# DataVizA Tutorial: DataMunging: Solutions

*Department of Econometrics and Business Statistics, Monash University*

*Tutorial 6*

## First Normal Form

1. Discuss whether the following databases satisfy first normal form.

Database A:

| Name | Social Media Username |
|---|---|
| Jane Smith | Facebook: jsChampion |
| Kamaru Usman | Twitter: kusman, LinkedIn: lx99 |
| Li Xiao | WeChat: lx99 |

Database B:

| Name | Social Media Username |
|---|---|
| Jane Smith | Facebook: jsChampion |
| Kamaru Usman | Twitter: kusman |
| Kamaru Usman | LinkedIn: lx99 |
| Li Xiao | WeChat: lx99 |

*Database A has the same issue as seen in lectures. Kamaru Usman has two social media accounts so the entry is not atomic. The second Database resolves this issue. However arguably the variable Social Media Username is still not atomic. There are two separate pieces of information, the social media platform and the username. A better database would store these as two separate variables.*

*Another point worth mentioning is that even the variable name might not be considered to be atomic. For example in some contexts it may be important to serparate family name from given names.*

## Swiss Exports: Full Data

The file *SwissExportsFull.csv* contains the full export data for Switzerland. Each row represents a different date. The first column is the date variable, the second column is the year only and each remaining column measures exports to a different country.

2. Read the data into R

```
library(tidyverse)
SwissWide<-read_csv('SwissExportsFull.csv')
```

```
## Warning: Missing column names filled in: 'X154' [154]
```

```
#This works but with a quirky warning.  One country code is NA for
#Namibia, however R treats NA as a missing value.  It can be fixed
#with

SwissWide<-read_csv('SwissExportsFull.csv')%>%
  rename(`NA`=X154)
```

```
## Warning: Missing column names filled in: 'X154' [154]
```

```
SwissWide
```

3. Get the data into long form using the `gather` function

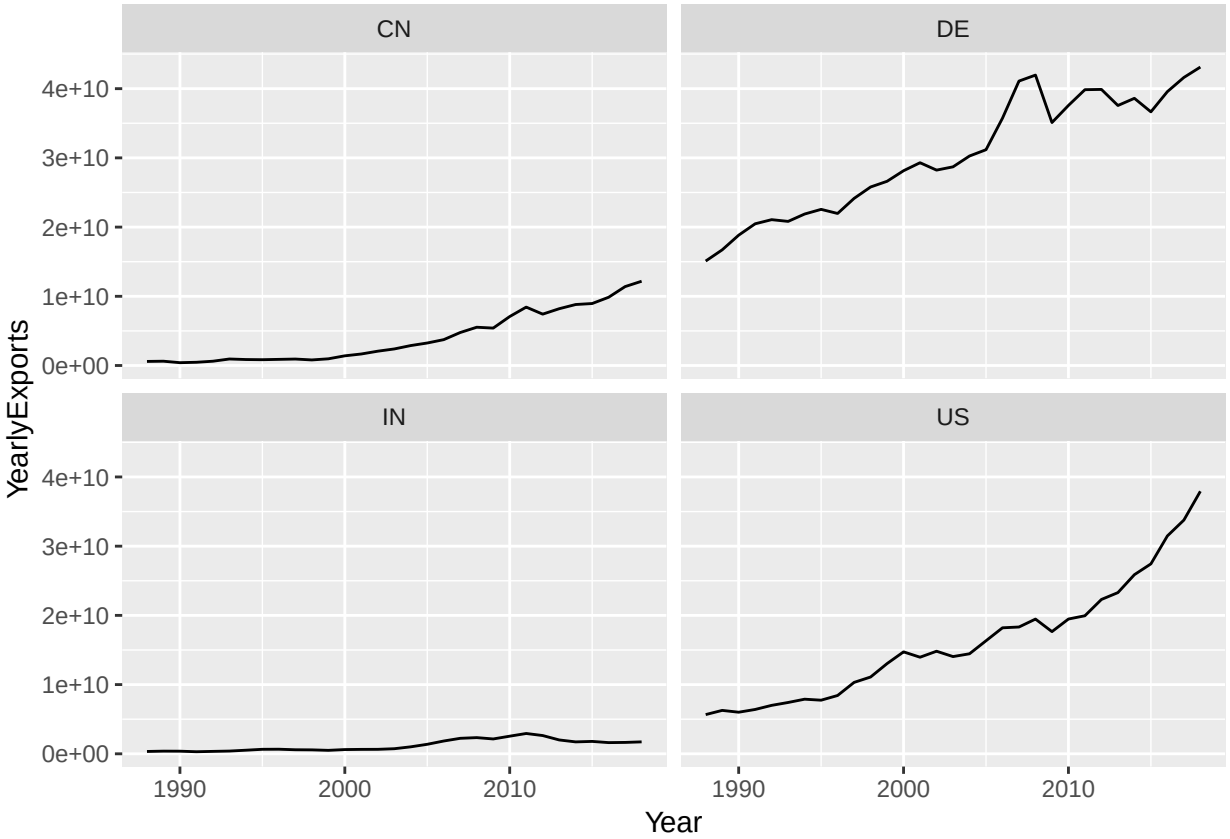
```
library(tidyverse)
SwissLong<-gather(data = SwissWide,
       key = Country, #Column names become variable
       value = Exports, #All numbers are exports
       -Date,-Year) #Do not gather these variables
SwissLong
```

