Lassoing Data

Coursera Course by John Hopkins University

INSTRUCTORS: Dr. Jeff Leek, Dr. Roger D. Peng, Dr. Brian Caffo

Contents

Intro

One of the major components of a data scientist's job is to collect and clean data. Whether at a small organization or a major enterprise, the first step in using data is getting, cleaning and understanding the data. In this course, we focus on R packages and a few outside tools that can be used to collect data from a variety of sources, from Excel files to databases like MySQL. We will also cover a variety of formats including JSON, XML, and flat files (.csv, .txt).

The emphasis of this course is on creating tidy data sets that can be used in downstream analyses

Finding Data and Reading Various File Types

Obtaining Data Motivation

```
* This course covers the "nitty gritty" of getting data ready for analysis
        + finding where the data are and extracting it out.
        + Tidy data principles and how to make data tiny
        + Practical implementation through a range of R packages
* Data often is not nicely in a `.csv`, but rather
        + parsed in a text file and needs to be processed
        + formatted in `JSON` format
        + Free text instructions where a phrase is to be extracted
        + In data bases like `mySQL` ("My Sequel") or `MongoDB` (Mon-go D-B)
* Where are data?
        + Websites
                - **[Online Datasets](data.baltimorecity.gov)**
        + APIs
* Steps for going from *Raw data* to *data communication*
**Raw data -> Processing script -> tidy data** -> data analysis -> data communication
        +This course focuses on going from Raw data to Tidy data
```

Raw and Processed Data

- **Data** - Values of qualitative or quantitative variables, belonging to a set of items. + **Qualitative**: Country of orgin, sex, treatment.
  + **Quantitative**: Height, weight, blood pressure
- **Raw Data** + The orginal source of the data + Often hard to use for data analyses
  + Data analysis *includes* processing
  + Raw data may only need to be processed once
- **Processed data** + Data that is ready for analysis
  + Processing can include merging, subsetting, transforming, etc.
  + There may be standards for how it's processed
  + All steps and actions taken should be recorded
- Raw Data could be considered in several layers. + If processing genomes - the og picture in the machine is the raw data, - the image is evaluated to determine the prodominet color, this could be considered raw data.
  - The machine then outputs a text file of these readings, this also is raw data that you would now need to proccess further past the machine. + The journey this data takes is to be mentioned as to not ignore the orgin of the true raw data

Components of Tidy Data

1. The raw data

2. A tidy data set

3. A code book describing each variable and its values in the tidy data set.

4. An explicit and exact recipe you used to go from 1 -> 2, 3… (This will be the R scripts you write)

- When looking at a particular data set, the *raw data* is the rawest form of the data you have access to. + Examples: - The binary file your measurement machine spits out
  - The unformatted Excel file with 10 worksheets the company you contracted with sent you
  - The complicated JSON data you got from scraping the Twitter API
  - The hand-entered numbers you collected looking through a microscope
  + You know the raw data is in the right format if you - You ran no software on the data
  - Did not manipulate any of the numbers in the data
  - You did not remove any data from the data set
  - You did not summarize the data in any way

- The tidy Data

1) Each variable you measure should be in one column

2) Each different observation of that variable should be in a different row

3) There should be one table for each "kind" of variable

4) If you have multiple tables, they should include a column in the table that allows them to be linked

```
+ *Some other tips*
        - Include a row at the top of each file with variable names.
        - Make variable names human readable; `AgeAtDiagnosis` instead of `AgeDx`
        - In general data should eb saved in one file per table.
```

- The Code Book + Information about the variables (including units!) int he data set not contained in the tidy data
  + Information about the summary choices you made
  + Information about the experimental study design you used

```
+ *Some other tips*
        - A common format for this document is a Word/text file (or markdown as thats co
        - There should be a section called "Study Design" that ahs a thorough descriptio
        - There should be a section called "Code book" that describes each variable and
```

- The instruction list + Ideally a computer script (in R :-) but I suppose Python is ok too…)
  + The input for the script is the raw data
  + The output is the processed, tidy data
  + There are no parameters to the script

```
+ In some cases it will not be possible to script every step. In that case you should pr
        1) Step 1 - take the raw file, run version 3.1.2 of summarize software with para
```

```
2) Step 2 - run the sodtware separately for each sample
3) Step 3 - take column three fo ouputfile.out for each sample and that iss the
```

- Be detailed in how you converted raw to tidy data. + Example: **(A Critique of Reinhard and Rogoff)[http://www.cc.com/video-clips/dcyvro/the-colbert-report-austerity-s-spreadsheet-error]**

Downloading Files

- A basic component of working with data is knowing your working directory
  - The two main commands are `getwd()` and `setwd()`.
  - Be aware of relative versus absolute paths
    - Relative - `setwd("./data")`, `setwd("../")`
    - Absolute - `setwd("/Users/jtleek/data/")`
  - Important difference in Windows, they us \ instead of /: `setwd("C:\\Users\\Andrew\\Downloads")`
- The directory that is **up** from where you are is like the parent folder.
- Checking for and creating directories
  - `file.exists("directoryName")` will check to see if the directory exists
  - `dir.create("directoryName)` will create a new directory called "`directoryName`" if it doesn't exist

```
if(!file.exists("data")){
  dir.create("data")
}
```

- Lassoing "cattle"(data) from the internet: `download.file()`
  - Downloads a file from the internet
  - Even if you could do this by hand, it helps for reproducibility
  - Useful for downloading tab-delimited, csv, and other files.
  - Important parameters are *url*, *destfile*, and *method* (Source of data, desitnation file, method)
    - Right click on file you want to dowload, select "*copy link location*" (**(Example with Balimore camera data)[https://data.baltimorecity.gov/Transportation/Baltimore-Fixed_Speed-Cameras/dz54-2aru]**)

```
if(!file.exists("data")){
  dir.create("data")
}

fileUrl <- "https://data.baltimorecity.gov/api/views/dz54-2aru/rows.csv?accessType=DOWNLOAD"
```

```r
download.file(fileUrl, destfile = "./data/cameras.csv", method = "curl")
##Because url is https "curl" has to be specified for Mac & Linix
list.files("./data")
```

```
##  [1] "cameras.csv"
##  [2] "cameras.xlsx"
##  [3] "Chicago.rds"
##  [4] "CRANpackages.csv"
##  [5] "debate_transcripts_v3_2020-02-26.csv"
##  [6] "Edu.csv"
##  [7] "GDP.csv"
##  [8] "IdahoHousing06.csv"
##  [9] "jeff.jpg"
## [10] "nytimes_presidential_elections_2016_results_county.csv"
## [11] "president_polls.csv"
## [12] "Q4Edu.csv"
## [13] "Q4GDP.csv"
## [14] "Q4IdahoHousing06.csv"
## [15] "restaurants.csv"
## [16] "review.csv"
## [17] "solutions.csv"
```

```r
dateDownloaded <- date()
dateDownloaded
```

```
## [1] "Sat Mar 28 20:45:53 2020"
```

- Additional notes about `download.file()`
  - If the url starts with *http* you can use `download.file()`
  - If the url starts with *https* on Mac of Linix you need to set `method = "curl"`
  - If your internet is poor or the file is big, this might take a while
  - Be sure to record when you downloaded.

Reading Local "Flat" Files

- Kind of a review from R lectures
- Most common way to load local data is `read.table()`
  - Requires more parameters than some of the other functions
  - Can be kinda slow, so its a poor mix with big data
  - Reads the data straight into RAM - So big data can cause issues
  - Important parameters:

        * `file` - *Indicates input file*
        * `header` - *Logical for if there is a header*
        * `sep` - *Character that seperates data, default is a tab*
        * `row.names` - *Optional vector of row names*

```r
cameraData <- read.table("./data/cameras.csv", sep = ",", header = TRUE)
head(cameraData)
```

```
##                                    address direction          street
## 1                  GARRISON BLVD & WABASH AVE       E/B     Garrison \n
## 2                      HILLEN ST & FORREST ST       W/B       Hillen \n
## 3                 EDMONDSON AVE & N ATHOL AVE       E/B      Edmonson\n
## 4                      YORK RD & GITTINGS AVE       S/B      York Rd \n
## 5                    RUSSELL ST & W HAMBURG ST       S/B       Russell\n
## 6 S MARTIN LUTHER KING JR BLVD & W PRATT ST       S/B MLK Jr. Blvd \n
##      crossStreet                intersection             Location.1
## 1     Wabash Ave     Garrison \n & Wabash Ave (39.341209, -76.683117)
## 2     Forrest St       Hillen \n & Forrest St  (39.29686, -76.605532)
## 3 Woodbridge Ave Edmonson\n  & Woodbridge Ave (39.293453, -76.689391)
## 4    Gitting Ave     York Rd \n & Gitting Ave (39.370493, -76.609812)
## 5     Hamburg St      Russell\n  & Hamburg St (39.279819, -76.623911)
## 6       Pratt St   MLK Jr. Blvd \n & Pratt St (39.286027, -76.627846)
##   X2010.Census.Neighborhoods X2010.Census.Wards.Precincts Zip.Codes
## 1                        252                           63     27295
## 2                        179                          108     13645
## 3                        213                           75     27950
## 4                         37                          270     14009
## 5                        250                          178     27953
## 6                         11                          168     27953
```

- `read.csv` automatically sets `sep = ","` and `header = TRUE`
- Addtional parameters for `read.table()`
    - `na.strings` - sets the character that represents a missing value
    - `nrows` - how many rows to read fo the file (e. g. `nrows = 10` reads in 10 lines)
    - `skip` - number of lines to skip befores starting to read
    - `quote` - tells R whether there are any quoted values ("Like This"); `quote=""` indicates there are no quotes
        * If ' or " are placed in data values setting `quote=""` will often resolve this

Reading Excel Files

- Still probably the most widely used format for sharing data

```
if(!file.exists("data")){
  dir.create("data")
}

fileUrl <- "https://data.baltimorecity.gov/api/views/dz54-2aru/rows.csv?accessType=DOWNLOAD&bom
fileLoc <- paste(getwd(), "/data/cameras.xlsx", sep = "")
download.file(fileUrl, destfile = fileLoc, method = "curl")
dateDownloaded <- date()
#xlsx is an excel file type
ogDir <- getwd()
setwd(paste(getwd(), "/data", sep = ""))
getwd()
```

```
## [1] "/home/phiprime/Documents/Education/R/LassoingDataNotes/data"
```

```
#Keep getting error, Abondoned in Place until later
# library(readxl)
# cameraData <- read_excel("cameras.xlsx", sheet = 1) #ERROR
# head(cameraData)
setwd(ogDir)
```

- Parameters:
  - `sheetIndex` - indicates the sheet to read from
  - `colIndex` - indicates range of columns to read from

  - `rowIndex` - indicates range of rows to read from
- Additional notes:
  - `write.xlsx` function will write out an Excel file

  - `read.xlsx2` is much faster than `read.xlsx` but for reading subsets of rows may be slightly unstable.

  - The XLConnect package has more options for writing and manipulating Excel files
    * XLConnect vignette is a good place to start for that package

  - Best to store data in `.csv` or tab separated files(`.tab`/`.txt`) as they're easier to distribute

Reading XML

- XML
  - Extensible mark up language that is frequently used to store structured data

  - Widely used in internet applications
    * Extracting XML is the basis for most web scraping

- Components:
  * Markup - labels that give the text structure

  * Content - the actual text of the document
- Tags, elements and attributes
  - Tags correspong to general labels
    * Start tags `<section>`
    * End tags `</section>`

    * Empty tags `<line-break />`

  - Elements are specific examples of tags
    * ‘ Hello, world

  - Attributes are componets of the label
    * `<img src="jeff.jpg" alt="instructor"/>`
    * `<step number="3"> Connect A to B. </step>`
- **(Example XML file)**[**http://www.w3schools.com/xml/simple.xml**]
  - `<food>` has multiple subclasses, such as `<name>`, `<description>`, . . .

```
library(XML)
library(RCurl)
fileUrl <- getURL("http://www.w3schools.com/xml/simple.xml")
#Throwing Error#doc <- xmlTreeParse(fileUrl, useInternalNodes = TRUE) #Parses xml into its use
# rootNode <- xmlRoot(doc) #"Wrapper for entire document"
# xmlName(rootNode) #Returns name of root
# names(rootNode)#Returns names of 1st branch from root
#
# #Looking at particular elements of XML
# rootNode[[1]]#REturns first food element
# rootNode[[1]][[1]]#Returns First element of First food element
#
# #Extracting parts of the file
# xmlSApply(rootNode, xmlValue)#Gets xmlValue of each tag under rootNode
```

- `XPath` language can let you get specific attributes out of the XML
  - **(Read More)**[**http://www.stat.berkeley.edu/~statcur/Workshop2/Presentations/XML.pdf**]
  - `/node` - Top level node

  - `//node` - Node at any level

  - `node[@attr-name]` Node with an attribute name

  - `node[@attr-name='bob']` - Node with attribute name attr-name=‘bob’

```
# #Get the items on the menu and prices)
# xpathSApply(rootNode, "//name", xmlValue)#Takes out all elements that are tagged with "name"
# xpathSApply(rootNode, "//price", xmlValue)
```

