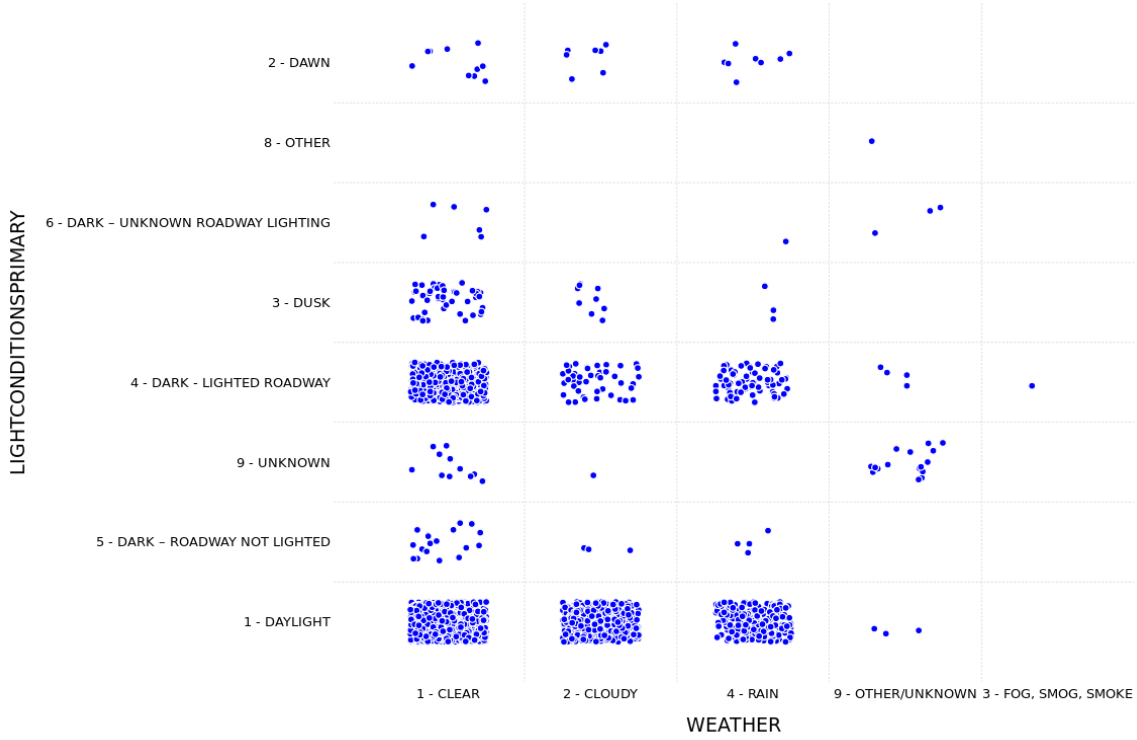
```
false
```

```
INFO: Pkg operations are not possible on JuliaBox. Please use the "Pa
ckages" menu at the top of the main screen.
INFO: Pkg operations are not possible on JuliaBox. Please use the "Pa
ckages" menu at the top of the main screen.
```

Step 14: Draw a scatter plot between weather and light conditions. State your observations. Please use set_default_plot_size(12inch, 8inch) function to adjust the figure size as needed for visibility.

In [37]:

```
myplot = Gadfly.plot(data, x=:WEATHER,           y=:LIGHTCONDITIONSPRIMARY,
                     Stat.x_jitter(range=0.5, seed=10),
                     Stat.y_jitter(range=0.5, seed=20),
                     Geom.point, white_panel);
draw(PNG(12inch, 8inch), myplot)
print("Observation: The categorical data shows that the most accidents occur during
```