4. Recall that in the previous tutorial, one issue was that monthly data were noisy. Using `group_by` and `summarise` create a new dataset of yearly aggregate exports to each country.

```
SwissLong%>% #This is much easier with long data
  select(-Date)%>% #Eventually cannot aggregate dates
  group_by(Year,Country)%>%
  summarise(YearlyExports=sum(Exports))->SwissYearly
SwissYearly
```

5. Plot time series line plots of Swiss exports to Germany (DE), the USA (US), China (CN) and India (IN). Facet by country.

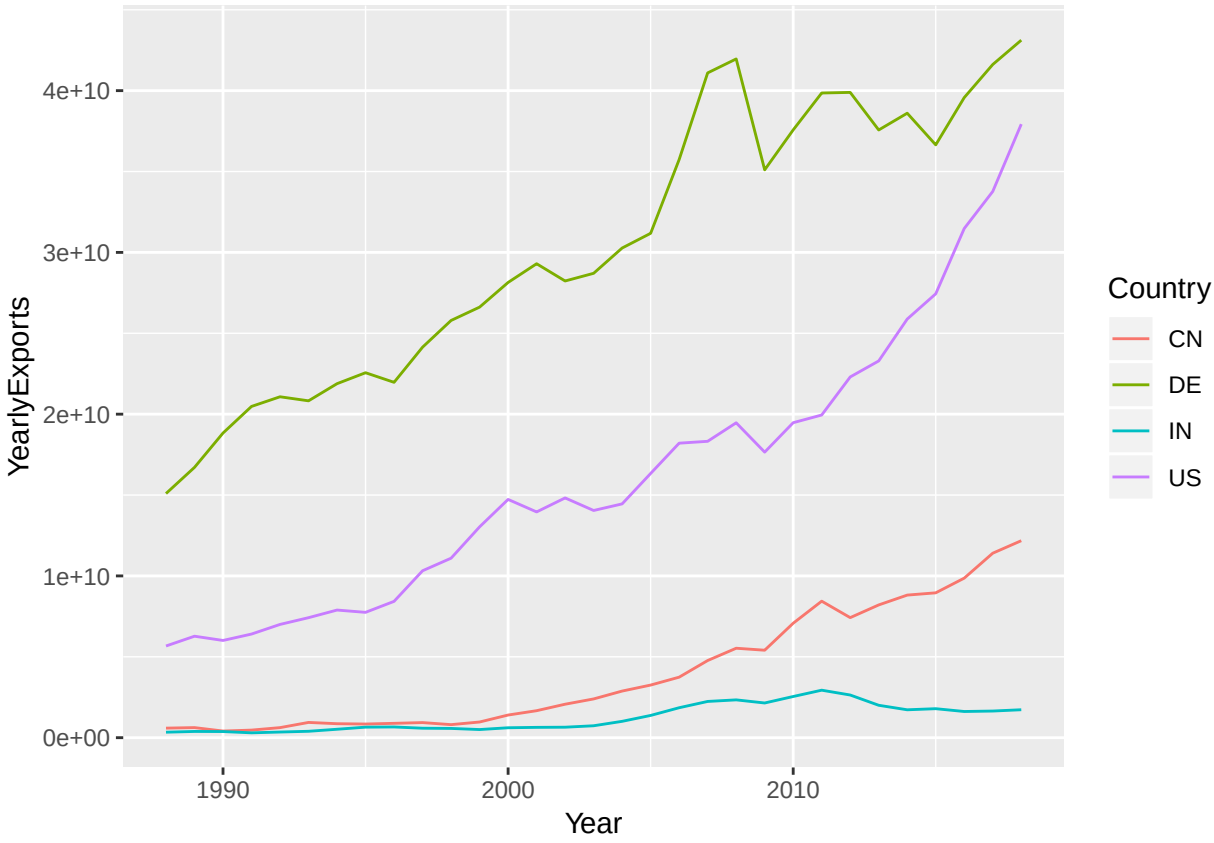
```
SwissYearly%>%
  filter(Country %in% c('DE','US','CN','IN'))%>%
  ggplot(aes(x=Year,y=YearlyExports))+
  geom_line()+
  facet_wrap(~Country)
```