- Exacting content by attributes from Source Code
  - Use `htmlTreeParse()` for reading in source code as XML
    * Following code has become out of date from the lecture, I tried to update it but it

```
library(XML)
library(RCurl)

fileUrl <- "https://www.espn.com/nfl/team/_/name/bal/baltimore-ravens"
doc <- htmlTreeParse(getURL(fileUrl), useInternalNodes = TRUE)
scores <- xpathSApply(doc, "//div[@class='score']", xmlValue)
teams <- xpathSApply (doc, "//div[@class='game-info']", xmlValue)
scores
```

```
##  [1] "28-12" "59-10" "23-17" "33-28" "40-25" "26-23" "23-17" "30-16" "37-20"
## [10] "49-13" "41-7"  "45-6"  "20-17" "24-17" "42-21" "31-15" "28-10" "29-0"
## [19] "26-13" "26-15" "20-7"
```

```
teams
```

```
##  [1] "vs  Titans"    "@  Dolphins"   "vs  Cardinals" "@  Chiefs"
##  [5] "vs  Browns"    "@  Steelers"   "vs  Bengals"   "@  Seahawks"
##  [9] "vs  Patriots"  "@  Bengals"    "vs  Texans"    "@  Rams"
## [13] "vs  49ers"     "@  Bills"      "vs  Jets"      "@  Browns"
## [17] "vs  Steelers"  "vs  Jaguars"   "vs  Packers"   "@  Eagles"
## [21] "@  Redskins"
```

Reading JSON

- JSON
  - JavaScript Object Notation

  - Lightweight data storage

  - Common format for data from application programming interfaces (APIs)

  - Similar structure to XML but different syntax/format

  - Data stored as:
    * Numbers (double)
    * Strings (double quoted)
    * Boolean (true or false)

    * Array (ordered, comma separated enclsoed in square brakets[])
    * Object (unordered, comma separated collection of key:value pairs in curley brackets {})
  - (Example)[https://api.github.com/users/jtleek/repos]

```
library(jsonlite)
jsonData <- fromJSON("https://api.github.com/users/jtleek/repos")
```

```r
names(jsonData) #Displays top level components of data.frame
```

```
##  [1] "id"                "node_id"           "name"
##  [4] "full_name"         "private"           "owner"
##  [7] "html_url"          "description"       "fork"
## [10] "url"               "forks_url"         "keys_url"
## [13] "collaborators_url" "teams_url"         "hooks_url"
## [16] "issue_events_url"  "events_url"        "assignees_url"
## [19] "branches_url"      "tags_url"          "blobs_url"
## [22] "git_tags_url"      "git_refs_url"      "trees_url"
## [25] "statuses_url"      "languages_url"     "stargazers_url"
## [28] "contributors_url"  "subscribers_url"   "subscription_url"
## [31] "commits_url"       "git_commits_url"   "comments_url"
## [34] "issue_comment_url" "contents_url"      "compare_url"
## [37] "merges_url"        "archive_url"       "downloads_url"
## [40] "issues_url"        "pulls_url"         "milestones_url"
## [43] "notifications_url" "labels_url"        "releases_url"
## [46] "deployments_url"   "created_at"        "updated_at"
## [49] "pushed_at"         "git_url"           "ssh_url"
## [52] "clone_url"         "svn_url"           "homepage"
## [55] "size"              "stargazers_count"  "watchers_count"
## [58] "language"          "has_issues"        "has_projects"
## [61] "has_downloads"     "has_wiki"          "has_pages"
## [64] "forks_count"       "mirror_url"        "archived"
## [67] "disabled"          "open_issues_count" "license"
## [70] "forks"             "open_issues"       "watchers"
## [73] "default_branch"
```

```r
names(jsonData$owner)#Goes to owner Data.frame, which is a data.frame itself
```

```
##  [1] "login"               "id"                  "node_id"
##  [4] "avatar_url"          "gravatar_id"         "url"
##  [7] "html_url"            "followers_url"       "following_url"
## [10] "gists_url"           "starred_url"         "subscriptions_url"
## [13] "organizations_url"   "repos_url"           "events_url"
## [16] "received_events_url" "type"                "site_admin"
```

```r
jsonData$owner$login#Leaf of data.frame; (looking at jtleek's depo)
```

```
##  [1] "jtleek" "jtleek" "jtleek" "jtleek" "jtleek" "jtleek" "jtleek" "jtleek"
##  [9] "jtleek" "jtleek" "jtleek" "jtleek" "jtleek" "jtleek" "jtleek" "jtleek"
## [17] "jtleek" "jtleek" "jtleek" "jtleek" "jtleek" "jtleek" "jtleek" "jtleek"
## [25] "jtleek" "jtleek" "jtleek" "jtleek" "jtleek" "jtleek"
```

