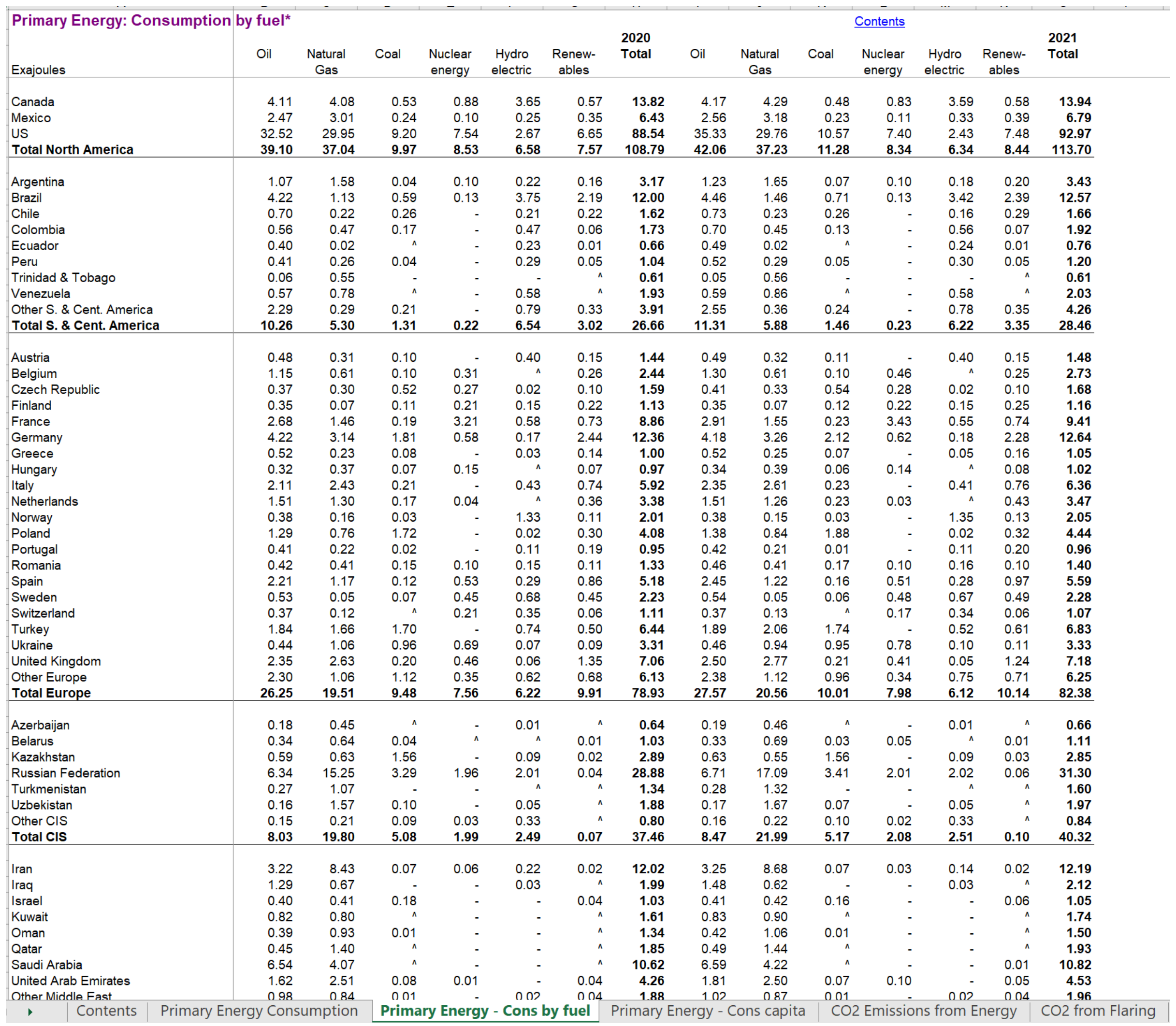
Data Exercise, Week 1

This week, our **Data Exercise** focuses on an interesting source of data, the Statistical Review of World Energy, now published by the bp-affiliated Energy Institute. We'll use this to learn a couple of new commands in R. The idea is to ramp you on-board with R slowly but surely, and this exercise is not graded. Later this month, you will have a graded exercise that uses some of these skills. For today, let's make a graph of primary energy consumption by source.

The reason I wanted you to use this data this week is so that you can set yourself up in parallel in Excel and R so that you can visualize what's happening in R behind the scenes and so that you can also get used to using spreadsheet-based data in R. You can do this exercise all in scripts, but I'd encourage you to build an RMarkdown document so that you get used to using them as you'll need them for assignments later on. If you want to do so, then you can adapt all of these *code chunks* into your own R-Markdown (RMD) file, and you can use `week2.Rmd` as your starting point. Knit it before you start playing with it, just so you see that it works, then make a clean version to use for this week's exercise.

There is also a lot of data cleaning to do, which will help you get your head around how data are stored in R.

First, as you likely normally would, download the XLSX file for the Statistical Review. Open it in Excel or Sheets, whichever you normally use. For this exercise, we're going to use the third tab, for Total Primary Energy Consumption By Fuel.