6. Plot these four lines on a single plot with each country in a different colour. Hint: Use the aesthetic

```
SwissYearly%>%
  filter(Country %in% c('DE','US','CN','IN'))%>%
```

```
  ggplot(aes(x=Year,y=YearlyExports,col=Country))+
  geom_line()
```




7. Comment on these plots

*Both plots show the same information, Swiss exports are mostly trending upwards, Germany and the US are much bigger trading partners for Switzerland than China and India. Any seasonality has been lost by taking the yearly aggregate.*

*The second plot allows us to more easily line up the impact of the GFC. There was a drop in exports to all markets but this was most pronounced in Germany and hardly there at all in China and India. Post GFC the recovery in exports has been quickest and strongest for the US and China. Exports to Germany were volatile but have picked up in the last three years while growth in exports to India has stagnated*

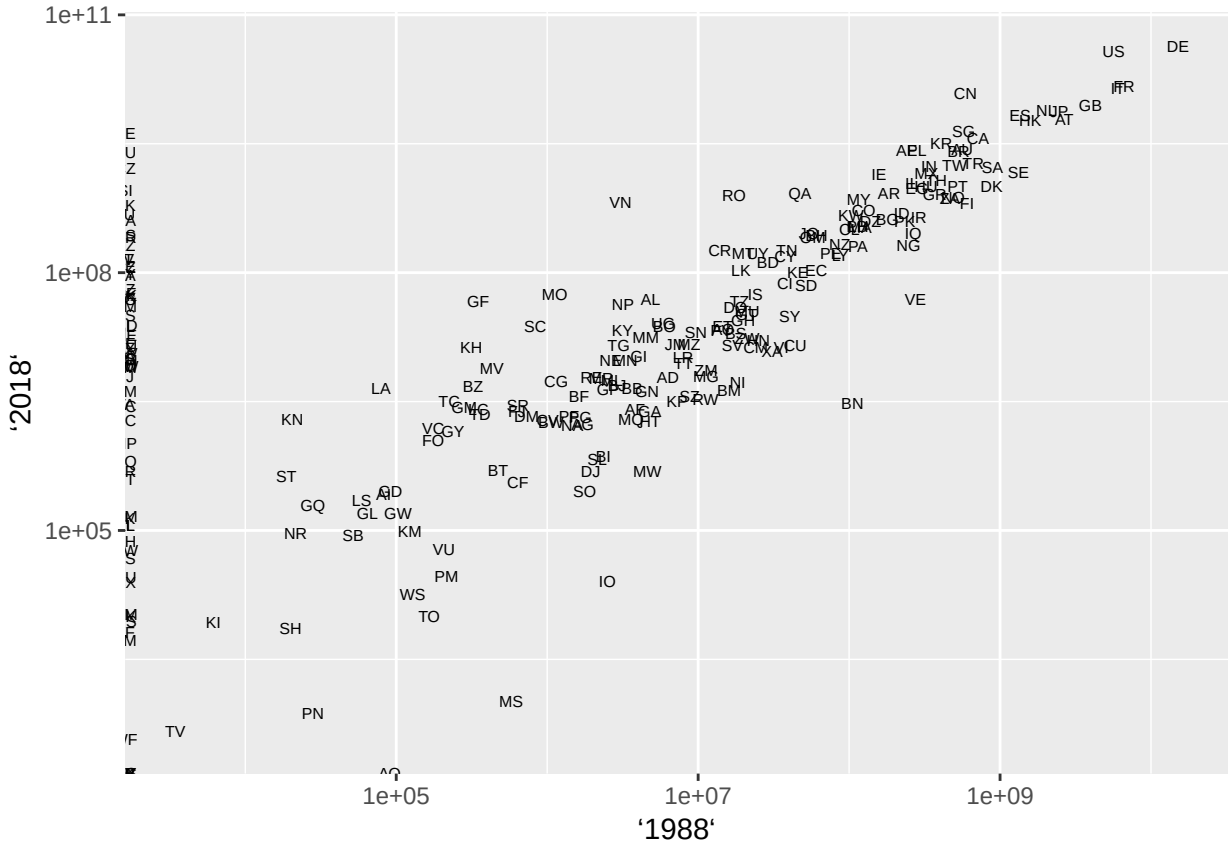
7. Now produce a scatterplot on a log-log scale of 1988 exports against 2018 exports. Use country abbreviations rather than points

```
SwissYearly%>%
  filter(Year %in% c(1988,2018))%>% #Filter years
  spread(Year,YearlyExports)-> #Need to spread
  SwissYearlyWide

SwissYearlyWide%>%
  ggplot(aes(x=`1988`,y=`2018`,label=Country))+
  geom_text(size=2)+
  scale_x_log10()+scale_y_log10()
```

```
## Warning: Transformation introduced infinite values in continuous x-axis
```