- Writing data frames to JSON

```r
myjson <- toJSON(iris, pretty=TRUE)
cat(myjson) #Display json File
```

```
## [
```

```
##   {
##     "Sepal.Length": 5.1,
##     "Sepal.Width": 3.5,
##     "Petal.Length": 1.4,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 4.9,
##     "Sepal.Width": 3,
##     "Petal.Length": 1.4,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 4.7,
##     "Sepal.Width": 3.2,
##     "Petal.Length": 1.3,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 4.6,
##     "Sepal.Width": 3.1,
##     "Petal.Length": 1.5,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5,
##     "Sepal.Width": 3.6,
##     "Petal.Length": 1.4,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.4,
##     "Sepal.Width": 3.9,
##     "Petal.Length": 1.7,
##     "Petal.Width": 0.4,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 4.6,
##     "Sepal.Width": 3.4,
##     "Petal.Length": 1.4,
##     "Petal.Width": 0.3,
##     "Species": "setosa"
```

```
##   },
##   {
##     "Sepal.Length": 5,
##     "Sepal.Width": 3.4,
##     "Petal.Length": 1.5,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 4.4,
##     "Sepal.Width": 2.9,
##     "Petal.Length": 1.4,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 4.9,
##     "Sepal.Width": 3.1,
##     "Petal.Length": 1.5,
##     "Petal.Width": 0.1,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.4,
##     "Sepal.Width": 3.7,
##     "Petal.Length": 1.5,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 4.8,
##     "Sepal.Width": 3.4,
##     "Petal.Length": 1.6,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 4.8,
##     "Sepal.Width": 3,
##     "Petal.Length": 1.4,
##     "Petal.Width": 0.1,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 4.3,
##     "Sepal.Width": 3,
##     "Petal.Length": 1.1,
##     "Petal.Width": 0.1,
```

```
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.8,
##     "Sepal.Width": 4,
##     "Petal.Length": 1.2,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.7,
##     "Sepal.Width": 4.4,
##     "Petal.Length": 1.5,
##     "Petal.Width": 0.4,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.4,
##     "Sepal.Width": 3.9,
##     "Petal.Length": 1.3,
##     "Petal.Width": 0.4,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.1,
##     "Sepal.Width": 3.5,
##     "Petal.Length": 1.4,
##     "Petal.Width": 0.3,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.7,
##     "Sepal.Width": 3.8,
##     "Petal.Length": 1.7,
##     "Petal.Width": 0.3,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.1,
##     "Sepal.Width": 3.8,
##     "Petal.Length": 1.5,
##     "Petal.Width": 0.3,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.4,
##     "Sepal.Width": 3.4,
##     "Petal.Length": 1.7,
```

```
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.1,
##     "Sepal.Width": 3.7,
##     "Petal.Length": 1.5,
##     "Petal.Width": 0.4,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 4.6,
##     "Sepal.Width": 3.6,
##     "Petal.Length": 1,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.1,
##     "Sepal.Width": 3.3,
##     "Petal.Length": 1.7,
##     "Petal.Width": 0.5,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 4.8,
##     "Sepal.Width": 3.4,
##     "Petal.Length": 1.9,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5,
##     "Sepal.Width": 3,
##     "Petal.Length": 1.6,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5,
##     "Sepal.Width": 3.4,
##     "Petal.Length": 1.6,
##     "Petal.Width": 0.4,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.2,
##     "Sepal.Width": 3.5,
```

```
##     "Petal.Length": 1.5,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.2,
##     "Sepal.Width": 3.4,
##     "Petal.Length": 1.4,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 4.7,
##     "Sepal.Width": 3.2,
##     "Petal.Length": 1.6,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 4.8,
##     "Sepal.Width": 3.1,
##     "Petal.Length": 1.6,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.4,
##     "Sepal.Width": 3.4,
##     "Petal.Length": 1.5,
##     "Petal.Width": 0.4,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.2,
##     "Sepal.Width": 4.1,
##     "Petal.Length": 1.5,
##     "Petal.Width": 0.1,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.5,
##     "Sepal.Width": 4.2,
##     "Petal.Length": 1.4,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 4.9,
```

```
##     "Sepal.Width": 3.1,
##     "Petal.Length": 1.5,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5,
##     "Sepal.Width": 3.2,
##     "Petal.Length": 1.2,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.5,
##     "Sepal.Width": 3.5,
##     "Petal.Length": 1.3,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 4.9,
##     "Sepal.Width": 3.6,
##     "Petal.Length": 1.4,
##     "Petal.Width": 0.1,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 4.4,
##     "Sepal.Width": 3,
##     "Petal.Length": 1.3,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.1,
##     "Sepal.Width": 3.4,
##     "Petal.Length": 1.5,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5,
##     "Sepal.Width": 3.5,
##     "Petal.Length": 1.3,
##     "Petal.Width": 0.3,
##     "Species": "setosa"
##   },
##   {
```

```
##     "Sepal.Length": 4.5,
##     "Sepal.Width": 2.3,
##     "Petal.Length": 1.3,
##     "Petal.Width": 0.3,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 4.4,
##     "Sepal.Width": 3.2,
##     "Petal.Length": 1.3,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5,
##     "Sepal.Width": 3.5,
##     "Petal.Length": 1.6,
##     "Petal.Width": 0.6,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.1,
##     "Sepal.Width": 3.8,
##     "Petal.Length": 1.9,
##     "Petal.Width": 0.4,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 4.8,
##     "Sepal.Width": 3,
##     "Petal.Length": 1.4,
##     "Petal.Width": 0.3,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5.1,
##     "Sepal.Width": 3.8,
##     "Petal.Length": 1.6,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 4.6,
##     "Sepal.Width": 3.2,
##     "Petal.Length": 1.4,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
```

```
##   {
##     "Sepal.Length": 5.3,
##     "Sepal.Width": 3.7,
##     "Petal.Length": 1.5,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 5,
##     "Sepal.Width": 3.3,
##     "Petal.Length": 1.4,
##     "Petal.Width": 0.2,
##     "Species": "setosa"
##   },
##   {
##     "Sepal.Length": 7,
##     "Sepal.Width": 3.2,
##     "Petal.Length": 4.7,
##     "Petal.Width": 1.4,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.4,
##     "Sepal.Width": 3.2,
##     "Petal.Length": 4.5,
##     "Petal.Width": 1.5,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.9,
##     "Sepal.Width": 3.1,
##     "Petal.Length": 4.9,
##     "Petal.Width": 1.5,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.5,
##     "Sepal.Width": 2.3,
##     "Petal.Length": 4,
##     "Petal.Width": 1.3,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.5,
##     "Sepal.Width": 2.8,
##     "Petal.Length": 4.6,
##     "Petal.Width": 1.5,
##     "Species": "versicolor"
```

```
##   },
##   {
##     "Sepal.Length": 5.7,
##     "Sepal.Width": 2.8,
##     "Petal.Length": 4.5,
##     "Petal.Width": 1.3,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.3,
##     "Sepal.Width": 3.3,
##     "Petal.Length": 4.7,
##     "Petal.Width": 1.6,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 4.9,
##     "Sepal.Width": 2.4,
##     "Petal.Length": 3.3,
##     "Petal.Width": 1,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.6,
##     "Sepal.Width": 2.9,
##     "Petal.Length": 4.6,
##     "Petal.Width": 1.3,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.2,
##     "Sepal.Width": 2.7,
##     "Petal.Length": 3.9,
##     "Petal.Width": 1.4,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5,
##     "Sepal.Width": 2,
##     "Petal.Length": 3.5,
##     "Petal.Width": 1,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.9,
##     "Sepal.Width": 3,
##     "Petal.Length": 4.2,
##     "Petal.Width": 1.5,
```

```
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6,
##     "Sepal.Width": 2.2,
##     "Petal.Length": 4,
##     "Petal.Width": 1,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.1,
##     "Sepal.Width": 2.9,
##     "Petal.Length": 4.7,
##     "Petal.Width": 1.4,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.6,
##     "Sepal.Width": 2.9,
##     "Petal.Length": 3.6,
##     "Petal.Width": 1.3,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.7,
##     "Sepal.Width": 3.1,
##     "Petal.Length": 4.4,
##     "Petal.Width": 1.4,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.6,
##     "Sepal.Width": 3,
##     "Petal.Length": 4.5,
##     "Petal.Width": 1.5,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.8,
##     "Sepal.Width": 2.7,
##     "Petal.Length": 4.1,
##     "Petal.Width": 1,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.2,
##     "Sepal.Width": 2.2,
##     "Petal.Length": 4.5,
```

```
##     "Petal.Width": 1.5,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.6,
##     "Sepal.Width": 2.5,
##     "Petal.Length": 3.9,
##     "Petal.Width": 1.1,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.9,
##     "Sepal.Width": 3.2,
##     "Petal.Length": 4.8,
##     "Petal.Width": 1.8,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.1,
##     "Sepal.Width": 2.8,
##     "Petal.Length": 4,
##     "Petal.Width": 1.3,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.3,
##     "Sepal.Width": 2.5,
##     "Petal.Length": 4.9,
##     "Petal.Width": 1.5,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.1,
##     "Sepal.Width": 2.8,
##     "Petal.Length": 4.7,
##     "Petal.Width": 1.2,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.4,
##     "Sepal.Width": 2.9,
##     "Petal.Length": 4.3,
##     "Petal.Width": 1.3,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.6,
##     "Sepal.Width": 3,
```

```
##     "Petal.Length": 4.4,
##     "Petal.Width": 1.4,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.8,
##     "Sepal.Width": 2.8,
##     "Petal.Length": 4.8,
##     "Petal.Width": 1.4,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.7,
##     "Sepal.Width": 3,
##     "Petal.Length": 5,
##     "Petal.Width": 1.7,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6,
##     "Sepal.Width": 2.9,
##     "Petal.Length": 4.5,
##     "Petal.Width": 1.5,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.7,
##     "Sepal.Width": 2.6,
##     "Petal.Length": 3.5,
##     "Petal.Width": 1,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.5,
##     "Sepal.Width": 2.4,
##     "Petal.Length": 3.8,
##     "Petal.Width": 1.1,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.5,
##     "Sepal.Width": 2.4,
##     "Petal.Length": 3.7,
##     "Petal.Width": 1,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.8,
```

```
##     "Sepal.Width": 2.7,
##     "Petal.Length": 3.9,
##     "Petal.Width": 1.2,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6,
##     "Sepal.Width": 2.7,
##     "Petal.Length": 5.1,
##     "Petal.Width": 1.6,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.4,
##     "Sepal.Width": 3,
##     "Petal.Length": 4.5,
##     "Petal.Width": 1.5,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6,
##     "Sepal.Width": 3.4,
##     "Petal.Length": 4.5,
##     "Petal.Width": 1.6,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.7,
##     "Sepal.Width": 3.1,
##     "Petal.Length": 4.7,
##     "Petal.Width": 1.5,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.3,
##     "Sepal.Width": 2.3,
##     "Petal.Length": 4.4,
##     "Petal.Width": 1.3,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.6,
##     "Sepal.Width": 3,
##     "Petal.Length": 4.1,
##     "Petal.Width": 1.3,
##     "Species": "versicolor"
##   },
##   {
```

```
##     "Sepal.Length": 5.5,
##     "Sepal.Width": 2.5,
##     "Petal.Length": 4,
##     "Petal.Width": 1.3,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.5,
##     "Sepal.Width": 2.6,
##     "Petal.Length": 4.4,
##     "Petal.Width": 1.2,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.1,
##     "Sepal.Width": 3,
##     "Petal.Length": 4.6,
##     "Petal.Width": 1.4,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.8,
##     "Sepal.Width": 2.6,
##     "Petal.Length": 4,
##     "Petal.Width": 1.2,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5,
##     "Sepal.Width": 2.3,
##     "Petal.Length": 3.3,
##     "Petal.Width": 1,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.6,
##     "Sepal.Width": 2.7,
##     "Petal.Length": 4.2,
##     "Petal.Width": 1.3,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.7,
##     "Sepal.Width": 3,
##     "Petal.Length": 4.2,
##     "Petal.Width": 1.2,
##     "Species": "versicolor"
##   },
```

```
##   {
##     "Sepal.Length": 5.7,
##     "Sepal.Width": 2.9,
##     "Petal.Length": 4.2,
##     "Petal.Width": 1.3,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.2,
##     "Sepal.Width": 2.9,
##     "Petal.Length": 4.3,
##     "Petal.Width": 1.3,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.1,
##     "Sepal.Width": 2.5,
##     "Petal.Length": 3,
##     "Petal.Width": 1.1,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 5.7,
##     "Sepal.Width": 2.8,
##     "Petal.Length": 4.1,
##     "Petal.Width": 1.3,
##     "Species": "versicolor"
##   },
##   {
##     "Sepal.Length": 6.3,
##     "Sepal.Width": 3.3,
##     "Petal.Length": 6,
##     "Petal.Width": 2.5,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 5.8,
##     "Sepal.Width": 2.7,
##     "Petal.Length": 5.1,
##     "Petal.Width": 1.9,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 7.1,
##     "Sepal.Width": 3,
##     "Petal.Length": 5.9,
##     "Petal.Width": 2.1,
##     "Species": "virginica"
```

```
##   },
##   {
##     "Sepal.Length": 6.3,
##     "Sepal.Width": 2.9,
##     "Petal.Length": 5.6,
##     "Petal.Width": 1.8,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.5,
##     "Sepal.Width": 3,
##     "Petal.Length": 5.8,
##     "Petal.Width": 2.2,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 7.6,
##     "Sepal.Width": 3,
##     "Petal.Length": 6.6,
##     "Petal.Width": 2.1,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 4.9,
##     "Sepal.Width": 2.5,
##     "Petal.Length": 4.5,
##     "Petal.Width": 1.7,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 7.3,
##     "Sepal.Width": 2.9,
##     "Petal.Length": 6.3,
##     "Petal.Width": 1.8,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.7,
##     "Sepal.Width": 2.5,
##     "Petal.Length": 5.8,
##     "Petal.Width": 1.8,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 7.2,
##     "Sepal.Width": 3.6,
##     "Petal.Length": 6.1,
##     "Petal.Width": 2.5,
```

```
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.5,
##     "Sepal.Width": 3.2,
##     "Petal.Length": 5.1,
##     "Petal.Width": 2,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.4,
##     "Sepal.Width": 2.7,
##     "Petal.Length": 5.3,
##     "Petal.Width": 1.9,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.8,
##     "Sepal.Width": 3,
##     "Petal.Length": 5.5,
##     "Petal.Width": 2.1,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 5.7,
##     "Sepal.Width": 2.5,
##     "Petal.Length": 5,
##     "Petal.Width": 2,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 5.8,
##     "Sepal.Width": 2.8,
##     "Petal.Length": 5.1,
##     "Petal.Width": 2.4,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.4,
##     "Sepal.Width": 3.2,
##     "Petal.Length": 5.3,
##     "Petal.Width": 2.3,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.5,
##     "Sepal.Width": 3,
##     "Petal.Length": 5.5,
```

```
##     "Petal.Width": 1.8,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 7.7,
##     "Sepal.Width": 3.8,
##     "Petal.Length": 6.7,
##     "Petal.Width": 2.2,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 7.7,
##     "Sepal.Width": 2.6,
##     "Petal.Length": 6.9,
##     "Petal.Width": 2.3,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6,
##     "Sepal.Width": 2.2,
##     "Petal.Length": 5,
##     "Petal.Width": 1.5,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.9,
##     "Sepal.Width": 3.2,
##     "Petal.Length": 5.7,
##     "Petal.Width": 2.3,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 5.6,
##     "Sepal.Width": 2.8,
##     "Petal.Length": 4.9,
##     "Petal.Width": 2,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 7.7,
##     "Sepal.Width": 2.8,
##     "Petal.Length": 6.7,
##     "Petal.Width": 2,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.3,
##     "Sepal.Width": 2.7,
```

```
##     "Petal.Length": 4.9,
##     "Petal.Width": 1.8,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.7,
##     "Sepal.Width": 3.3,
##     "Petal.Length": 5.7,
##     "Petal.Width": 2.1,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 7.2,
##     "Sepal.Width": 3.2,
##     "Petal.Length": 6,
##     "Petal.Width": 1.8,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.2,
##     "Sepal.Width": 2.8,
##     "Petal.Length": 4.8,
##     "Petal.Width": 1.8,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.1,
##     "Sepal.Width": 3,
##     "Petal.Length": 4.9,
##     "Petal.Width": 1.8,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.4,
##     "Sepal.Width": 2.8,
##     "Petal.Length": 5.6,
##     "Petal.Width": 2.1,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 7.2,
##     "Sepal.Width": 3,
##     "Petal.Length": 5.8,
##     "Petal.Width": 1.6,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 7.4,
```

```
##     "Sepal.Width": 2.8,
##     "Petal.Length": 6.1,
##     "Petal.Width": 1.9,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 7.9,
##     "Sepal.Width": 3.8,
##     "Petal.Length": 6.4,
##     "Petal.Width": 2,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.4,
##     "Sepal.Width": 2.8,
##     "Petal.Length": 5.6,
##     "Petal.Width": 2.2,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.3,
##     "Sepal.Width": 2.8,
##     "Petal.Length": 5.1,
##     "Petal.Width": 1.5,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.1,
##     "Sepal.Width": 2.6,
##     "Petal.Length": 5.6,
##     "Petal.Width": 1.4,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 7.7,
##     "Sepal.Width": 3,
##     "Petal.Length": 6.1,
##     "Petal.Width": 2.3,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.3,
##     "Sepal.Width": 3.4,
##     "Petal.Length": 5.6,
##     "Petal.Width": 2.4,
##     "Species": "virginica"
##   },
##   {
```

```
##     "Sepal.Length": 6.4,
##     "Sepal.Width": 3.1,
##     "Petal.Length": 5.5,
##     "Petal.Width": 1.8,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6,
##     "Sepal.Width": 3,
##     "Petal.Length": 4.8,
##     "Petal.Width": 1.8,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.9,
##     "Sepal.Width": 3.1,
##     "Petal.Length": 5.4,
##     "Petal.Width": 2.1,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.7,
##     "Sepal.Width": 3.1,
##     "Petal.Length": 5.6,
##     "Petal.Width": 2.4,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.9,
##     "Sepal.Width": 3.1,
##     "Petal.Length": 5.1,
##     "Petal.Width": 2.3,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 5.8,
##     "Sepal.Width": 2.7,
##     "Petal.Length": 5.1,
##     "Petal.Width": 1.9,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.8,
##     "Sepal.Width": 3.2,
##     "Petal.Length": 5.9,
##     "Petal.Width": 2.3,
##     "Species": "virginica"
##   },
```

```
##   {
##     "Sepal.Length": 6.7,
##     "Sepal.Width": 3.3,
##     "Petal.Length": 5.7,
##     "Petal.Width": 2.5,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.7,
##     "Sepal.Width": 3,
##     "Petal.Length": 5.2,
##     "Petal.Width": 2.3,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.3,
##     "Sepal.Width": 2.5,
##     "Petal.Length": 5,
##     "Petal.Width": 1.9,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.5,
##     "Sepal.Width": 3,
##     "Petal.Length": 5.2,
##     "Petal.Width": 2,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 6.2,
##     "Sepal.Width": 3.4,
##     "Petal.Length": 5.4,
##     "Petal.Width": 2.3,
##     "Species": "virginica"
##   },
##   {
##     "Sepal.Length": 5.9,
##     "Sepal.Width": 3,
##     "Petal.Length": 5.1,
##     "Petal.Width": 1.8,
##     "Species": "virginica"
##   }
## ]
```

```r
iris2 <- fromJSON(myjson)
head(iris2)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
```

```
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

The data.table Package

- data.table
  - Inherets from data.frame
    - All functions that accept data.frame work on data.table
  - Written in C so it is much faster
  - Much, much faster at subsetting, group, and updating variables
  - A little bit new syntax

```
library(data.table)
DF <- data.frame(x=rnorm(9), y=rep(c("a", "b", "c"), each = 3), z = rnorm(9))
head(DF,3)
```

```
##             x y           z
## 1 -0.03192439 a -0.14476061
## 2  0.50662370 a -0.01257955
## 3 -0.52391211 a  0.92812159
```

```
DT <- data.table(x=rnorm(9), y=rep(c("a", "b", "c"), each = 3), z = rnorm(9))
head(DT,3)
```

```
##             x y          z
## 1: -0.8113566 a 0.17650763
## 2:  1.2194702 a 1.05538574
## 3:  0.2718714 a 0.09038212
```

```
#See all the data tables in memory
tables
```

```
## function (mb = TRUE, order.col = "NAME", width = 80, env = parent.frame(),
##     silent = FALSE, index = FALSE)
## {
##     all_obj = objects(envir = env, all.names = TRUE)
##     is_DT = which(vapply_1b(all_obj, function(x) is.data.table(get(x,
##         envir = env))))
##     if (!length(is_DT)) {
##         if (!silent)
##             cat("No objects of class data.table exist in", if (identical(env,
##                 .GlobalEnv))
##                 ".GlobalEnv"
##             else format(env), "\n")
```

```
##         return(invisible(data.table(NULL)))
##     }
##     DT_names = all_obj[is_DT]
##     info = rbindlist(lapply(DT_names, function(dt_n) {
##         DT = get(dt_n, envir = env)
##         data.table(NAME = dt_n, NROW = nrow(DT), NCOL = ncol(DT),
##             MB = if (mb)
##                 round(as.numeric(object.size(DT))/1024^2), COLS = list(names(DT)),
##             KEY = list(key(DT)), INDICES = if (index)
##                 list(indices(DT)))
##     }))
##     if (!order.col %chin% names(info))
##         stop("order.col='", order.col, "' not a column name of info")
##     info = info[base::order(info[[order.col]])]
##     if (!silent) {
##         pretty_format = function(x, width) {
##             format(prettyNum(x, big.mark = ","), width = width,
##                 justify = "right")
##         }
##         tt = copy(info)
##         tt[, `:=`(NROW, pretty_format(NROW, width = 4L))]
##         tt[, `:=`(NCOL, pretty_format(NCOL, width = 4L))]
##         if (mb)
##             tt[, `:=`(MB, pretty_format(MB, width = 2L))]
##         print(tt, class = FALSE, nrows = Inf)
##         if (mb)
##             cat("Total: ", prettyNum(sum(info$MB), big.mark = ","),
##                 "MB\n", sep = "")
##     }
##     invisible(info)
## }
## <bytecode: 0x55f57d05ec68>
## <environment: namespace:data.table>
```

```
#Subsetting Datat.table
DT[2,]
```

```
##          x y        z
## 1: 1.21947 a 1.055386
```

```
DT[DT$y=="a",]
```

```
##             x y          z
## 1: -0.8113566 a 0.17650763
## 2:  1.2194702 a 1.05538574
## 3:  0.2718714 a 0.09038212
```

```
#If no comma, Data Tables will subset by row
DT[c(2,3)]
```

```
##              x y          z
## 1: 1.2194702 a 1.05538574
## 2: 0.2718714 a 0.09038212
```