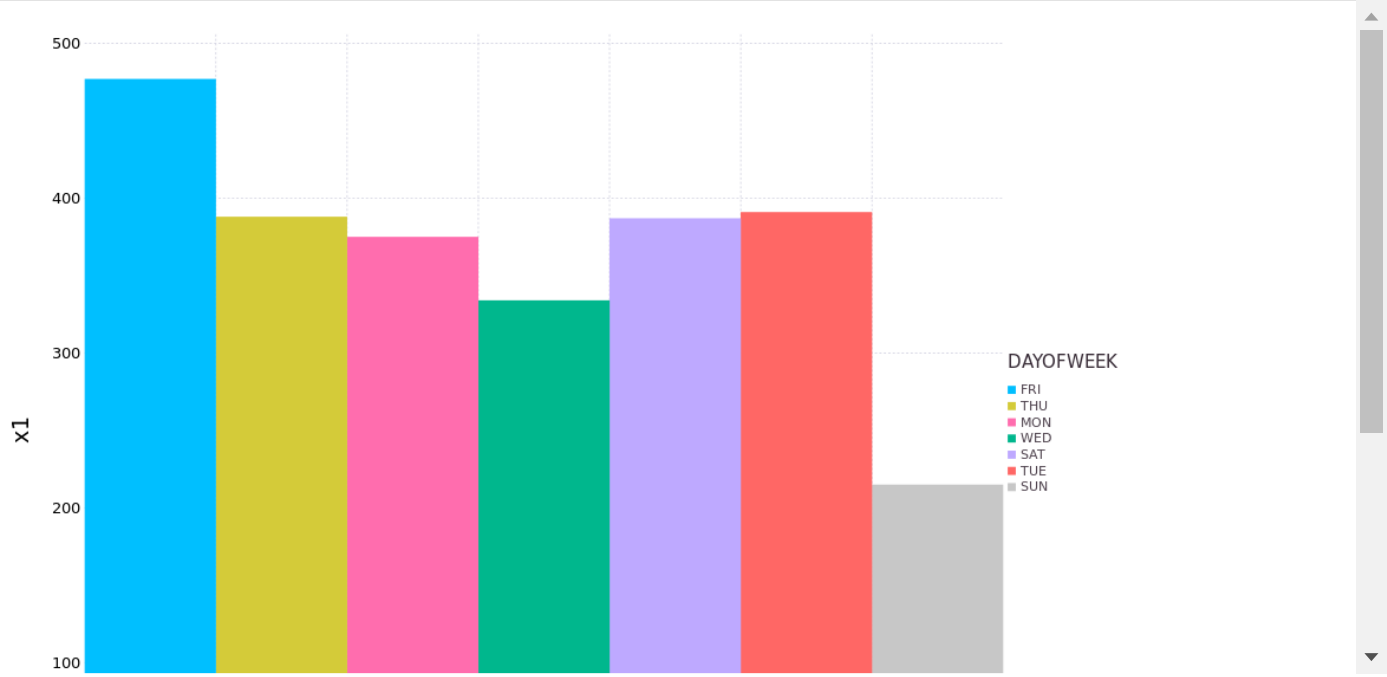



```
Observation: The categorical data shows that the most accidents occur
during Daylight and Dark light conditions. Furthermore, weather condi
tions like clear, rainy and cloudy are causes of most crashes.
```

Step 15: Make a plot to view the number of crashes on different days of the week. On which day of the week fewer crashes happen? On which day of the week more crashes happen?

In [38]:

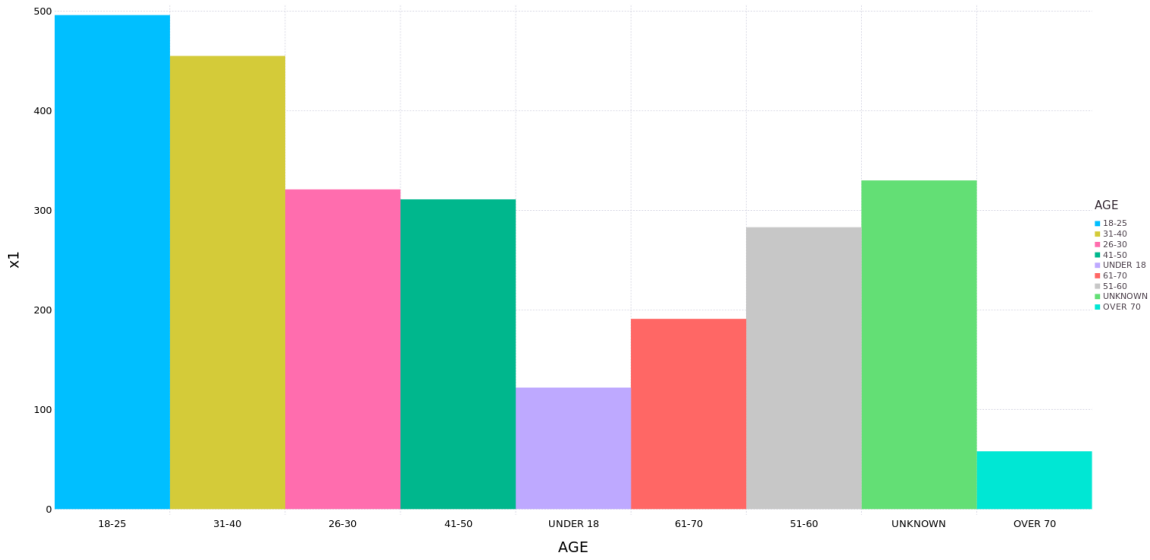
```
df = by(data1, :DAYOFWEEK, nrow)
myplot = plot(df, x="DAYOFWEEK", y ="x1", Geom.bar, color="DAYOFWEEK" ,white_panel
draw(PNG(10inch, 8inch), myplot)
print("Observation: Sunday is the day of the Week when fewst crashes happen and fri
```