Primary Energy: Consumption by fuel*

Contents

Exajoules	Oil	Natural Gas	Coal	Nuclear energy	Hydro electric	Renew-ables	2020 Total	Oil	Natural Gas	Coal	Nuclear energy	Hydro electric	Renew-ables	2021 Total
Canada	4.11	4.08	0.53	0.88	3.65	0.57	**13.82**	4.17	4.29	0.48	0.83	3.59	0.58	**13.94**
Mexico	2.47	3.01	0.24	0.10	0.25	0.35	**6.43**	2.56	3.18	0.23	0.11	0.33	0.39	**6.79**
US	32.52	29.95	9.20	7.54	2.67	6.65	**88.54**	35.33	29.76	10.57	7.40	2.43	7.48	**92.97**
Total North America	**39.10**	**37.04**	**9.97**	**8.53**	**6.58**	**7.57**	**108.79**	**42.06**	**37.23**	**11.28**	**8.34**	**6.34**	**8.44**	**113.70**
Argentina	1.07	1.58	0.04	0.10	0.22	0.16	**3.17**	1.23	1.65	0.07	0.10	0.18	0.20	**3.43**
Brazil	4.22	1.13	0.59	0.13	3.75	2.19	**12.00**	4.46	1.46	0.71	0.13	3.42	2.39	**12.57**
Chile	0.70	0.22	0.26	-	0.21	0.22	**1.62**	0.73	0.23	0.26	-	0.16	0.29	**1.66**
Colombia	0.56	0.47	0.17	-	0.47	0.06	**1.73**	0.70	0.45	0.13	-	0.56	0.07	**1.92**
Ecuador	0.40	0.02	^	-	0.23	0.01	**0.66**	0.49	0.02	^	-	0.24	0.01	**0.76**
Peru	0.41	0.26	0.04	-	0.29	0.05	**1.04**	0.52	0.29	0.05	-	0.30	0.05	**1.20**
Trinidad & Tobago	0.06	0.55	-	-	-	^	**0.61**	0.05	0.56	-	-	-	^	**0.61**
Venezuela	0.57	0.78	^	-	0.58	^	**1.93**	0.59	0.86	^	-	0.58	^	**2.03**
Other S. & Cent. America	2.29	0.29	0.21	-	0.79	0.33	**3.91**	2.55	0.36	0.24	-	0.78	0.35	**4.26**
Total S. & Cent. America	**10.26**	**5.30**	**1.31**	**0.22**	**6.54**	**3.02**	**26.66**	**11.31**	**5.88**	**1.46**	**0.23**	**6.22**	**3.35**	**28.46**
Austria	0.48	0.31	0.10	-	0.40	0.15	**1.44**	0.49	0.32	0.11	-	0.40	0.15	**1.48**
Belgium	1.15	0.61	0.10	0.31	^	0.26	**2.44**	1.30	0.61	0.10	0.46	^	0.25	**2.73**
Czech Republic	0.37	0.30	0.52	0.27	0.02	0.10	**1.59**	0.41	0.33	0.54	0.28	0.02	0.10	**1.68**
Finland	0.35	0.07	0.11	0.21	0.15	0.22	**1.13**	0.35	0.07	0.12	0.22	0.15	0.25	**1.16**
France	2.68	1.46	0.19	3.21	0.58	0.73	**8.86**	2.91	1.55	0.23	3.43	0.55	0.74	**9.41**
Germany	4.22	3.14	1.81	0.58	0.17	2.44	**12.36**	4.18	3.26	2.12	0.62	0.18	2.28	**12.64**
Greece	0.52	0.23	0.08	-	0.03	0.14	**1.00**	0.52	0.25	0.07	-	0.05	0.16	**1.05**
Hungary	0.32	0.37	0.07	0.15	^	0.07	**0.97**	0.34	0.39	0.06	0.14	^	0.08	**1.02**
Italy	2.11	2.43	0.21	-	0.43	0.74	**5.92**	2.35	2.61	0.23	-	0.41	0.76	**6.36**
Netherlands	1.51	1.30	0.17	0.04	^	0.36	**3.38**	1.51	1.26	0.23	0.03	^	0.43	**3.47**
Norway	0.38	0.16	0.03	-	1.33	0.11	**2.01**	0.38	0.15	0.03	-	1.35	0.13	**2.05**
Poland	1.29	0.76	1.72	-	0.02	0.30	**4.08**	1.38	0.84	1.88	-	0.02	0.32	**4.44**
Portugal	0.41	0.22	0.02	-	0.11	0.19	**0.95**	0.42	0.21	0.01	-	0.11	0.20	**0.96**
Romania	0.42	0.41	0.15	0.10	0.15	0.11	**1.33**	0.46	0.41	0.17	0.10	0.16	0.10	**1.40**
Spain	2.21	1.17	0.12	0.53	0.29	0.86	**5.18**	2.45	1.22	0.16	0.51	0.28	0.97	**5.59**
Sweden	0.53	0.05	0.07	0.45	0.68	0.45	**2.23**	0.54	0.05	0.06	0.48	0.67	0.49	**2.28**
Switzerland	0.37	0.12	^	0.21	0.35	0.06	**1.11**	0.37	0.13	^	0.17	0.34	0.06	**1.07**
Turkey	1.84	1.66	1.70	-	0.74	0.50	**6.44**	1.89	2.06	1.74	-	0.52	0.61	**6.83**
Ukraine	0.44	1.06	0.96	0.69	0.07	0.09	**3.31**	0.46	0.94	0.95	0.78	0.10	0.11	**3.33**
United Kingdom	2.35	2.63	0.20	0.46	0.06	1.35	**7.06**	2.50	2.77	0.21	0.41	0.05	1.24	**7.18**
Other Europe	2.30	1.06	1.12	0.35	0.62	0.68	**6.13**	2.38	1.12	0.96	0.34	0.75	0.71	**6.25**
Total Europe	**26.25**	**19.51**	**9.48**	**7.56**	**6.22**	**9.91**	**78.93**	**27.57**	**20.56**	**10.01**	**7.98**	**6.12**	**10.14**	**82.38**
Azerbaijan	0.18	0.45	^	-	0.01	^	**0.64**	0.19	0.46	^	-	0.01	^	**0.66**
Belarus	0.34	0.64	0.04	^	^	0.01	**1.03**	0.33	0.69	0.03	0.05	^	0.01	**1.11**
Kazakhstan	0.59	0.63	1.56	-	0.09	0.02	**2.89**	0.63	0.55	1.56	-	0.09	0.03	**2.85**
Russian Federation	6.34	15.25	3.29	1.96	2.01	0.04	**28.88**	6.71	17.09	3.41	2.01	2.02	0.06	**31.30**
Turkmenistan	0.27	1.07	-	-	^	^	**1.34**	0.28	1.32	-	-	^	^	**1.60**
Uzbekistan	0.16	1.57	0.10	-	0.05	^	**1.88**	0.17	1.67	0.07	-	0.05	^	**1.97**
Other CIS	0.15	0.21	0.09	0.03	0.33	^	**0.80**	0.16	0.22	0.10	0.02	0.33	^	**0.84**
Total CIS	**8.03**	**19.80**	**5.08**	**1.99**	**2.49**	**0.07**	**37.46**	**8.47**	**21.99**	**5.17**	**2.08**	**2.51**	**0.10**	**40.32**
Iran	3.22	8.43	0.07	0.06	0.22	0.02	**12.02**	3.25	8.68	0.07	0.03	0.14	0.02	**12.19**
Iraq	1.29	0.67	-	-	0.03	^	**1.99**	1.48	0.62	-	-	0.03	^	**2.12**
Israel	0.40	0.41	0.18	-	-	0.04	**1.03**	0.41	0.42	0.16	-	-	0.06	**1.05**
Kuwait	0.82	0.80	^	-	-	^	**1.61**	0.83	0.90	^	-	-	^	**1.74**
Oman	0.39	0.93	0.01	-	-	^	**1.34**	0.42	1.06	0.01	-	-	^	**1.50**
Qatar	0.45	1.40	^	-	-	^	**1.85**	0.49	1.44	^	-	-	^	**1.93**
Saudi Arabia	6.54	4.07	^	-	-	^	**10.62**	6.59	4.22	^	-	-	0.01	**10.82**
United Arab Emirates	1.62	2.51	0.08	0.01	-	0.04	**4.26**	1.81	2.50	0.07	0.10	-	0.05	**4.53**
Other Middle East	0.98	0.84	0.01	-	0.02	0.04	**1.88**	1.02	0.87	0.01	-	0.02	0.04	**1.96**