```
#Subsetting columns doesn't work the same
DT[,c(2,3)]
```

```
##    y           z
## 1: a  0.17650763
## 2: a  1.05538574
## 3: a  0.09038212
## 4: b  1.37279174
## 5: b  0.40354546
## 6: b -0.48100291
## 7: c -0.56201923
## 8: c  0.22968128
## 9: c -2.48769417
```

- Column subsetting in data.table
  - The subsetting sunction is modified for data.table
  - The argument you pass after the comma is called an "expression"
  - In R an expression is a collection of statements enclosed in curley brakets

```
{
  x <- 1
  y <- 2
}
k <- {print(10); 5}
```

```
## [1] 10
```

```
print(k)
```

```
## [1] 5
```

- Calculating calues for variables with expressions

```
DT[,list(mean(x), sum(z))] #Returns mean of x values and sum of Z values
```

```
##             V1         V2
## 1: -0.3750109 -0.2024223
```

```
DT[,table(y)]
```

```
## y
## a b c
## 3 3 3
```

- Adding new columns

```
DT[,w:=z^2]
```

```
DT2 <- DT #Incorrect
DT2 <- copy(DT) #Correct
```

- Multiple-step operations

```
DT[,m:= {tmp <-  (x+z); log2(tmp+5)}]
```

- plyr like operations

```
DT[,a:=x>0]

DT[,b:= mean(x+w), by=a]
```

- Special variables
  - .N An integer, length 1, containing the number of times a particular groups appears

```
set.seed(123);
DT <- data.table(x=sample(letters[1:3], 1E5, TRUE))
DT[, .N, by = x]#.N indicates to count
```

```
##    x     N
## 1: c 33294
## 2: b 33305
## 3: a 33401
```

- Keys
  - A unique aspect of `data.tables`
  - Able to sort and subset more rapiddly than a data.frame

```
DT <- data.table(x=rep(c("a", "b", "c"),each=100), y=rnorm(300))
setkey(DT,x)
DT['a']#Finds all values of `x` that are == to 'a'
```

```
##      x           y
##   1: a  0.88631257
##   2: a  2.82858132
##   3: a  2.03145429
##   4: a  1.90675413
##   5: a  0.21490826
##   6: a -0.86273413
##   7: a -2.20493863
##   8: a  0.24105923
##   9: a  1.83832419
##  10: a  0.79205468
##  11: a  0.65053469
##  12: a -1.53912061
##  13: a -0.60830053
##  14: a  0.38195644
##  15: a -1.07500044
##  16: a  0.21994264
##  17: a -0.78288781
```

```
##  18: a -1.11003346
##  19: a -1.65871456
##  20: a -0.50147343
##  21: a  1.91636375
##  22: a  1.41236645
##  23: a  0.92260986
##  24: a  1.01106201
##  25: a  0.57213026
##  26: a -0.62843126
##  27: a -0.36316140
##  28: a -1.05858811
##  29: a -0.42935803
##  30: a  0.86941467
##  31: a -0.54001647
##  32: a -1.14647747
##  33: a -0.17151840
##  34: a -0.56368340
##  35: a -0.42994346
##  36: a -1.23723779
##  37: a  0.15901329
##  38: a -1.16711067
##  39: a -0.08111944
##  40: a -0.51667953
##  41: a  0.99540703
##  42: a  0.79752142
##  43: a  0.53895224
##  44: a -1.40405605
##  45: a  0.40144065
##  46: a -0.52432237
##  47: a -0.83952146
##  48: a  0.47556591
##  49: a -0.01194696
##  50: a  0.10319780
##  51: a -0.38575415
##  52: a  1.11726438
##  53: a -0.49961390
##  54: a -0.44735091
##  55: a -0.23784512
##  56: a -0.86939374
##  57: a  1.14887678
##  58: a  0.53864996
##  59: a -0.10680992
##  60: a  0.60053649
##  61: a -1.47499445
##  62: a  0.98126964
##  63: a -0.61118738
##  64: a  0.08938648
##  65: a -0.01327227
```

```
##  66: a -0.97219341
##  67: a -0.57946225
##  68: a  0.14963144
##  69: a  0.47640689
##  70: a  0.44729682
##  71: a -0.19180956
##  72: a  0.51712710
##  73: a  0.40338273
##  74: a  1.78411385
##  75: a  0.27775645
##  76: a  0.77394978
##  77: a -2.08081928
##  78: a -0.35920889
##  79: a -0.45932217
##  80: a  0.20181947
##  81: a  0.62401138
##  82: a -0.25722981
##  83: a  0.94414021
##  84: a  0.25074808
##  85: a -0.72784257
##  86: a  0.36881323
##  87: a  0.44415068
##  88: a -1.00535422
##  89: a -0.33152471
##  90: a -0.37039325
##  91: a -0.79701529
##  92: a  0.28148559
##  93: a  0.33307250
##  94: a  0.52690325
##  95: a -0.78168949
##  96: a -0.02793948
##  97: a -1.74492339
##  98: a  0.65284209
##  99: a -0.93830821
## 100: a  0.62753159
##      x           y
```

- Keys can also be used to facilitate joining data.tables

```
DT1 <- data.table(x=c('a', 'a', 'b', 'dt1'), y = 1:4)
DT2 <- data.table(x=c('a', 'b', 'dt2'), z=5:7)
setkey(DT1, x)
setkey(DT2, x)

merge(DT1, DT2)
```

```
##    x y z
## 1: a 1 5
## 2: a 2 5
```

```
## 3: b 3 6
```

- Also helpful for quickly reading from the disk

```r
big_df <- data.frame(x=rnorm(1E6), y=rnorm(1E6))
file <- tempfile()
write.table(big_df, file=file, row.names=FALSE, col.names = TRUE, sep="\t", quote=FALSE)
system.time(fread(file))
```

```
##    user  system elapsed
##   0.120   0.012   0.070
```

```r
system.time(read.table(file, header=TRUE, sep="\t"))
```

```
##    user  system elapsed
##   5.133   0.094   5.261
```

Quiz Scribbles

```r
#1
URL <- "https://d396qusza40orc.cloudfront.net/getdata%2Fdata%2Fss06hid.csv"
download.file(URL, destfile = "./Idaho_Housing.csv", method = "curl")
ID <- read.csv("Idaho_Housing.csv")
sum(!is.na(ID[ID$VAL==24,"VAL"]))
```

```
## [1] 53
```

```r
#2
URL <- "https://d396qusza40orc.cloudfront.net/getdata%2Fdata%2FDATA.gov_NGAP.xlsx"
filename <- paste(getwd(), "/Q2Data.xlsx", sep = "")
download.file(URL, destfile = filename, method = "curl")

library(xlsx)
dat <- read.xlsx(filename, sheetIndex = 1, rowIndex = 18:23, colIndex = 7:15)
sum(dat$Zip*dat$Ext, na.rm = TRUE)
```

```
## [1] 36534720
```

```r
#4
URL <- "https://d396qusza40orc.cloudfront.net/getdata%2Fdata%2Frestaurants.xml"
filename <- paste(getwd(), "/Q4Data.xml", sep = "")
download.file(URL, filename, "curl")

library(XML)
dat <- xmlTreeParse(filename, useInternalNodes = T)
rootNode <- xmlRoot(dat)

ZIPs <- xpathSApply(rootNode, "//zipcode", xmlValue)
qualify <- ZIPs == "21231"
sum(qualify)
```

```
## [1] 127
```

```r
#5
URL <- "https://d396qusza40orc.cloudfront.net/getdata%2Fdata%2Fss06pid.csv"
filename <- paste(getwd(), "/Q5Data", sep = "")
download.file(URL, filename, "curl")

library(data.table)
DT <- fread(filename)

# print("tapply - False Winner")
# tapply(DT$pwgtp15, DT$SEX, mean)
# system.time(tapply(DT$pwgtp15, DT$SEX, mean))

print("mean(by)")
```

```
## [1] "mean(by)"
```

```r
mean(DT$pwgtp15, by = DT$SEX)
```

```
## [1] 98.21613
```

```r
system.time(mean(DT$pwgtp15, by = DT$SEX))
```

```
##    user  system elapsed
##   0.000   0.000   0.001
```

```r
# print("sapply - False Winner")
# sapply(split(DT$pwgtp15, DT$SEX), mean)
# system.time(sapply(split(DT$pwgtp15, DT$SEX), mean))



print("DT[,...] - apparently this is the fastest")
```

```
## [1] "DT[,...] - apparently this is the fastest"
```

```r
DT[,mean(pwgtp15), by = SEX]
```

```
##    SEX       V1
## 1:   1 99.80667
## 2:   2 96.66534
```

```r
system.time(DT[,mean(pwgtp15), by = SEX])
```

```
##    user  system elapsed
##   0.002   0.000   0.002
```

```r
 print("mean")
```

```
## [1] "mean"
```

```r
{mean(DT[DT$SEX==1,]$pwgtp15); mean(DT[DT$SEX==2,]$pwgtp15)}
```

```
## [1] 96.66534
```

```r
system.time({mean(DT[DT$SEX==1,]$pwgtp15); mean(DT[DT$SEX==2,]$pwgtp15)})
```

```
##    user  system elapsed
##   0.009   0.000   0.006
```

```r
# print("rowMeans - Wrong")
# rowMeans(DT)
# system.time({rowMeans(DT)[DT$SEX==1]; rowMeans(DT)[DT$SEX==2]})
```

Data Storage Sytems and Extracting Data From Web or Databases

>>

Reading from MySQL

- Overview
  - Free and widelt used open source database software
  - Widely used in internet based applications
  - Data are structured in:
    - Databases
    - Tables within databases (Dataset)
    - Fields within tables (Col fo dataset)
  - Each row is called a record
- Further Reading
  - **(Wikipedia)[http://en.wikipedia.org/wiki/MySQL]**
  - **(Documentation)[http://www.mysql.com/]**
- **(Example structure)[http://dev.mysql.com/doc/employee/en/sakila-structure.html]**
- **(How to install)[http://dev.mysql.com/doc/refman/5.7/en/installing.html]**
- **(UCSC database example)[http://genome.ucsc.edu]**
  - `dbConnect()` - used to make a connection to a database (SQL or otherwise)

```r
library("RMySQL")
```

```
## Loading required package: DBI
```

```r
ucscDb <- dbConnect(MySQL(), user = "genome",
                    host = "genome-mysql.cse.ucsc.edu")
result <- dbGetQuery(ucscDb, "show databases;")
#^2nd param is a MySQL cmd that we're sending to the ucscDb database using the dbGetQuery fn

dbDisconnect(ucscDb) #Important to disconnect from a SQL server, returns logical
```

```
## [1] TRUE
```

```r
result
```

```
##               Database
## 1               acaChl1
## 2               ailMel1
## 3               allMis1
## 4               allSin1
## 5               amaVit1
## 6               anaPla1
## 7               ancCey1
## 8               angJap1
## 9               anoCar1
## 10              anoCar2
## 11              anoGam1
## 12              anoGam3
## 13              apaSpi1
## 14              apaVit1
## 15              apiMel1
## 16              apiMel2
## 17              aplCal1
## 18              aptFor1
## 19              aptMan1
## 20              aquChr2
## 21              araMac1
## 22              ascSuu1
## 23              balAcu1
## 24              balPav1
## 25              bisBis1
## 26              bosTau2
## 27              bosTau3
## 28              bosTau4
## 29              bosTau5
## 30              bosTau6
## 31              bosTau7
## 32              bosTau8
## 33              bosTau9
## 34            bosTauMd3
## 35              braFlo1
## 36              bruMal2
```

```
## 37              bucRhi1
## 38              burXyl1
## 39              caeAng2
## 40              caeJap1
## 41              caeJap4
## 42               caePb1
## 43               caePb2
## 44               caePb3
## 45              caeRem2
## 46              caeRem3
## 47              caeRem4
## 48             caeSp111
## 49              caeSp51
## 50              calAnn1
## 51              calJac1
## 52              calJac3
## 53              calMil1
## 54              canFam1
## 55              canFam2
## 56              canFam3
## 57              capCar1
## 58              carCri1
## 59              cavPor3
## 60                  cb1
## 61                  cb3
## 62                  cb4
## 63                 ce10
## 64                 ce11
## 65                  ce2
## 66                  ce4
## 67                  ce6
## 68              cerSim1
## 69              chaVoc2
## 70              cheMyd1
## 71              chlSab2
## 72              chlUnd1
## 73              choHof1
## 74              chrPic1
## 75              chrPic2
## 76                  ci1
## 77                  ci2
## 78                  ci3
## 79              colLiv1
## 80              colStr1
## 81              corBra1
## 82              corCor1
## 83              cotJap2
## 84              criGri1
```

```
## 85          criGriChoV1
## 86          criGriChoV2
## 87              cucCan1
## 88              danRer1
## 89             danRer10
## 90             danRer11
## 91              danRer2
## 92              danRer3
## 93              danRer4
## 94              danRer5
## 95              danRer6
## 96              danRer7
## 97              dasNov3
## 98              dipOrd1
## 99              dirImm1
## 100                 dm1
## 101                 dm2
## 102                 dm3
## 103                 dm6
## 104                 dp2
## 105                 dp3
## 106             droAna1
## 107             droAna2
## 108             droEre1
## 109             droGri1
## 110             droMoj1
## 111             droMoj2
## 112             droPer1
## 113             droSec1
## 114             droSim1
## 115             droSim2
## 116             droVir1
## 117             droVir2
## 118             droYak1
## 119             droYak2
## 120             eboVir3
## 121             echTel1
## 122             echTel2
## 123             egrGar1
## 124             equCab1
## 125             equCab2
## 126             equCab3
## 127             eriEur1
## 128             eriEur2
## 129             eurHel1
## 130             falChe1
## 131             falPer1
## 132             felCat3
```

