**Detecting outliers with Z-Scores**

As Dhavide demonstrated in the video using the zscore function, you can apply a .transform() method after grouping to apply a function to groups of data independently. The z-score is also useful to find outliers: a z-score value of +/- 3 is generally considered to be an outlier.

In this example, you're going to normalize the Gapminder data in 2010 for life expectancy and fertility by the *z-score per region*. Using boolean indexing, you will filter out countries that have high fertility rates and low life expectancy for their region.

The Gapminder DataFrame for 2010 indexed by 'Country' is provided for you as gapminder\_2010.

**INSTRUCTIONS**

* Import zscore from scipy.stats.
* Group gapminder\_2010 by 'region' and transform the ['life','fertility'] columns by zscore.
* Construct a boolean Series of the bitwise or between standardized['life'] < -3 and standardized['fertility'] > 3.
* Filter gapminder\_2010 using .loc[] and the outliers Boolean Series. Save the result as gm\_outliers.
* Print gm\_outliers.

# Import zscore

from scipy.stats import zscore

# Group gapminder\_2010: standardized

standardized = gapminder\_2010.groupby('region')['life','fertility'].transform(zscore)

# Construct a Boolean Series to identify outliers: outliers

outliers = (standardized['life'] < -3) | (standardized['fertility'] > 3)

# Filter gapminder\_2010 by the outliers: gm\_outliers

gm\_outliers = gapminder\_2010.loc[outliers]

# Print gm\_outliers

print(gm\_outliers)