Contents
Primary Energy Consumption
Primary Energy - Cons by fuel
Primary Energy - Cons capita
CO2 Emissions from Energy
CO2 from Flaring

We're going to learn about a few things here: reading from Excel using `library(readxl)` (make sure you have the library installed), more filtering skills, and making stacked bar charts. I'm going to use some `kableExtra` tables in this document just to make things a bit easier for you to see, but you don't need to use those parts to make the graph at the end.

As usual, we'll start with loading the packages we need. Create a code chunk in your document that looks like this:

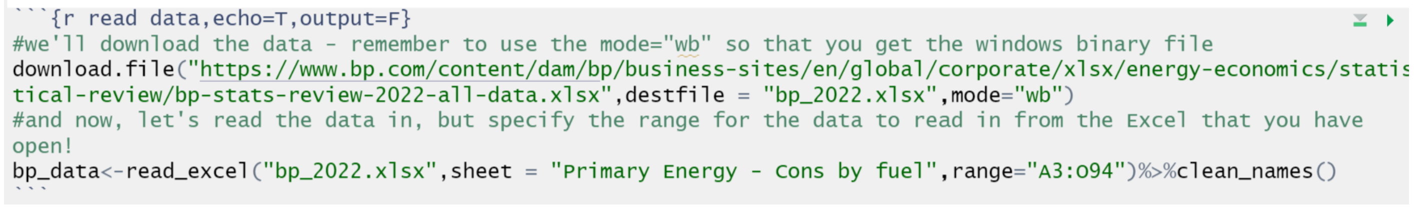
```
library(tidyverse)
library(kableExtra)
library(readxl)
library(janitor)
```

Let's download the data. Create a code chunk to do this and then to read it into R.

```
#check to see if the file exists. The first line after the if will only run if it doesn't
#you don't need this line, but it's a nice efficiency improvement for your code and one that I should use more often
if(!file.exists("bp_2023.xlsx"))
#we'll download the data giving it a local file name that's easy to deal with - remember to use the mode="wb" so that you get the
    download.file("https://www.energyinst.org/__data/assets/excel_doc/0007/1055545/EI-stats-review-all-data.xlsx",destfile = "bp_
#and now, let's read the data in, but specify the range for the data to read in from the Excel that you have open!
bp_data<-read_excel("bp_2023.xlsx",sheet = "Primary Energy - Cons by fuel",range="A3:O94")%>%clean_names()
```

Now's the first lesson of the day: how to execute code chunks while working on your markdown document.

These parts are *code chunks* and you can execute them individually (you should start from the top and do them in sequence to make sure packages are loaded - by clicking that green triangle). You can also basically *start over* by clicking the grey triangle with the green bar, which will run all your code chunks from the start of your document. This is the equivalent to running the commands in sequence in a script.