```
## 133            felCat4
## 134            felCat5
## 135            felCat8
## 136            felCat9
## 137            ficAlb2
## 138                fr1
## 139                fr2
## 140                fr3
## 141            fulGla1
## 142            gadMor1
## 143            galGal2
## 144            galGal3
## 145            galGal4
## 146            galGal5
## 147            galGal6
## 148            galVar1
## 149            gasAcu1
## 150            gavSte1
## 151             gbMeta
## 152            geoFor1
## 153                 go
## 154           go080130
## 155           go140213
## 156           go150121
## 157           go180426
## 158            gorGor3
## 159            gorGor4
## 160            gorGor5
## 161            haeCon2
## 162            halAlb1
## 163            halLeu1
## 164            hetBac1
## 165            hetGla1
## 166            hetGla2
## 167               hg16
## 168               hg17
## 169               hg18
## 170               hg19
## 171        hg19Patch10
## 172        hg19Patch13
## 173               hg38
## 174        hg38Patch11
## 175            hgFixed
## 176          hgcentral
## 177 information_schema
## 178            latCha1
## 179            lepDis1
## 180            letCam1
```

```
## 181            loaLoa1
## 182            loxAfr3
## 183            macEug1
## 184            macEug2
## 185            macFas5
## 186            manPen1
## 187            melGal1
## 188            melGal5
## 189            melHap1
## 190            melInc2
## 191            melUnd1
## 192            merNub1
## 193            mesUni1
## 194            micMur1
## 195            micMur2
## 196               mm10
## 197         mm10Patch4
## 198                mm5
## 199                mm6
## 200                mm7
## 201                mm8
## 202                mm9
## 203            monDom1
## 204            monDom4
## 205            monDom5
## 206            musFur1
## 207            myoLuc2
## 208            nanPar1
## 209            nasLar1
## 210            necAme1
## 211            nipNip1
## 212            nomLeu1
## 213            nomLeu2
## 214            nomLeu3
## 215            ochPri2
## 216            ochPri3
## 217            oncVol1
## 218            opiHoa1
## 219            oreNil1
## 220            oreNil2
## 221            oreNil3
## 222            ornAna1
## 223            ornAna2
## 224            oryCun2
## 225            oryLat2
## 226            otoGar3
## 227            oviAri1
## 228            oviAri3
```

```
## 229            oviAri4
## 230            panPan1
## 231            panPan2
## 232            panRed1
## 233            panTro1
## 234            panTro2
## 235            panTro3
## 236            panTro4
## 237            panTro5
## 238            panTro6
## 239            papAnu2
## 240            papAnu4
## 241            papHam1
## 242            pelCri1
## 243            pelSin1
## 244 performance_schema
## 245            petMar1
## 246            petMar2
## 247            petMar3
## 248            phaCar1
## 249            phaLep1
## 250            phoRub1
## 251            picPub1
## 252            ponAbe2
## 253            ponAbe3
## 254            priExs1
## 255            priPac1
## 256            priPac3
## 257            proCap1
## 258     proteins120806
## 259     proteins121210
## 260     proteins140122
## 261     proteins150225
## 262     proteins160229
## 263     proteins180404
## 264           proteome
## 265            pteGut1
## 266            pteVam1
## 267            pygAde1
## 268            pytBiv1
## 269            rheMac1
## 270           rheMac10
## 271            rheMac2
## 272            rheMac3
## 273            rheMac8
## 274            rhiRox1
## 275                rn3
## 276                rn4
```

```
## 277               rn5
## 278               rn6
## 279           sacCer1
## 280           sacCer2
## 281           sacCer3
## 282           saiBol1
## 283           sarHar1
## 284           serCan1
## 285           sorAra1
## 286           sorAra2
## 287          sp120323
## 288          sp121210
## 289          sp140122
## 290          sp150225
## 291          sp160229
## 292          sp180404
## 293           speTri2
## 294           strCam1
## 295           strPur1
## 296           strPur2
## 297           strRat2
## 298          susScr11
## 299           susScr2
## 300           susScr3
## 301           taeGut1
## 302           taeGut2
## 303           tarSyr1
## 304           tarSyr2
## 305           tauEry1
## 306           tetNig1
## 307           tetNig2
## 308           thaSir1
## 309           tinGut2
## 310           triMan1
## 311           triSpi1
## 312           triSui1
## 313           tupBel1
## 314           turTru2
## 315           tytAlb1
## 316           uniProt
## 317           vicPac1
## 318           vicPac2
## 319          visiGene
## 320           wuhCor1
## 321           xenLae2
## 322           xenTro1
## 323           xenTro2
## 324           xenTro3
```

```
## 325              xenTro7
## 326              xenTro9
## 327              zonAlb1
```

- Connecting to hg19 (particular build of human genome) and listing tables
  - When using `dbConnect` we pass botht he mysql server and particular database we wish to connect with + Remember, each table is simular to a data.frame

```
hg19 <- dbConnect(MySQL(), user = "genome", db = "hg19",
                  host = "genome-mysql.cse.ucsc.edu")
allTables <- dbListTables(hg19)
length(allTables)
```

```
## [1] 12444
```

```
allTables[1:5]
```

```
## [1] "HInv"         "HInvGeneMrna" "acembly"      "acemblyClass" "acemblyPep"
```

- Get dimensions of a specific table
  - `affyU133Plus2` is a measurement technology for measuring something about the genome
  - Remember: Fields are similar to a colName within a data.frame

```
dbListFields(hg19, "affyU133Plus2")
```

```
##  [1] "bin"         "matches"     "misMatches"  "repMatches"  "nCount"
##  [6] "qNumInsert"  "qBaseInsert" "tNumInsert"  "tBaseInsert" "strand"
## [11] "qName"       "qSize"       "qStart"      "qEnd"        "tName"
## [16] "tSize"       "tStart"      "tEnd"        "blockCount"  "blockSizes"
## [21] "qStarts"     "tStarts"
```

```
dbGetQuery(hg19, "select count(*) from affyU133Plus2")
```

```
##   count(*)
## 1    58463
```

```
#^"select count(*) from XXX" instructs mysql to return the num of fields in XXX

## Read from the table
affyData <- dbReadTable (hg19, "affyU133Plus2")
```

```
## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 0 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 1 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 2 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 3 imported as
## numeric
```

```
## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 4 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 5 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 6 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 7 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 8 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 11 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 12 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 13 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 15 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 16 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 17 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 18 imported as
## numeric
```

```r
head(affyData)
```

```
##   bin matches misMatches repMatches nCount qNumInsert qBaseInsert tNumInsert
## 1 585     530          4          0     23          3          41          3
## 2 585    3355         17          0    109          9          67          9
## 3 585    4156         14          0     83         16          18          2
## 4 585    4667          9          0     68         21          42          3
## 5 585    5180         14          0    167         10          38          1
## 6 585     468          5          0     14          0           0          0
##   tBaseInsert strand        qName qSize qStart qEnd tName     tSize tStart
## 1         898      -  225995_x_at   637      5  603  chr1 249250621  14361
## 2       11621      -  225035_x_at  3635      0 3548  chr1 249250621  14381
## 3          93      -  226340_x_at  4318      3 4274  chr1 249250621  14399
## 4        5743      - 1557034_s_at  4834     48 4834  chr1 249250621  14406
## 5          29      -    231811_at  5399      0 5399  chr1 249250621  19688
## 6           0      -    236841_at   487      0  487  chr1 249250621  27542
##    tEnd blockCount
## 1 15816          5
```

```
## 2 29483         17
## 3 18745         18
## 4 24893         23
## 5 25078         11
## 6 28029          1
##                                                                  blockSizes
## 1                                                         93,144,229,70,21,
## 2              73,375,71,165,303,360,198,661,201,1,260,250,74,73,98,155,163,
## 3                 690,10,32,33,376,4,5,15,5,11,7,41,277,859,141,51,443,1253,
## 4 99,352,286,24,49,14,6,5,8,149,14,44,98,12,10,355,837,59,8,1500,133,624,58,
## 5                                    131,26,1300,6,4,11,4,7,358,3359,155,
## 6                                                                      487,
##
## 1                                                                            34,132
## 2                       87,165,540,647,818,1123,1484,1682,2343,2545,2546,2808,3058,3133,320
## 3                  44,735,746,779,813,1190,1195,1201,1217,1223,1235,1243,1285,1564,2423,256
## 4 0,99,452,739,764,814,829,836,842,851,1001,1016,1061,1160,1173,1184,1540,2381,2441,2450,395
## 5                                                 0,132,159,1460,1467,1472,1484,1489,149
## 6
##
## 1
## 2                                  14381,14454,14969,15075,15240,15543,15903,16104,16853
## 3                            14399,15089,15099,15131,15164,15540,15544,15549,15564,15569
## 4 14406,20227,20579,20865,20889,20938,20952,20958,20963,20971,21120,21134,21178,21276,21288
## 5                                                                     19688,19819,19845
## 6
```

- Select a specific subset
  - select *all observations* from *this table* where *colName (are)* between *desired range*

```
query <- dbSendQuery(hg19, "select * from affyU133Plus2 where misMatches between 1 and 3")
```

```
## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 0 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 1 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 2 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 3 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 4 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 5 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 6 imported as
## numeric
```

```
## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 7 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 8 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 11 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 12 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 13 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 15 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 16 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 17 imported as
## numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 18 imported as
## numeric
```

```
affyMis <- fetch(query)
quantile (affyMis$misMatches)
```

```
##   0%  25%  50%  75% 100%
##    1    1    2    2    3
```

```
##Select only the first bit of data:
affyMisSmall <- fetch(query, n = 10)
dbClearResult(query)
```

```
## [1] TRUE
```

```
#^This will clear the query that is sitting in the MySQL server, returns logical

dim(affyMisSmall)#To see the small subset we just selected
```

```
## [1] 10 22
```

- More queries are in the MySQL documentation
- ALWAYS remember to close the connection

```
dbDisconnect(hg19)
```

```
## [1] TRUE
```

- Further resources

    - **(RMySQL vignette)[http://cran.r-project.org/web/paclages/RMySQL/RMySQL.pdf]**
    - **(List of commands)[http://www.pantz.org/software/mysql/mysqlcommands.html]**
        * **DO NOT:** delete, add or join things; Only select (Unless your intent is to update the server)
        * In general be careful with mysql commands, because you can delete data that others are working on.
    - **(A nice blog post summarizing some other comamnds)[http://www.r-bloggers.com/mysql-and-r/]**

Reading from HDF5

- Heirarchical Data Format
    - Used for storing large data sets
    - Supports storing a range of data types
    - Optimizes reading and writing to the disk in R
- Data is stored in *groups* containing zero or more data sets and metadata. Each group has:
    - a *group header* with group name and list of attributes
    - a *group symbol table* with a list of objects in the group
- *Datasets* are a multidimensional array of data elements with metadata. Can have:
    - a *header* with name, datatype, dataspace, and storage layout
    - a *data array* with the data
- Package is installed with bioconductor
    - First time you install a package with `biocLite` you'll also have to execute `install.packages("BiocManager"); BiocManager::install("rhdf5")` to load the `biocLite()` function. (Note: I had a lot of trouble with this, so additional steps may be required)
- The lecture is modeled very closely on **(the rhdf5 tutorial on bionconductor's website)[http://www.bioconductor.org/packages/release/bioc/vignettes/rhdf5/inst/doc/rhdf5.pdf]**
- The HDF group **(has information on HDF5 in general)[http://www.hdfgroup.org/HDF5/]**

```
library(rhdf5)
created <- h5createFile("example.h5")
```

```
## file '/home/phiprime/Documents/Education/R/LassoingDataNotes/example.h5' already exists.
```

```
created #TRUE if a new file was made, false if it already exists
```

```
## [1] FALSE
```

```r
created <- h5createGroup("example.h5", "foo")
```

```
## Can not create group. Object with name 'foo' already exists.
```

```r
created <- h5createGroup("example.h5", "baa")
```

```
## Can not create group. Object with name 'baa' already exists.
```

```r
created <- h5createGroup("example.h5", "foo/foobaa")
```

```
## Can not create group. Object with name 'foo/foobaa' already exists.
```

```r
h5ls("example.h5")
```

```
##         group   name       otype   dclass  dim
## 0           /    baa   H5I_GROUP
## 1           /     df H5I_DATASET COMPOUND    5
## 2           /    foo   H5I_GROUP
## 3        /foo      A H5I_DATASET  INTEGER  x 2
## 4        /foo foobaa   H5I_GROUP
## 5 /foo/foobaa      B H5I_DATASET    FLOAT  x 2
```

- Write to groups

```r
A <- matrix(1:10, nrow = 5, ncol = 2)
h5write(A, "example.h5", "foo/A")
B <- array(seq(0.1, 2.0, by = 0.1), dim = c(5, 2, 2))
attr(B, "scale") <- "liter" #Attribute gets added to metadata for colName == "B"
h5write(B, "example.h5", "foo/foobaa/B")
h5ls("example.h5")
```

```
##         group   name       otype   dclass  dim
## 0           /    baa   H5I_GROUP
## 1           /     df H5I_DATASET COMPOUND    5
## 2           /    foo   H5I_GROUP
## 3        /foo      A H5I_DATASET  INTEGER  x 2
## 4        /foo foobaa   H5I_GROUP
## 5 /foo/foobaa      B H5I_DATASET    FLOAT  x 2
```

- Write a data set

```r
df <- data.frame(1L:5L, seq(0,1, length.out = 5),
                 c("ab", "cde", "fghi", "a", "s"), stringsAsFactors = FALSE)
if(is.null(h5ls("example.h5")))
{  h5write(df, "example.h5", "df") }
h5ls("example.h5")
```