Step 16: Make a plot to view the number of crashes reported per age-group. State your observations. State your observations.

In [41]:

```
="x1", Geom.bar, color="AGE" ,white_panel )

crashes occur in younger age group between 18-25 and 26-30. We can also infer that a
```




```
Observation: We can see that most crashes occur in younger age group
between 18-25 and 26-30. We can also infer that as people age, their
tendeny to crash decreses.
```

Step 17: Use the following two lines of code to load the "iris" dataset:

using RDatasets

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 [42]:

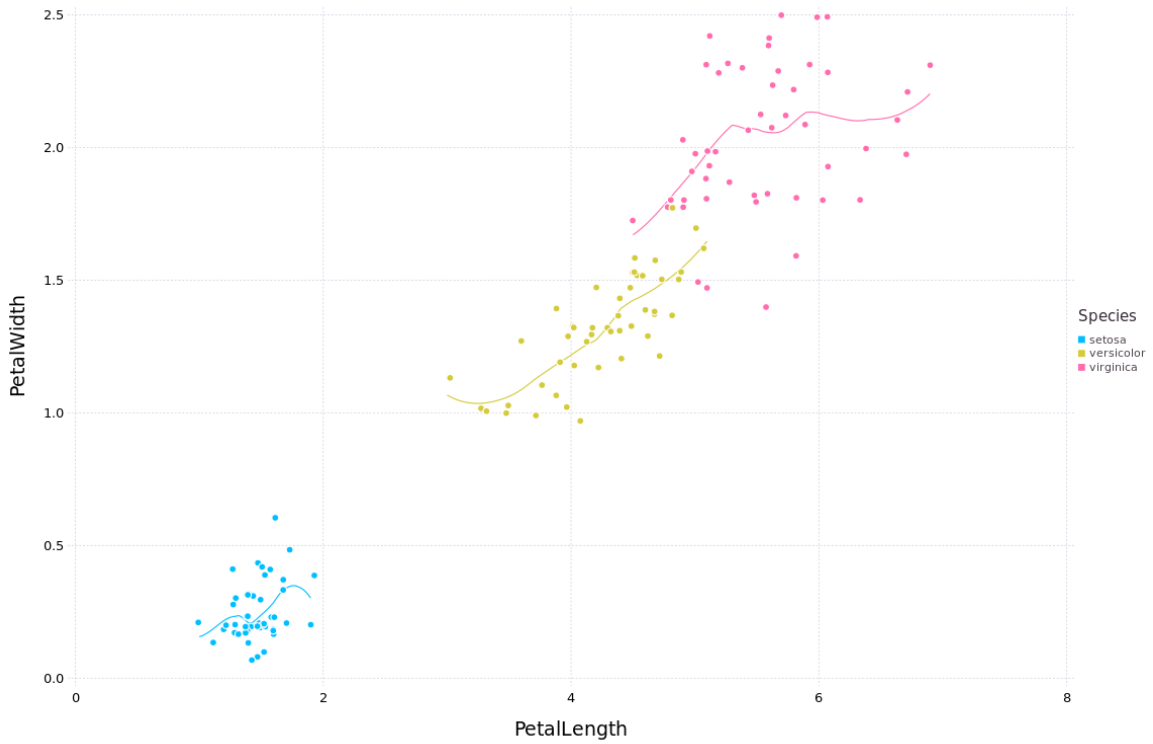
```
using RDatasets

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

Out[42]:

```
5-element Array{Symbol,1}:
 :SepalLength
 :SepalWidth
 :PetalLength
 :PetalWidth
 :Species
```

In [44]:

```
myplot = plot(iris, x=:PetalLength, y=:PetalWidth,
              Stat.x_jitter(range=0.7, seed=10),
              Stat.y_jitter(range=0.7, seed=20),
              Geom.point,Geom.smooth, color="Species", white_panel)
draw(PNG(12inch, 8inch), myplot)
print("Observations: Each species can be classified by their petalwidth and petalle
```

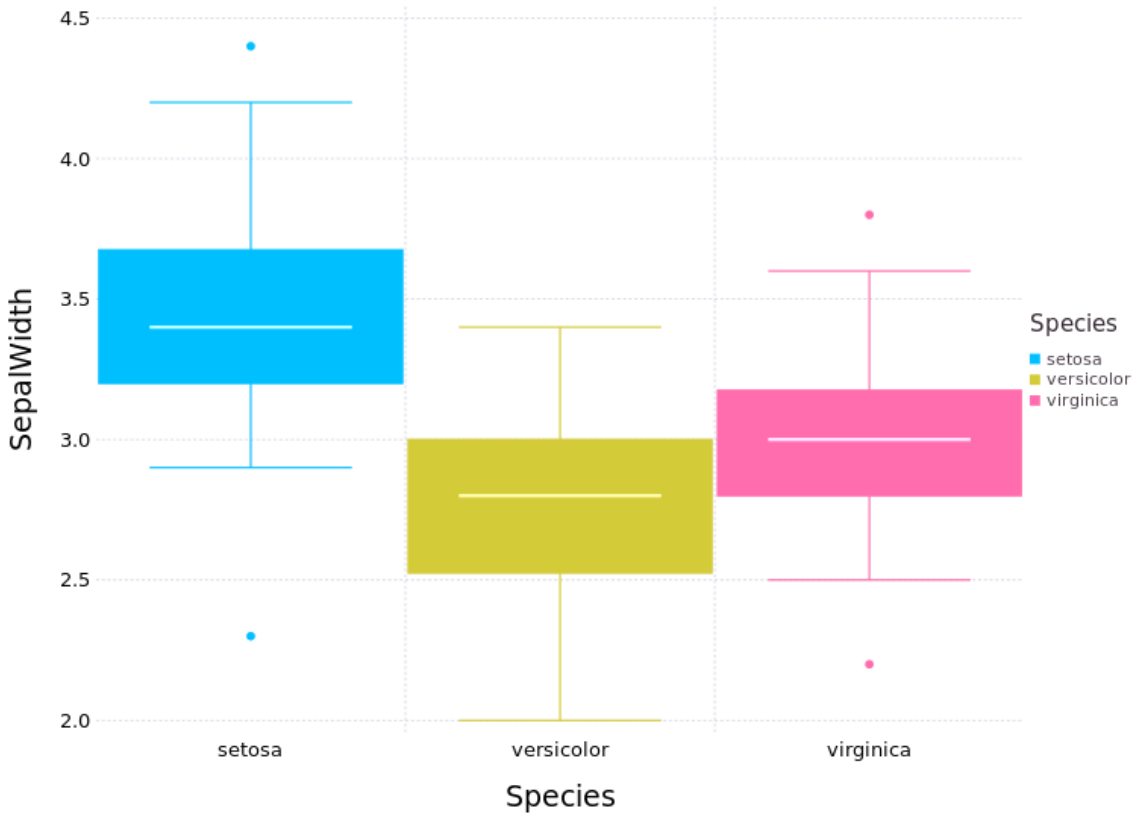