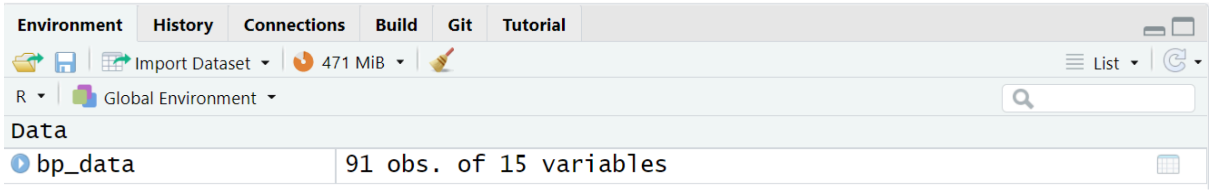


~~~
```{r read data,echo=T,output=F}
#we'll download the data - remember to use the mode="wb" so that you get the windows binary file
download.file("https://www.bp.com/content/dam/bp/business-sites/en/global/corporate/xlsx/energy-economics/statis
tical-review/bp-stats-review-2022-all-data.xlsx",destfile = "bp_2022.xlsx",mode="wb")
#and now, let's read the data in, but specify the range for the data to read in from the Excel that you have
open!
bp_data<-read_excel("bp_2022.xlsx",sheet = "Primary Energy - Cons by fuel",range="A3:O94")%>%clean_names()
```
~~~

Code Chunks

Important note: running a code chunk is like running an R script or executing commands from the console. When you eventually go to *knit* your RMarkdown document, it is basically running in a new, clean environment, so it will re-do all the code chunks in sequence starting from scratch.

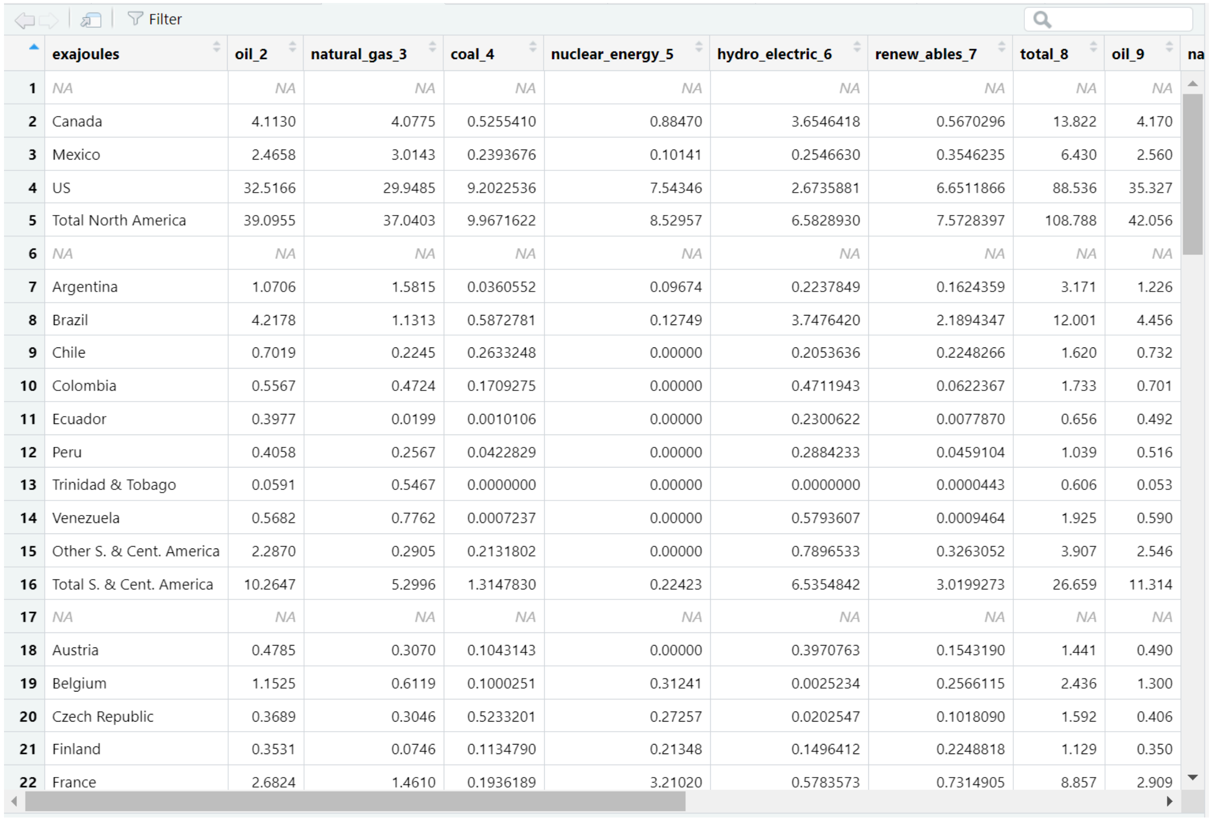
So, try it - copy and add the first two code chunks in this document to a new document of your own (or use the template) and, once you run then, you'll see that you have the `bp_data` in memory. You can see that in the top right corner of your RStudio screen.



```
Data
bp_data    91 obs. of 15 variables
```

Code Chunks

Click on it and you'll open a viewer window and you'll be able to see how R read in the data. It will look something like this:



	exajoules	oil_2	natural_gas_3	coal_4	nuclear_energy_5	hydro_electric_6	renew_ables_7	total_8	oil_9	na
1	NA	NA	NA	NA	NA	NA	NA	NA	NA	
2	Canada	4.1130	4.0775	0.5255410	0.88470	3.6546418	0.5670296	13.822	4.170	
3	Mexico	2.4658	3.0143	0.2393676	0.10141	0.2546630	0.3546235	6.430	2.560	
4	US	32.5166	29.9485	9.2022536	7.54346	2.6735881	6.6511866	88.536	35.327	
5	Total North America	39.0955	37.0403	9.9671622	8.52957	6.5828930	7.5728397	108.788	42.056	
6	NA	NA	NA	NA	NA	NA	NA	NA	NA	
7	Argentina	1.0706	1.5815	0.0360552	0.09674	0.2237849	0.1624359	3.171	1.226	
8	Brazil	4.2178	1.1313	0.5872781	0.12749	3.7476420	2.1894347	12.001	4.456	
9	Chile	0.7019	0.2245	0.2633248	0.00000	0.2053636	0.2248266	1.620	0.732	
10	Colombia	0.5567	0.4724	0.1709275	0.00000	0.4711943	0.0622367	1.733	0.701	
11	Ecuador	0.3977	0.0199	0.0010106	0.00000	0.2300622	0.0077870	0.656	0.492	
12	Peru	0.4058	0.2567	0.0422829	0.00000	0.2884233	0.0459104	1.039	0.516	
13	Trinidad & Tobago	0.0591	0.5467	0.0000000	0.00000	0.0000000	0.0000443	0.606	0.053	
14	Venezuela	0.5682	0.7762	0.0007237	0.00000	0.5793607	0.0009464	1.925	0.590	
15	Other S. & Cent. America	2.2870	0.2905	0.2131802	0.00000	0.7896533	0.3263052	3.907	2.546	
16	Total S. & Cent. America	10.2647	5.2996	1.3147830	0.22423	6.5354842	3.0199273	26.659	11.314	
17	NA	NA	NA	NA	NA	NA	NA	NA	NA	
18	Austria	0.4785	0.3070	0.1043143	0.00000	0.3970763	0.1543190	1.441	0.490	
19	Belgium	1.1525	0.6119	0.1000251	0.31241	0.0025234	0.2566115	2.436	1.300	
20	Czech Republic	0.3689	0.3046	0.5233201	0.27257	0.0202547	0.1018090	1.592	0.406	
21	Finland	0.3531	0.0746	0.1134790	0.21348	0.1496412	0.2248818	1.129	0.350	
22	France	2.6824	1.4610	0.1936189	3.21020	0.5783573	0.7314905	8.857	2.909	

Code Chunks

It's ugly. But, you know what you're dealing with. Let's clean it up. In this case, it will be much easier for you to do while looking at the Excel as well.

First thing we know is that, for 2023 data, we only want columns 1 and 9-15 of the worksheet, so let's do that using the `select` command.

```
#select is for picking columns. You can use it with numbers or names
#right now, our names aren't clear, so let's use numbers
bp_data<-bp_data %>% select(1,seq(9,15))
```

That gives us a much cleaner data set, but the names are all messed up. So, let's fix them.

```
#assign new column names
names(bp_data)<-c("country","Oil","Natural Gas","Coal","Nuclear","Hydro","Renewables","Total")
#make sure they're "clean"
bp_data<-bp_data%>%clean_names()
```

So, now we have a much nicer data set, but there's still the problem of all the blank rows. So, let's clean that up too, using `filter`:

```
#remove observations that have an NA for country (ie. choose the ones that don't)
#this is going to annoy you, but ! is like "not" in this context.
bp_data<-bp_data%>%filter(!is.na(country))
```

If you mess code up at some point, you can always go back and re-read the data and start over, executing each of the chunks on the way back to here to make sure it works.

Now, I'm going to use a little bit of a fancier table maker to show you the data I've got after running all of this, using the `kableExtra` package:

```
  bp_data %>%
    kbl(escape = FALSE) %>%
    kable_styling(fixed_thead = T,bootstrap_options = c("hover", "condensed","responsive"))%>%
  scroll_box(width = "1000px", height = "400px")%>%
#adding this I() at the end of a chain of commands is like a placeholder in case you would otherwise end with a %>%.
# It makes it easier to paste in code from elsewhere
      I()
```

country	oil	natural_gas	coal	nuclear	hydro	renewables	total
Canada	4.267	4.379	0.386	0.780	3.740	0.591	14.143
Mexico	4.118	3.477	0.251	0.098	0.335	0.446	8.725

country	oil	natural_gas	coal	nuclear	hydro	renewables	total
US	36.150	31.724	9.868	7.315	2.427	8.427	95.910
Total North America	44.535	39.579	10.506	8.193	6.502	9.464	118.778
Argentina	1.378	1.645	0.052	0.067	0.224	0.238	3.604
Brazil	5.007	1.151	0.586	0.131	4.009	2.526	13.410
Chile	0.797	0.269	0.220	0.000	0.209	0.294	1.789
Colombia	0.961	0.453	0.103	0.000	0.604	0.073	2.194
Ecuador	0.527	0.020	0.003	0.000	0.231	0.006	0.787
Peru	0.497	0.346	0.029	0.000	0.279	0.054	1.206
Trinidad & Tobago	0.049	0.542	0.000	0.000	0.000	0.000	0.591

I still don't have exactly what I want, because I still have a lot of lines that don't contain data I want to graph - I just want to use the totals, so I'll show you a couple of tricks to clean those out.

```r
#keep only observations that have total in the country cell using the grepl command
#grepl is a logical search function, so it will keep any observation (row) for which the term Total (case sensitive) appears in t
bp_data<-bp_data%>%filter(grepl("Total",country))
```

`grepl` is a really handy command that returns true for any cell that contains the word "Total" and false otherwise. It will be case sensitive, so check to make sure it worked. There are tricks to make it more general, but we'll deal with those later.

Now, in this case, I want to make a graph by region, so I'm going to clean up the data a bit more. I'll use `gsub` to find and replace text to strip the word "Total" out of all my regions.

```r
#mutate to strip out the totals (note the space after total inside the quotes)
#gsub is a text substitution command, like a find and replace
bp_data<-bp_data%>%mutate(country=gsub("Total ","",country))
```

And now we have the data that we want to graph:

```
  bp_data %>%
    kbl(escape = FALSE) %>%
    kable_styling(fixed_thead = T,bootstrap_options = c("hover", "condensed","responsive"))%>%
  #scroll_box(width = "1000px", height = "400px")%>%
#adding this I() at the end of a chain of commands is like a placeholder in case you would otherwise end with a %>%.
# It makes it easier to paste in code from elsewhere
      I()
```

country	oil	natural_gas	coal	nuclear	hydro	renewables	total
North America	44.53	39.58	10.51	8.193	6.502	9.464	118.8
S. & Cent. America	12.37	5.82	1.19	0.198	7.004	3.526	30.1
Europe	28.72	17.96	10.07	6.678	5.321	11.064	79.8
CIS	9.10	19.84	4.87	2.083	2.326	0.139	38.4
Middle East	17.97	20.18	0.37	0.240	0.116	0.256	39.1
Africa	8.39	5.85	3.97	0.091	1.471	0.485	20.3
Asia Pacific	69.61	32.66	130.50	6.646	17.940	20.241	277.6
World	190.69	141.89	161.47	24.128	40.679	45.176	604.0

This is going to be the part that you find counter-intuitive, and is one of the hardest things about graphing data in groups (or using flat data files), but there are going to be a lot of those in our future, so let's learn about them.