```
##         group   name       otype   dclass  dim
## 0           /    baa   H5I_GROUP
## 1           /     df H5I_DATASET COMPOUND    5
## 2           /    foo   H5I_GROUP
## 3        /foo      A H5I_DATASET  INTEGER  x 2
```

```
## 4         /foo foobaa   H5I_GROUP
## 5 /foo/foobaa      B H5I_DATASET    FLOAT  x 2
```

- Reading data

```
readA <- h5read("example.h5", "foo/A")
readB <- h5read("example.h5", "foo/foobaa/B")
readdf <- h5read("example.h5", "df")
readA
```

```
##      [,1] [,2]
## [1,]    1    6
## [2,]    2    7
## [3,]    3    8
## [4,]    4    9
## [5,]    5   10
```

- Writing and reading chunks

```
h5write(c(12,13,14), "example.h5", "foo/A", index = list(1:3,1)) #index indicates where data s
h5read("example.h5", "foo/A")
```

```
##      [,1] [,2]
## [1,]   12    6
## [2,]   13    7
## [3,]   14    8
## [4,]    4    9
## [5,]    5   10
```

Reading from The Web

- *Webscraping* - Programtically extracting data from the HTML code of websites.
  - It can be a great way to get data **How Netflix reverse engineered Hollywood**
  - Sometimes this is against the terms of service for the website

  - Attepting to read too many pages too quickly can get your IP address blocked
- Getting data off webpages - `readLines()`

```
con <- url("https://scholar.google.com/citations?user=HI-I6C0AAAAJ&hl=en&oi=ao")
htmlCode = readLines(con)
```

```
## Warning in readLines(con): incomplete final line found on 'https://
## scholar.google.com/citations?user=HI-I6C0AAAAJ&hl=en&oi=ao'
```

```
close(con)
substr(htmlCode, 1, 80)#For a preview
```

```
##  [1] "<!doctype html><html><head><title>Jeff Leek - Google Scholar Citations</title><m"
##  [2] "var ha=ba,ia=function(a,b){a.prototype=aa(b.prototype);a.prototype.constructor=a"
##  [3] "pa=qa?0<+qa[1]:r(\"Android\")?!0:window.matchMedia&&window.matchMedia(\"(pointer)\")"
##  [4] "\"\"))}return b.join(\"&\")},za=function(a,b){var c=a.elements[b];c||(c=document.cre"
```

```
##  [5] "var Ga=function(a){var b=a.b,c=b.length;a=a.m;for(var d=0,e=0;e<c;e++){var f=b[e"
##  [6] "8:0)};function Pa(a,b){return(b=b&&sa())?{passive:b,capture:a}:a}function Ia(a,b"
##  [7] "if(Sa?\"complete\"!=Ta:\"loading\"==Ta)La=new z,Sa?B(document,\"readystatechange\",Ra
##  [8] "function $a(a){a=void 0===a?!1:a;G.pop()(a)}function Ya(a,b){for(b=void 0===b?!1"
##  [9] "B(document,\"focus\",function(a){var b=G.length;if(b)for(var c=Za(a.target);c<b;){"
## [10] "var ib=function(a,b){b=db(b);a=eb(a);a=fb(a)||\"#\";gb=J(b);hb?window.history.push"
## [11] "d.substr(e+1)}else f=d,d=\"\";f&&(b[decodeURIComponent(f)]=decodeURIComponent(d))}"
## [12] "function ob(){setTimeout(function(){if(!pb){var a=window.history.state;pb=!0;gb="
## [13] "if(\"undefined\"==typeof GSP)ub=!1;else{var vb=.001*Date.now(),wb=GSP.eventId,xb=!"
## [14] "var Ib=ub;\"onpageshow\"in window?B(window,\"pageshow\",ob):D(ob);B(window,hb?\"popst
## [15] "function P(a,b,c){var d=$b;\"string\"===typeof b&&(ac[0]=b,b=ac);var e=b.length;a="
## [16] "function $b(a,b,c,d){var e=cc[c];e||(\"touchstart\"!=c&&\"mouseover\"!=c&&\"mouseout\
## [17] "function Zb(a){for(var b=a.target;b&&b!=document&&!b.disabled&&!p(b,\"gs_dis\");){"
## [18] "function Xb(a,b,c){a:{for(var d=c.currentTarget;d&&d!=document;){var e=fc(a,d);i"
## [19] "function kc(a,b,c,d){c=(d=(c=Ub[c])&&c[d])?d.length:0;for(var e=0;e<c;e++){var f"
## [20] "function pc(a,b,c,d,e){for(var f;b&&a;){if(c(b)){if(e)return b}else for(f=nc(b,d"
## [21] "m.id+\"-bdy\")||m,F=t(m.getAttribute(\"data-shd\")||\"gs_md_s\"),A=Bc(m),da=!!A&&p(A,\
## [22] "tb);Mc(a);yc.add(e);e();A&&w&&(f(),xc.addListener(f));k(l,\"gs_nscl\");l.style.top"
## [23] "v)}))};function Bc(a){a=a.parentNode;return p(a,\"gs_md_wnw\")?a:null}function Nc("
## [24] "function Ec(a,b,c){q(b,\"gs_md_ldg\",!1);for(var d=b.querySelectorAll(\"[data-duid]"
## [25] "var Mc=function(a){if(a=document.querySelector(\"#\"+a+\">.gs_md_bdy\"))a.scrollTop="
## [26] "function Lc(){return{n:Hc,p:S,h:Gc}}var T=null,Dc=\"\",R=\"\",Cc=\"\",S=\"\",Gc=\"\",
## [27] "Qc.prototype.ea=function(a){var b=this;L(a);if((a=this.w)&&!this.o){var c=\"json="
## [28] "f[w];e=void 0;var v=\"\"+(l[w]||\"\"),F=h.parentNode.querySelector(\".gs_in_txts\");q
## [29] "var Rc=function(a,b){a=a.w;var c=a.getAttribute(\"data-bsel\");a=c?document.queryS"
## [30] "a.appendChild(b);Jb()}};Yb(Qc,[new M(\".gs_ajax_frm\",{submit:Qc.prototype.ea})]);"
## [31] "D(function(){Wc=Ib?\"&bn=1\":\"\";Ib&&$c()});B(window,\"pageshow\",function(a){a.pers
## [32] "B(document,Uc,function(a){if(!(\"click\"==a.type&&a.button||\"mouseup\"==a.type&&1!="
## [33] "Wc;0<c.indexOf(\"scisig=\")?(a.setAttribute(\"href\",Vc+b),a.removeAttribute(\"data-c
## [34] "V.prototype.R=function(a){var b=a.a.keyCode;if(38==b||40==b)L(a),this.open(38==b"
## [35] "var ad=function(a,b){var c=b.currentTarget,d=t(a.ga),e=a.F();c!=e&&(d.value=c.ge"
## [36] "Yb(cd,[new M(\".gs_md_ulr\",{}),new M(\".gs_md_li\",{keydown:cd.prototype.G})]);P(\"#
## [37] "P(\"#gs_hdr_tsi\",[\"focus\",\"blur\"],function(a){function b(){var h=d.getBoundingCli
## [38] "P(\"#gs_hdr_tsc\",\"mousedown\",function(a){L(a);var b=t(\"gs_hdr_tsi\");b.value=\"\"
## [39] "var gd=function(a){a=a.a.keyCode;return 32==a||13==a},fd=function(a){O(\"gs-press"
## [40] "\"mouseup\",\"click\"],md=r(\"Android\")&&!r(\"Chrome\"),nd=0,od=0,pd=[\"touchstart\"
## [41] "function Rd(a){var b=t(\"gsc_cods_frm\");if(b){b=b.elements;var c=b[1];b[0].disabl"
## [42] "b||\"\")}a&&(Sd=+a.getAttribute(\"data-max\")||0,za(t(\"gsc_cods_save\"),\"xsrf\").val
## [43] "function Ud(a){for(var b=\"\",c=la(a.f),d=c.length;d--;)b+=a.get(c[d]);return b}va"
## [44] "P([\".gsc_ccb_add\",\".gsc_ccb_del\"],\"click\",function(a){a=a.currentTarget;if(!Fd(a
## [45] "P(\"#gsc_cod_trev\",\"click\",function(){var a=t(\"gsc_cods_res\").cloneNode(!0),b=a.c
## [46] "P([\"#gsc_coauth_opn\",\".gsc_rsb_btne\",\".gsc_rsb_btnv\"],\"click\",function(){Md(\
## [47] "var ne=window.location.href.split(\"#\")[0];ce=ne.replace(/([?&])(cstart|pagesize)"
## [48] "P(\"#gsc_bpf_more\",\"click\",function(a){var b=a.currentTarget,c=ee,d=100>c?100-c:1"
## [49] "t(\"gsc_a_b\").rows.length),e.innerHTML=f;b.disabled=!h.N}else Wd(2)})});P([\"#gsc_"
## [50] "P(\"#gsc_md_hist\",\"gs-md-lded\",function(){var a=t(\"gsc_md_hist_c\");if(!a.innerHT
## [51] "P(\"#gsc_prf_btne\",\"click\",function(){var a=t(\"gsc_md_pro-d\");a.setAttribute(\"da
## [52] "P(\".gsc_prf_tab\",\"click\",function(a){var b=t(\"gsc_bdy\");b.setAttribute(\"data-ta
```

```
## [53] "P(\"#gsc_md_cbyd\",\"gs-md-lded\",function(){var a=t(\"gsc_md_cbyd_f\"),b=t(\"gsc_md_c
## [54] "P(\"#gsc_md_cbym_e\",\"click\",function(a){a=a.currentTarget.getAttribute(\"data-href\
## [55] "P(\".gsc_a_acm\",\"click\",function(a){L(a);var b=a.currentTarget;a=b.href;var c=b.g"
## [56] "P(\"#gs_sth\",\"gs-sth-change\",function(a){var b=t(\"gsc_a_tr0\"),c=t(\"gsc_a_trh\")
## [57] "P(\"#gs_md_cita-d\",\"gs-md-lded\",function(){var a=t(\"gsc_ocd_bdy\");if(a){var b=a.g
## [58] "P(\"#gs_md_cita-b-upload\",\"click\",function(){var a=t(\"gsc_ocd_bdy\");a&&(n(a,\"gsc
## [59] "P(\".gsc_ecd_form_tsel\",\"click\",function(a){var b=t(\"gsc_ecd_table\"),c=za(t(\"gsc
## [60] "P(\"#gs_bdy\",\"gs-upload-success\",function(){var a=t(\"gsc_vcd_form\").getAttribute
## [61] "}({\"bouncePrefix\":\"http://scholar.google.com\",\"neverBounce\":!1,\"customAC\":0,\"
```

- Parsing with XML

```r
library(XML)
library(RCurl)
url <- "https://scholar.google.com/citations?user=HI-I6C0AAAAJ&hl=en&oi=ao"
html <- htmlTreeParse(getURL(url), useInternalNodes = TRUE)
xpathSApply(html, "//title", xmlValue)
```

```
## [1] "Jeff Leek - Google Scholar Citations"
```

```r
xpathSApply(html, "//td[@id='col-citedby']", xmlValue) #They changed the title and I can't fin
```

```
## list()
```

- GET fromt he httr package

```r
library(httr)
url <- "https://scholar.google.com/citations?user=HI-I6C0AAAAJ&hl=en&oi=ao"
html2 <- GET(url)
content2 <- content(html2, as = "text")
parsedHtml <- htmlParse(content2, asText = TRUE)
xpathSApply(parsedHtml, "//title", xmlValue)
```

```
## [1] "Jeff Leek - Google Scholar Citations"
```

- Accessing websites with passwords

```r
pg1 <- GET("http://httpbin.org/basic-auth/user/passwd")
pg1 #returns 401 error
```

```
## Response [http://httpbin.org/basic-auth/user/passwd]
##   Date: 2020-03-29 00:46
##   Status: 401
##   Content-Type: <unknown>
## <EMPTY BODY>
```

```r
pg2 <- GET("http://httpbin.org/basic-auth/user/passwd",
           authenticate("user", "passwd"))
pg2 #Username and password are accepted
```

```
## Response [http://httpbin.org/basic-auth/user/passwd]
##   Date: 2020-03-29 00:46
##   Status: 200
```

```
##   Content-Type: application/json
##   Size: 47 B
## {
##   "authenticated": true,
##   "user": "user"
## }
```

```r
names(pg2)
```

```
##  [1] "url"         "status_code" "headers"     "all_headers" "cookies"
##  [6] "content"     "date"        "times"       "request"     "handle"
```

- Using handles allows you to not have to re-authenticate

```r
google = handle("http://google.com")
pg1 <-  GET(handle = google, path = "/")
pg2 = GET(handle = google, path = "search")
```

- Further resources
  - **(R Bloggers sahs a number of examples of web scraping)[http://www.r-bloggers.com/?s=Web+Scraping]**

  - **(The httr help file has useful examples)[http:cran.r-project.org/web/packages/httr/httr.pdf]**

Reading from APIs

- *API* - Application Programming Interfaces
  - Often have to create a dev account **(Twitter)[https://dev.twitter.com/apps]**
- … (Little instruction was provided on how to set up the API, so I'm not even sure if I did it correctly)
  - Access Token: **1182685077089722369-ydEBDC6bazluY5j8wGj0oqIx9ayQ7B**

  - Access token secret: **RctQSVtfGelTCfs2rIVMalfmMNyQk40NLZhhgECk0ekCP**

  - API key: **s9FXX7R5Cjb5Oyr8NpH1HDTzj**

  - API secret key: **7c3JnIIviRiapbkD0Ipp25n6cBFvUuHOdSiDSqd62LKLhOe5zl**