```
Observations: Each species can be classified by their petalwidth and
petallength, i.e. viginica has petal width and petal length more than
1.5 and 4 respectively. This gives us a basic feature to classify dif
ferent species based on their petallength and petalwidth.
```

Step 18: Using IRIS dataset draw a box plot to compare the SepalWidth for the three plant species. What observations can you make based on this plot?

In [45]:

```
myplot = plot(iris, x="Species", y="SepalWidth",Geom.boxplot,color="Species", white
draw(PNG(8inch, 6inch), myplot)
print("Observation: We can see that max sepalwidth of versicolor and virginica is a
```

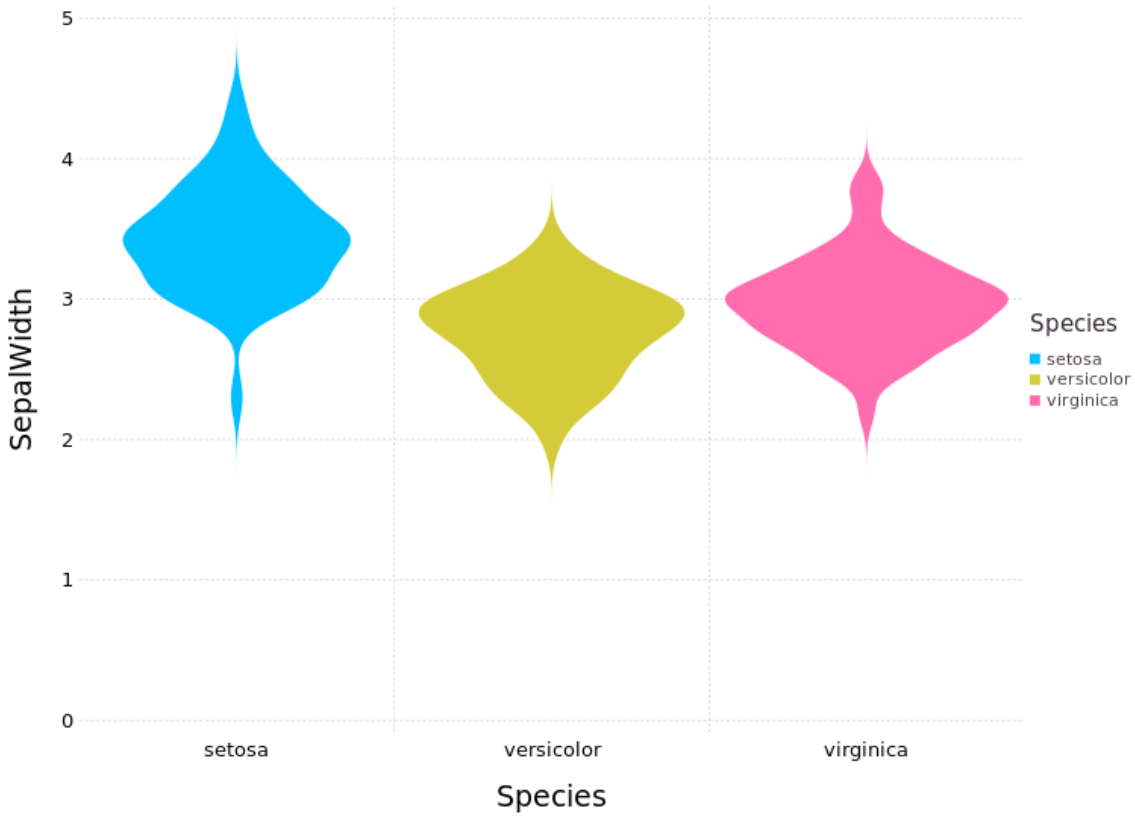



```
Observation: We can see that max sepalwidth of versicolor and virgini
ca is almost equal to mean of setosa.
```

Step 19: Draw a violin plot for SepalWidth (similar to the box plot above) and state any new observations you may have.

In [46]:

```
myplot = plot(iris, x="Species", y="SepalWidth",Geom.violin, color="Species" ,white
draw(PNG(8inch, 6inch), myplot)
print("Observaation: mean density of versicolar < mean density of virginica < mean
```




```
Observaation: mean density of versicolar < mean density of virginica
< mean density of setosa
```

In [ ]:

```

```