To make this work well, we want to transform our data from ***wide***, with the variables one in each column, to ***long*** with data stacked vertically and an indicator variable for the groupings we need. It's easier to see than to describe, so let's do it and then talk about it:

```
# we're going to pivot the data to a longer style, by country
# we're going to end up values going to a variable called tpes and names going to a variable called source
  bp_data<-bp_data%>%pivot_longer(-country,names_to = "source", values_to = "tpes")

#show me the data
bp_data %>%
    kbl(escape = FALSE) %>%
    kable_styling(fixed_thead = T,bootstrap_options = c("hover", "condensed","responsive"),full_width = T)%>%
  scroll_box(width = "1000px", height = "400px")%>%
#adding this I() at the end of a chain of commands is like a placeholder in case you would otherwise end with a %>%.
# It makes it easier to paste in code from elsewhere
      I()
```

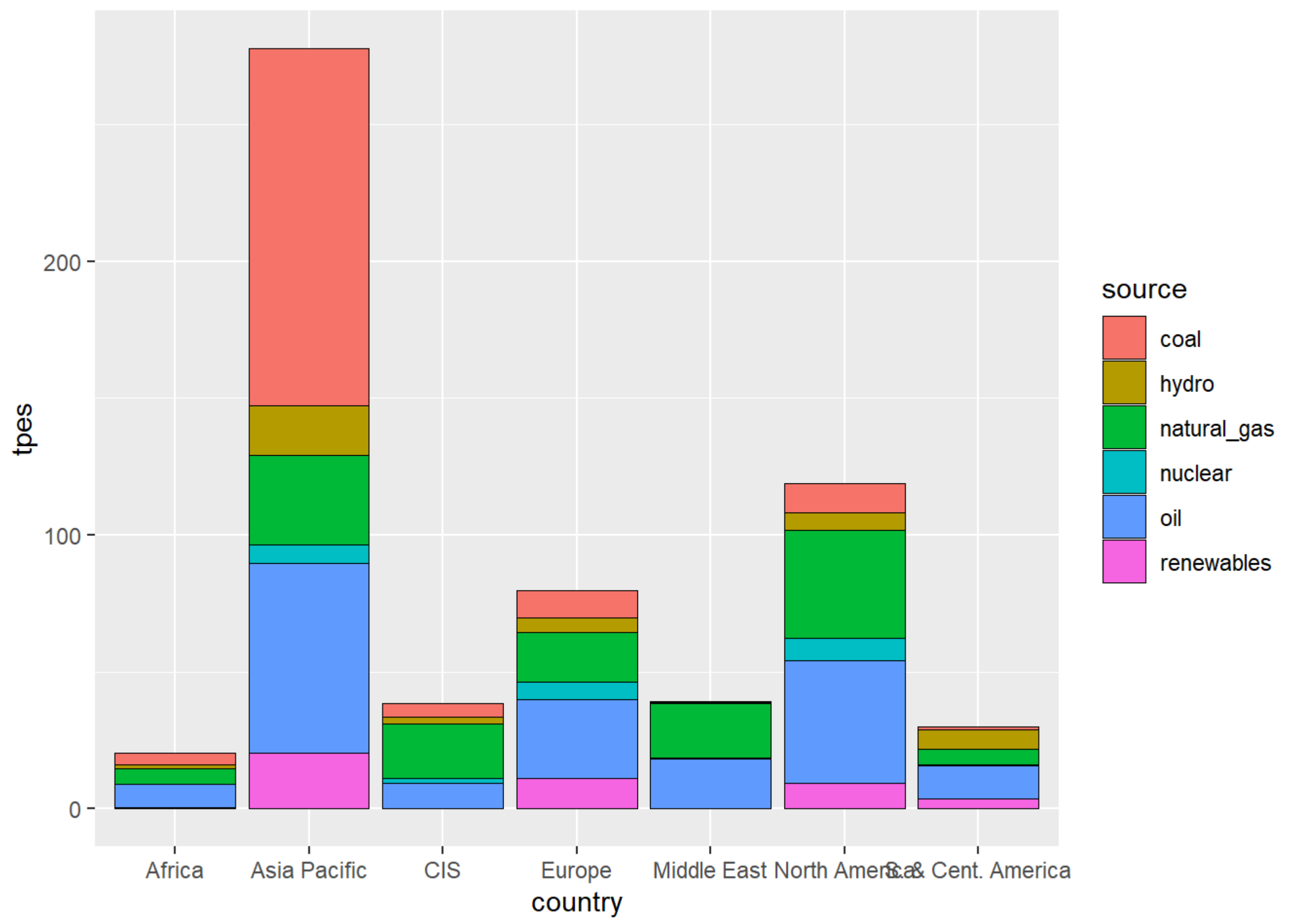
country	source	tpes
North America	oil	44.535
North America	natural_gas	39.579
North America	coal	10.506
North America	nuclear	8.193
North America	hydro	6.502
North America	renewables	9.464
North America	total	118.778
S. & Cent. America	oil	12.372
S. & Cent. America	natural_gas	5.822
S. & Cent. America	coal	1.186
S. & Cent. America	nuclear	0.198
S. & Cent. America	hydro	7.004
S. & Cent. America	renewables	3.526

You're probably not used to seeing data presented this way, and it's not as nice for table visuals, but it's great for graphing.