```r
##Running this in markdown crashed R for me
library(httr)
 myapp <- oauth_app("twitter",
                    key = "s9FXX7R5Cjb5Oyr8NpH1HDTzj",
                    secret = "7c3JnIIviRiapbkD0Ipp25n6cBFvUuHOdSiDSqd62LKLhOe5zl")
 sig <- sign_oauth1.0(myapp,
                       token = "1182685077089722369-ydEBDC6bazluY5j8wGj0oqIx9ayQ7B",
                       token_secret = "7c3JnIIviRiapbkD0Ipp25n6cBFvUuHOdSiDSqd62LKLhOe5zl")
 homeTL <- GET("https://api.twitter.com/1.1/statuses/home_timeline.json", sig)
```

- "https://api.twitter.com/1.1/statuses/home_timeline.json" is a particular URL depending on what data you wish to get out.
    - `1.1` - version of api
    - `statuses/home_timeline` - specifies data

    - `.json` specifies output (as of 1.1 twitter only has JSON)
- Converting the JSON object

```
#Then this shid can't authenticate me
library(httr)
json1 <- content(homeTL)
library(jsonlite)
json2 <- jsonlite::fromJSON(toJSON(json1))
json2[1]
```

```
## $errors
##   code                    message
## 1   32 Could not authenticate you.
```

- There's documentation about twitter API... somewhere, the f***in' link doesn't work

- Overview

    - httr allows `GET`, `POST`, `PUT`, `DELETE` requests if you are authorized

    - You can authernticate with a user name or a password

    - Most modern APIs use something like oauth

    - httr works well with Facebook, Google, Twitter, Github, etc.

Reading from Other Sources

- Roger has a video, **(There's An R Package for That)[https://www.youtube.com/watch?v=yhTerzNFLbo]**

- In general the best way to find out if the R package exists is to search " R package"

    - For example: "MySQL R package"

- Interacting more directly with files

    - `file` - open a connection to a text file

    - `url` - open a connection to a url

    - `gzfile` - open a connection to a .gz file

- `bzfile` - open a connection to a .bz2 file
- `?connections` for more information
- ***Remember to close connections***

- Foreign Packages - Helpful if you work with people that use other programming languages
  - Loads data from `Minitab`, `S`, `SAS`, `SPSS`, `Stata`, `Systat`
  - Basic functions are `read.fileType`
    - `read.arff` (Weka)
    - `read.dta` (Stata)
    - `read.mtp` (Minitab)
    - `read.octave` (Octave)
    - `read.spss` (SPSS)
    - `read.xport` (SAS)
  - **(See the help page for more details)[http://cran.r-project.org/web/packages/foreign/foreign.pdf]**

- Examples of other database packages
  - RPostresSQL provides a DBI-compliant database connection from R.
    - **(Tutorial)[https://code.google.com/p/rpostgresql/]**
    - **(help file)[http://cran.r-project.org/web/packages/RPostgreSQL/RPostgreSQL.pdf]**
  - RODBC provides interfaces to multiple databases including PostgreQL, MySQL, Microsoft Access and SQLite.
    - **(Tutorial)[http://cran.r-project.org/web/packages/RODBC/vignettes/RODBC.pdf]**
    - **(help file)[http://cran.r-project.org/web/packages/RODBC/RODBC.pdf]**
  - RMongo
    - **(Tutorial)[http://cran.r-project.org/web/packages/RMongo/RMongo.pdf]**
    - **(example of Rmongo)[http://www.r-bloggers.com/r-and-mongodb/]
    - **(example of rmongodb)[http://cran.r-project.org/web/packages/rmongodb/rmongodb.pdf]**
    - Both of which provide interfaces to MongoDb.

- Reading images
  - **(jpeg)[http://cran.r-project.org/web/packages/jpeg/index.html]**
  - **(readbitmap)[http://cran.r-project.org/web/packages/readbitmap/index.html]**
  - **(png)[http://cran.r-project.org/web/packages/png/index.html]**

  - **(EBImage (Bioconductor))[http://www.bioconductor.org/packages/2.13/bioc/html/EBImage.html]**

- Reading GIS data

  - **(rgdal)[http://cran.r-project.org/web/packages/rgdal/index.html]**

  - **(rgeos)[http://cran.r-project.org/web/packages/rgeos/index.html]**
  - **(raster)[http://cran.r-project.org/web/packages/raster/index.html]**

- Reading music data

  - **(tuneR)[http://cran.r-project.org/web/packages/tuneR/]**

  - **(seewave)[http://rug.mnhn.fr/seewave/]**

Quiz Scribbles

1) Register an application with the Github API **(here)[https://github.com/settings/applications]**. Access the API to get information on your instructors repositories (hint: this is the url you want "https://api.github.com/users/jtleek/repos"). Use this data to find the time that the datasharing repo was created. What time was it created?

(This tutorial may be useful)[https://github.com/hadley/httr/blob/master/demo/oauth2-github.r]. You may also need to run the code in the base R package and not R studio.

```
#1 - Authentication
myapp <- oauth_app("github",
             key = "cd0a71fdf4a07ef2d2a2",
             secret = "8b8a5ebe8a36f41e6970f4f8dd8467dbc0a7c451")
github_token <- oauth2.0_token(oauth_endpoints("github"), myapp)

gtoken <- config(token = github_token)
req <- GET("https://api.github.com/rate_limit", gtoken)
stop_for_status(req)
content(req)
```

```
#1 - Finding when jtleek's datasharing repo was created
##(Couldn't figure it out with API so just used JSON)
library(httr)
library(jsonlite)
info <- fromJSON("https://api.github.com/users/jtleek/repos")
target <- info$name=="datasharing"
info$created_at[target]
```

2) The `sqldf` package allows for execution of SQL commands on R data frames. We will use the sqldf package to practice the queries we might send with the `dbSendQuery` command in `library(RMySQL)`.

Download the American Community Survey data … and load it into an R object called `acs`

```r
url <- "https://d396qusza40orc.cloudfront.net/getdata%2Fdata%2Fss06pid.csv"
saveLoc <- paste(getwd(), "/AmericanCommunitySurvey.csv", sep ="")
download.file(url, saveLoc, method = "curl")
# ...
library(RCurl)
acs <- read.csv(saveLoc)
```

Which of the following commands will select only the data for the probability weights `pwgtp1` with ages less than 50?

```r
#library(RSQLite)
#library(sqldf)

#Incorrect# sqldf("select pwgtp1 from acs")

#Incorrect# sqldf("select * from acs where AGEP < 50")

#sqldf("select pwgtp1 from acs where AGEP < 50")

#Incorrect# sqldf("select * from acs")
```

3) Using the same data frame you created in the previous problem, what is the equivalent function to `unique(acs$AGEP)`?

```r
# error <- "Syntax error"
# ogFn <- unique(acs$AGEP)
# Op1 <- error #sqldf("select unique AGEP from acs")
# Op2 <- sqldf("select distinct pwgtp1 from acs")
# Op3 <- error #sqldf("select AGEP where unique from acs")
# Op4 <- sqldf("select distinct AGEP from acs")
# selection <- c(Op1, Op2, Op3, Op4)
# #ogFn ##Output was gross so I omitted this
# selection[2]
# selection[4]
```

4) How many characters are in the 10th, 20th, 30th and 100th lines of HTML from **this** page:

```r
# url <- "http://biostat.jhsph.edu/~jleek/contact.html"
# htmlCode <- readLines(url(url))
# nchar(htmlCode[c(10,20,30,100)])
```

5) Read **this** data set into R and report the sum of the numbers in the 4th of the 9th columns. (**Original source of the data**)

```r
url <- "https://d396qusza40orc.cloudfront.net/getdata%2Fwksst8110.for"
saveLoc <- paste(getwd(), "/Quiz2_Q5_Dataset.for", sep="")
download.file(url, saveLoc, "curl")
info <- read.fwf(saveLoc, widths = c(10, rep(c(-5, 4, 4),4)))
info <- info[-4:-1,] #removing header
sum(as.numeric(as.character(info[,4])),as.numeric(as.character(info[,9])))
```

```
## [1] 32463.2
```

Side tangent

While reading **a FiveThirtyEight article** I noticed that there was a 1 vote diffrence between Whether to convict President Trump on a charge of: **obstruction of Congress** (47-53), and **abuse of power** (48-52). So the (obvious) question is, who was the swing voter? The following codeblock is to determine this.

```
#obstruction of Congress
SenateVote <- function(URL, filename) {
library(XML)
url <- URL
saveLoc <- paste(getwd(), filename, sep = "/")
download.file(url, saveLoc)
info <- xmlTreeParse(saveLoc, useInternalNodes = TRUE)
root <- xmlRoot(info)
members <- root[["members"]]
senatorsLN <- xpathSApply(members, "//last_name", xmlValue)
senatorsVote <- xpathSApply(members, "//vote_cast", xmlValue)
df <- data.frame(name = senatorsLN, vote = senatorsVote)
df
}

URL <- ("https://www.senate.gov/legislative/LIS/roll_call_votes/vote1162/vote_116_2_00034.xml")
filename <- "Obstruction.xml"
obstruction <- SenateVote(URL, filename)

URL <- "https://www.senate.gov/legislative/LIS/roll_call_votes/vote1162/vote_116_2_00033.xml"
filename <- "Abuse.xml"
abuse <- SenateVote(URL, filename)

dif <- (obstruction[,2] != abuse[,2])
data.frame(Name = abuse[dif,1], Obstruction = obstruction[dif,2], Abuse = abuse[dif,2])
```

```
##     Name Obstruction  Abuse
## 1 Romney  Not Guilty Guilty
```

Looks like it was our man, "R-money".

Organizing, Merging, and Managing the Data

- When getting a dataset the first goal is to convert it into *tidy data.*
  - each variable is a column

  - each observation is a row

- Each table/file stores data about one kind of observation (e. g. people/hospitals)

##**Hadley's paper on tidy data** * Tidy data facilitates inital exploration and analysis of the data
* Sometimes what one variable is will vary depending on the field the analysis is in (Pg. 4) *(tidy data) ensures that values of different variables from the same observationa re always paired* a good ordering of variables and observations make sit easier to can the raw values + Fixed variables should come first (e. g. `country`, `year`) + measured variables shoudl follow (e. g. `week 1`, `week 2`, ...) + Rows can then be ordered by the first variable, breaking ties witht he second and subsequent variables
* Tools for tidying: + Melting - turning columns into rows (`cols` of varrying income should be in one col, `income`) + String splitting - separating cnames when they contain more than 1 variable + Casting - creating varNames from elements in data * **The datasets and the R code used to tidy them in this paper** * Five most common problems with messy data: 1) Column headers are values, not variable names
+ Tidyed with melting + Ex: cols - income bracker, rows - religion; should be that cols are: `religon`, `income`, `freq` 2) Multiple variables are stored in one column
+ Tidyed with string splitting + Ex: sex & age are stored in a column (`m014`, `m1524`, `...`, `f014`, `f1524`, `...`) 3) Variables are stored in both rows and columns
+ Tidyed with melting followed by casting + Ex: weather data with `d1:31` listed as colNames as well as element determining `tmax` or `tmin` for max and min tempature (respectively) 4) Multiple types of observational units are stored in the same table
+ Tidyed by separating observational data into seperate tables
+ Ex: Song Billboard dataset - `artist`, `song name` and `time` are one observation, then `rank` and it's respective `week` 5) A single observational unit is stored in multiple tables
+ Tidyed by pulling the data into one table + Ex: Baby names per year pulled into a database of all baby names

Subsetting and Sorting

- Subsetting - quick review

```
set.seed(13435)
X <- data.frame("var1"=sample(1:5), "var2"=sample(6:10), "var3"=sample(11:15))
X <- X[sample(1:5),] #shuffle the df by row
X$var2[c(1,3)] <- NA #Insert NA at 1st and 3rd index of var2
X
```

```
##   var1 var2 var3
## 5    2   NA   11
## 4    4   10   12
## 1    3   NA   14
## 2    1    7   15
## 3    5    6   13
```

```
X[,1] #Returns first col
```

```
## [1] 2 4 3 1 5
```

```r
X[,"var1"]#Can also use variable name
```

```
## [1] 2 4 3 1 5
```

```r
X[1:2, "var2"] #output first two values of var2
```

```
## [1] NA 10
```

- Logicals ands & ors

```r
X[(X$var1 <= 3 & X$var3 > 11),] #Ex of an "and" logical
```

```
##   var1 var2 var3
## 1    3   NA   14
## 2    1    7   15
```

```r
X[(X$var1 <= 3 | X$var3 > 15),] #Ex of an "or" logical
```

```
##   var1 var2 var3
## 5    2   NA   11
## 1    3   NA   14
## 2    1    7   15
```

- Dealing with missing values

```r
X[which(X$var2 > 8),]# returns values 'which' var2 > 8 & skips NA values
```

```
##   var1 var2 var3
## 4    4   10   12
```

- Sorting

```r
sort(X$var1) #increasing by default
```

```
## [1] 1 2 3 4 5
```

```r
sort(X$var1, decreasing = TRUE)
```

```
## [1] 5 4 3 2 1
```

```r
sort(X$var2) #Removes NAs by default
```

```
## [1]  6  7 10
```

```r
sort(X$var2, na.last = TRUE)
```

```
## [1]  6  7 10 NA NA
```

```r
sort(X$var2, na.last = FALSE)#Puts NAs up front
```

```
## [1] NA NA  6  7 10
```

- Ordering (Sorts whole d.f. with respect to 1 or more vars)

```r
X[order(X$var1),]
```

```
##   var1 var2 var3
```

```
## 2    1    7   15
## 5    2   NA   11
## 1    3   NA   14
## 4    4   10   12
## 3    5    6   13
```

```
X[order(X$var1, X$var3),]#Same result since there are no ties in var1
```

```
##   var1 var2 var3
## 2    1    7   15
## 5    2   NA   11
## 1    3   NA   14
## 4    4   10   12
## 3    5    6   13
```

- Ordering with `plyr` package
  - The `plyr` package was written by Hadley Wickham