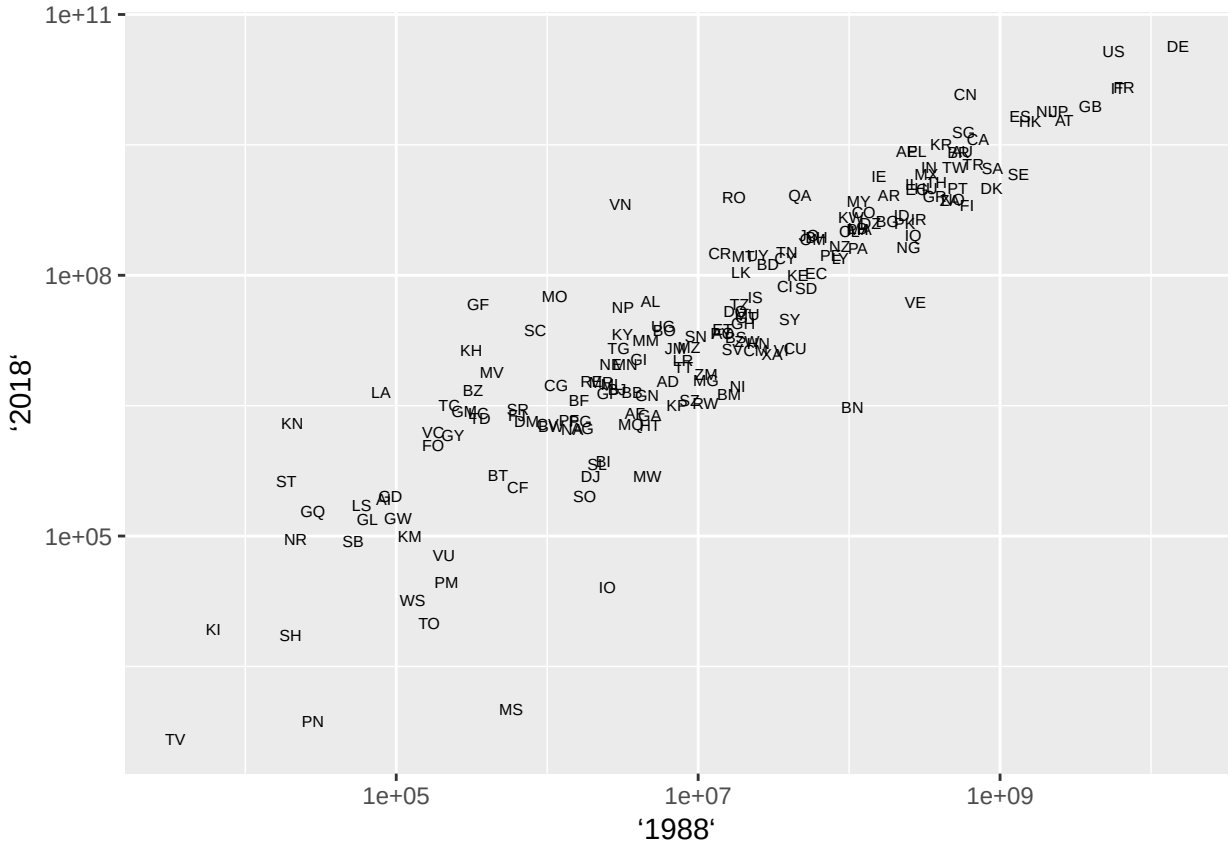
```
## Warning: Transformation introduced infinite values in continuous y-axis
```




8. Produce the same plot but remove all countries for which exports are zero in either 1988 or 2018.

```
#Although it works, avoid the temptation to define the logical
#statement as it is written in words !((`1988`==0)|(`2018`==0))

SwissYearlyWide%>%
  filter((`1988`!=0)&(`2018`!=0))%>% #Filter years
  ggplot(aes(x=`1988`,y=`2018`,label=Country))+
  geom_text(size=2)+
  scale_x_log10()+scale_y_log10()
```




```
#Note that the anomalies with French Guyana and Papua New Guinea
#are smoothed out when the yearly aggregate is used.
```