Let me show you what I mean: let's make a stacked bar plot of the individual constituents of primary energy consumption by region.

```
bp_data %>%
  #take out the global observations
  filter(country!="World")%>%
  #take out the total column
  filter(source!="total")%>%
  #send the data to ggplot
  ggplot()+ #make a graph
```

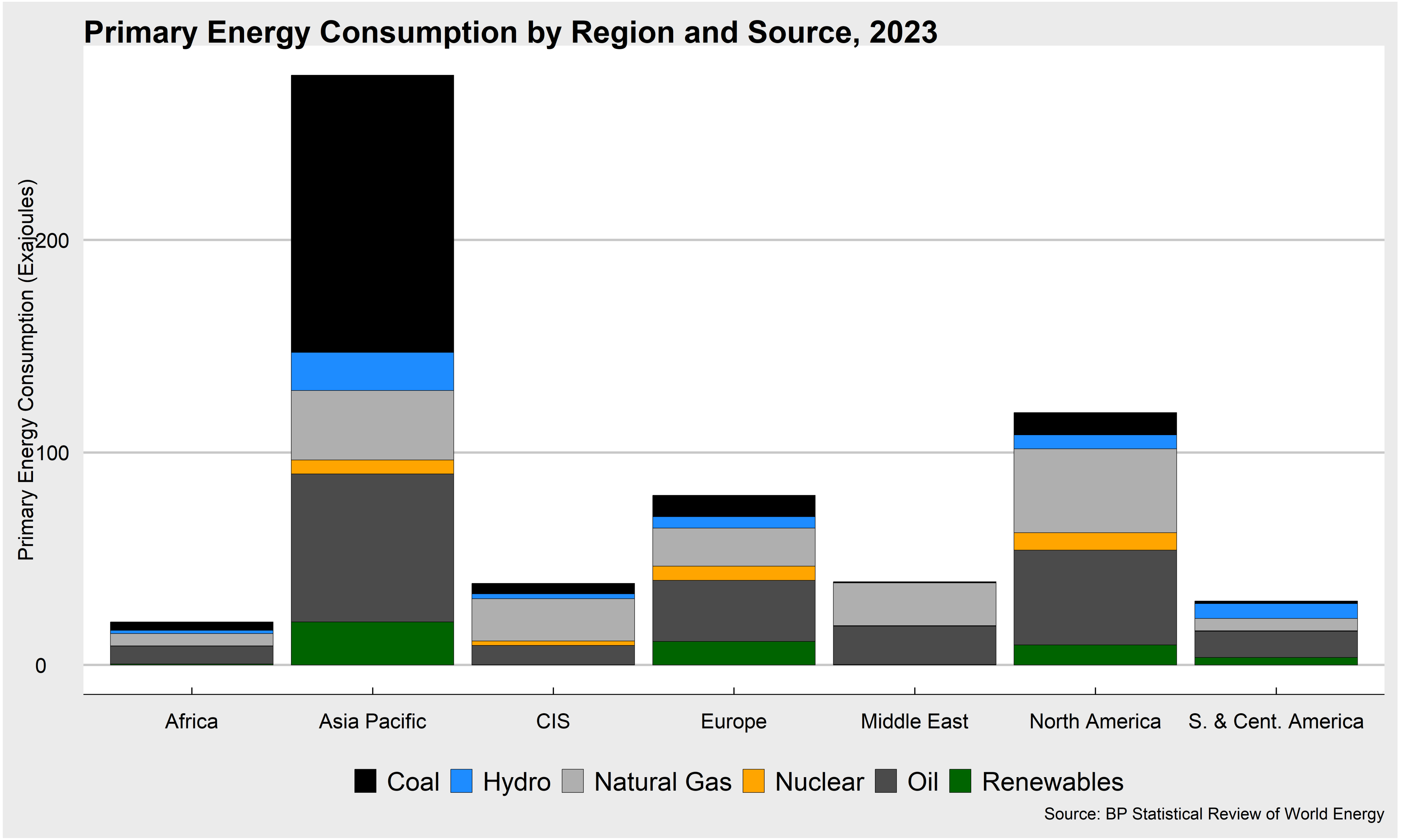
```
# here, we are going to add a column "geom" to our graph
# the data are country on the x axis and tpes on the y axis
geom_col(aes(country,tpes,fill=source),color="black",linewidth=.25)#the color "black" are the "linewidth" describe the outlines
```



That's a basic graph.

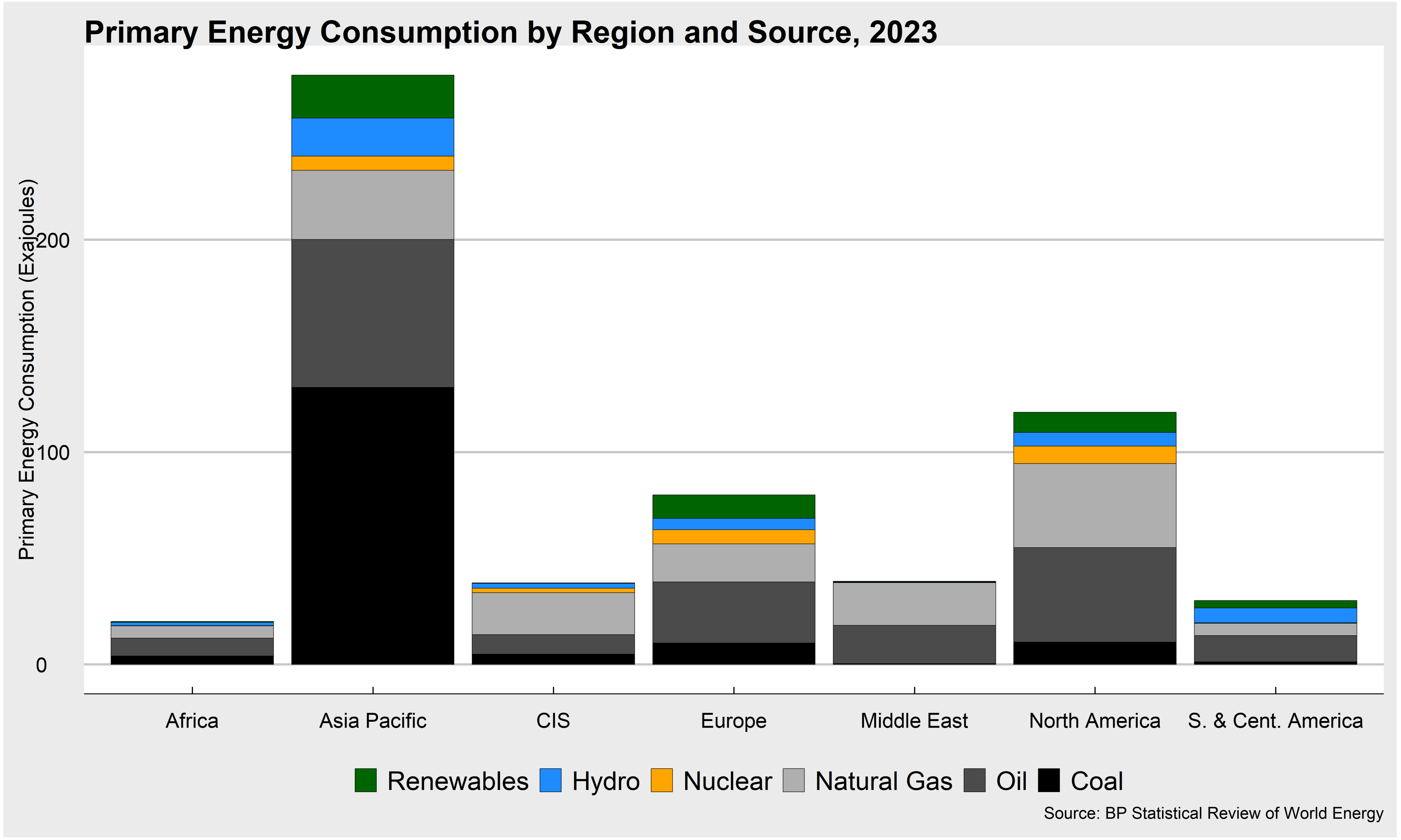
I think we might like to clean it up a bit, and get some colors to suit our sources a bit better. Let's use a couple of other libraries, `ggthemes` and `tools`, so make sure you have those installed if you're going to try to use them.

```r
library(tools)
library(ggthemes)
bp_data %>%
  filter(country!="World")%>%
  filter(source!="total")%>%
  mutate(source=toTitleCase(gsub("_"," ",source)))%>% #strip out the underscore, convert to title case
  #take out the global observations and total column

  ggplot()+
  geom_col(aes(country,tpes,fill=source),color="black",linewidth=.25)+
  theme_economist_white()+#for fun, let's use the economist theme
  guides(fill=guide_legend(nrow = 1))+
  theme(legend.position = "bottom")+
  scale_fill_manual("",values=c("Coal"="black","Oil"="grey30","Natural Gas"="grey70","Nuclear"="orange","Hydro"="dodgerblue","Ren
  theme(text = element_text(size=16))+
  labs(y="Primary Energy Consumption (Exajoules)",x="",
       title="Primary Energy Consumption by Region and Source, 2023",
       caption="Source: BP Statistical Review of World Energy")
```



And, one of the frustrations in ggplot is that it assigns the variable groups as *factors* and, by default, they are sorted alphabetically. In this case, it doesn't matter much, but just so you know what's out there, let me show you one additional step which is to create your own factor for the sources of energy, and use the package `forcats` to reorder them. We'll deal with these more in the future, so this is just a little added feature you might decide you need at some point.

```r
library(forcats)
bp_data %>%
  filter(country!="World")%>%
  filter(source!="total")%>%
  mutate(source=toTitleCase(gsub("_"," ",source)))%>% #strip out the underscore, convert to title case
  mutate(source=as_factor(source))%>% #make the sources factors in the order they appear in the data
  mutate(source=fct_relevel(source,"Coal"))%>% #use fct-relevel to make coal the first factor level
  mutate(source=fct_rev(source))%>% #reverse the order
  #take out the global observations and total column

  ggplot()+
  geom_col(aes(country,tpes,fill=source),color="black",linewidth=.25)+
  theme_economist_white()+#for fun, let's use the economist theme
  guides(fill=guide_legend(nrow = 1))+
  theme(legend.position = "bottom")+
  scale_fill_manual("",values=c("Coal"="black","Oil"="grey30","Natural Gas"="grey70","Nuclear"="orange","Hydro"="dodgerblue","Ren
  theme(text = element_text(size=16))+
  labs(y="Primary Energy Consumption (Exajoules)",x="",
       title="Primary Energy Consumption by Region and Source, 2023",
       caption="Source: BP Statistical Review of World Energy")
```



And, that's this week's data lesson. Try it out for yourself in your own document, using these code chunks as you see fit, to start to get ready for the first graded assignment later in the month. Here's my `week1_ajl.Rmd` file for your reference. If you got lost here a bit, then you can download that file, knit it, and then work backwards through it.