```
library(plyr)
arrange(X,var1)
```

```
##   var1 var2 var3
## 1    1    7   15
## 2    2   NA   11
## 3    3   NA   14
## 4    4   10   12
## 5    5    6   13
```

```
arrange(X,desc(var1))#Puts it in descending order
```

```
##   var1 var2 var3
## 1    5    6   13
## 2    4   10   12
## 3    3   NA   14
## 4    2   NA   11
## 5    1    7   15
```

- Adding rows and columns

```
X$var4 <- rnorm(5)
X
```

```
##   var1 var2 var3       var4
## 5    2   NA   11 -0.4150458
## 4    4   10   12  2.5437602
## 1    3   NA   14  1.5545298
## 2    1    7   15 -0.6192328
## 3    5    6   13 -0.9261035
```

```
#Or with cbind command (column bind)
Y <- cbind(X, rnorm(5))#binds in order of params
Y
```

```
##   var1 var2 var3       var4    rnorm(5)
## 5    2   NA   11 -0.4150458 -0.66549949
## 4    4   10   12  2.5437602 -0.02166735
## 1    3   NA   14  1.5545298 -0.17411953
## 2    1    7   15 -0.6192328  0.23900438
## 3    5    6   13 -0.9261035 -1.83245959
```

```
#Similar function called rbind
```

- **Andrew Jaffe's lecture notes**

Summarizing Data

- Key process of data cleaning is to identify any quirks or weird issues you need to address before doing your analysis

- Example is using **Restaurant data from the city of Baltimore**

```
if(!file.exists("./data")){dir.create("./data")}
fileUrl <- "https://data.baltimorecity.gov/api/views/k5ry-ef3g/rows.csv?accessType=DOWNLOAD"
saveLoc <- paste0(getwd(), "/data/restaurants.csv")
download.file(fileUrl, saveLoc, method = "curl")
restData <- read.csv(saveLoc)
```

Look at a bit of the data

```
head(restData, n = 3) #n will determine number of rows to show, default is 6
```

```
##    name zipCode neighborhood councilDistrict policeDistrict
## 1   410   21206    Frankford               2   NORTHEASTERN
## 2  1919   21231  Fells Point               1   SOUTHEASTERN
## 3 SAUTE   21224       Canton               1   SOUTHEASTERN
##                         Location.1 X2010.Census.Neighborhoods
## 1 4509 BELAIR ROAD\nBaltimore, MD                          NA
## 2    1919 FLEET ST\nBaltimore, MD                          NA
## 3   2844 HUDSON ST\nBaltimore, MD                          NA
##   X2010.Census.Wards.Precincts Zip.Codes
## 1                           NA        NA
## 2                           NA        NA
## 3                           NA        NA
```

```
tail(restData, n = 3)
```

```
##                name zipCode  neighborhood councilDistrict policeDistrict
## 1325 ZINK'S CAF\u0090   21213 Belair-Edison              13   NORTHEASTERN
## 1326     ZISSIMOS BAR   21211       Hampden               7       NORTHERN
## 1327           ZORBAS   21224     Greektown               2   SOUTHEASTERN
##                            Location.1 X2010.Census.Neighborhoods
```

```
## 1325 3300 LAWNVIEW AVE\nBaltimore, MD                         NA
## 1326      1023 36TH ST\nBaltimore, MD                         NA
## 1327  4710 EASTERN Ave\nBaltimore, MD                         NA
##      X2010.Census.Wards.Precincts Zip.Codes
## 1325                           NA        NA
## 1326                           NA        NA
## 1327                           NA        NA
```

Make summary

- Gives a summary for every variable
- Text based variables will show a count
- Quanitative variables will show a stat summary
  - In the below example the Min. value shows a negative value for a zipCode, which shouldn't have occured

```r
summary(restData)
```

```
##                            name         zipCode             neighborhood
##  MCDONALD'S                    :   8   Min.   :-21226   Downtown     :128
##  POPEYES FAMOUS FRIED CHICKEN:   7   1st Qu.: 21202   Fells Point : 91
##  SUBWAY                        :   6   Median : 21218   Inner Harbor: 89
##  KENTUCKY FRIED CHICKEN        :   5   Mean   : 21185   Canton       : 81
##  BURGER KING                   :   4   3rd Qu.: 21226   Federal Hill: 42
##  DUNKIN DONUTS                 :   4   Max.   : 21287   Mount Vernon: 33
##  (Other)                       :1293                     (Other)      :863
##  councilDistrict       policeDistrict                         Location.1
##  Min.   : 1.000   SOUTHEASTERN:385    1101 RUSSELL ST\nBaltimore, MD:   9
##  1st Qu.: 2.000   CENTRAL     :288    201 PRATT ST\nBaltimore, MD   :   8
##  Median : 9.000   SOUTHERN    :213    2400 BOSTON ST\nBaltimore, MD :   8
##  Mean   : 7.191   NORTHERN    :157    300 LIGHT ST\nBaltimore, MD   :   5
##  3rd Qu.:11.000   NORTHEASTERN: 72    300 CHARLES ST\nBaltimore, MD :   4
##  Max.   :14.000   EASTERN     : 67    301 LIGHT ST\nBaltimore, MD   :   4
##                   (Other)     :145    (Other)                       :1289
##  X2010.Census.Neighborhoods X2010.Census.Wards.Precincts Zip.Codes
##  Mode:logical               Mode:logical                 Mode:logical
##  NA's:1327                  NA's:1327                    NA's:1327
##
##
##
##
##
```

```r
#Checking that negative zipcode
restData$name[(restData$zipCode == -21226)]
```

```
## [1] TASTE INTERNATIONAL RESTAURANT & LOUNGE/BAR
## 1277 Levels: #1 CHINESE KITCHEN #1 chinese restaurant 1919 19TH HOLE ... ZORBAS
```

```r
#Perhaps the negative was to list this location twice
restData$name[(restData$zipCode == 21226)][16]
```

```
## [1] TASTE INTERNATIONAL RESTAURANT
## 1277 Levels: #1 CHINESE KITCHEN #1 chinese restaurant 1919 19TH HOLE ... ZORBAS
```

str command

- Tells info about data type of variable and it's classes

```r
str(restData)
```

```
## 'data.frame':    1327 obs. of  9 variables:
##  $ name                          : Factor w/ 1277 levels "#1 CHINESE KITCHEN",..: 9 3 992 1 2
##  $ zipCode                       : int  21206 21231 21224 21211 21223 21218 21205 21211 21205
##  $ neighborhood                  : Factor w/ 173 levels "Abell","Arlington",..: 53 52 18 66 10
##  $ councilDistrict               : int  2 1 1 14 9 14 13 7 13 1 ...
##  $ policeDistrict                : Factor w/ 9 levels "CENTRAL","EASTERN",..: 3 6 6 4 8 3 6 4
##  $ Location.1                    : Factor w/ 1210 levels "1 BIDDLE ST\nBaltimore, MD",..: 835
##  $ X2010.Census.Neighborhoods    : logi  NA NA NA NA NA NA ...
##  $ X2010.Census.Wards.Precincts: logi  NA NA NA NA NA NA ...
##  $ Zip.Codes                     : logi  NA NA NA NA NA NA ...
```

Quantiles of quantitative variables

```r
quantile(restData$councilDistrict, na.rm = TRUE)
```

```
##   0%  25%  50%  75% 100% 
##    1    2    9   11   14
```

```r
#Can also change the percentiles of the function
quantile(restData$councilDistrict, probs = c(0.5,0.75,0.9))
```

```
## 50% 75% 90% 
##   9  11  12
```

Make table

```r
table(restData$zipCode)
```

```
## 
## -21226  21201  21202  21205  21206  21207  21208  21209  21210  21211  21212 
##      1    136    201     27     30      4      1      8     23     41     28 
##  21213  21214  21215  21216  21217  21218  21220  21222  21223  21224  21225 
##     31     17     54     10     32     69      1      7     56    199     19 
##  21226  21227  21229  21230  21231  21234  21237  21239  21251  21287 
##     18      4     13    156    127      7      1      3      2      1
```

```r
#2D table
table(restData$councilDistrict, restData$zipCode)[,1:15]#1:15 to limit output
```

```
##
##      -21226 21201 21202 21205 21206 21207 21208 21209 21210 21211 21212 21213
##   1       0     0    37     0     0     0     0     0     0     0     0     2
##   2       0     0     0     3    27     0     0     0     0     0     0     0
##   3       0     0     0     0     0     0     0     0     0     0     0     2
##   4       0     0     0     0     0     0     0     0     0     0    27     0
##   5       0     0     0     0     0     3     0     6     0     0     0     0
##   6       0     0     0     0     0     0     0     1    19     0     0     0
##   7       0     0     0     0     0     0     0     1     0    27     0     0
##   8       0     0     0     0     0     1     0     0     0     0     0     0
##   9       0     1     0     0     0     0     0     0     0     0     0     0
##   10      1     0     1     0     0     0     0     0     0     0     0     0
##   11      0   115   139     0     0     0     1     0     0     0     1     0
##   12      0    20    24     4     0     0     0     0     0     0     0    13
##   13      0     0     0    20     3     0     0     0     0     0     0    13
##   14      0     0     0     0     0     0     0     0     4    14     0     1
##
##      21214 21215 21216
##   1      0     0     0
##   2      0     0     0
##   3     17     0     0
##   4      0     0     0
##   5      0    31     0
##   6      0    15     1
##   7      0     6     7
##   8      0     0     0
##   9      0     0     2
##   10     0     0     0
##   11     0     0     0
##   12     0     0     0
##   13     0     1     0
##   14     0     1     0
```

Check for missing values

```r
sum(is.na(restData$councilDistrict))#Returns num of NA values
```

```
## [1] 0
```

```r
any(is.na(restData$councilDistrict))#Returns TRUE if any of the values are NA
```

```
## [1] FALSE
```

```r
all(restData$zipCode > 0)#FALSE if at least 1 value is <0
```

```
## [1] FALSE
```

Row and column sums

```r
colSums(is.na(restData))
```

```
##                          name                        zipCode
##                             0                              0
##                  neighborhood                councilDistrict
##                             0                              0
##                policeDistrict                     Location.1
##                             0                              0
##    X2010.Census.Neighborhoods X2010.Census.Wards.Precincts
##                          1327                           1327
##                     Zip.Codes
##                          1327
```

```r
all(colSums(is.na(restData))==0)#TRUE if there are no NA values
```

```
## [1] FALSE
```

Values with specific characteristics

```r
table(restData$zipCode %in% c("21212"))
```

```
##
## FALSE  TRUE
##  1299    28
```

```r
table(restData$zipCode %in% c("21212", "21213"))
```

```
##
## FALSE  TRUE
##  1268    59
```

```r
#Subsetting with ^that logical (table isn't part of the command)
restData[restData$zipCode %in% c("21212", "21213"),]
```

```
##                                  name zipCode                neighborhood
## 29                  BAY ATLANTIC CLUB   21212                    Downtown
## 39                        BERMUDA BAR   21213               Broadway East
## 92                          ATWATER'S   21212   Chinquapin Park-Belvedere
## 111          BALTIMORE ESTONIAN SOCIETY   21213          South Clifton Park
## 187                          CAFE ZEN   21212                    Rosebank
## 220                 CERIELLO FINE FOODS   21212   Chinquapin Park-Belvedere
## 266     CLIFTON PARK GOLF COURSE SNACK BAR   21213                 Darley Park
## 276               CLUB HOUSE BAR & GRILL   21213 Orangeville Industrial Area
## 289                CLUBHOUSE BAR & GRILL   21213 Orangeville Industrial Area
```

```
## 291                              COCKY LOU'S   21213             Broadway East
## 362        DREAM TAVERN, CARRIBEAN  U.S.A.   21213             Broadway East
## 373                            DUNKIN DONUTS   21212                  Homeland
## 383         EASTSIDE  SPORTS  SOCIAL  CLUB   21213             Broadway East
## 417                         FIELDS OLD TRAIL   21212                Mid-Govans
## 475                                GRAND CRU   21212 Chinquapin Park-Belvedere
## 545                               RANDY'S BAR   21213             Broadway East
## 604      MURPHY'S NEIGHBORHOOD BAR & GRILL   21212                Mid-Govans
## 616                                   NEOPOL   21212 Chinquapin Park-Belvedere
## 620             NEW CLUB THUNDERBIRD INC.   21213               Middle East
## 626                      NEW MAYFIELD, INC.   21213             Belair-Edison
## 678                             IKAN SEAFOOD   21212 Chinquapin Park-Belvedere
## 711                             KAY-CEE CLUB   21212                  Homeland
## 763                                   LA'RAE   21213                    Oliver
## 777                    LEMONGRASS BALTIMORE   21213              Little Italy
## 779                     LEN'S SANDWICH SHOP   21213             Broadway East
## 845                               MCDONALD'S   21213        South Clifton Park
## 852                               MCDONALD'S   21212            Radnor-Winston
## 873                     NEW REX LIQUORS,INC.   21212               Wilson Park
## 895                                OK TAVERN   21213             Biddle Street
## 919                             PANERA BREAD   21212               Lake Walker
## 940                      PEIWEI ASIAN DINER   21212               Cedarcroft
## 949                     PERGUSA ENTERPRISES   21212                  Rosebank
## 957                 PHANTOM'S BAR AND GRILL   21213             Belair-Edison
## 976            POPEYES FAMOUS FRIED CHICKEN   21212            Winston-Govans
## 994                           ROBBIE'S NEST   21213             Broadway East
## 1017                             RUTLAND BAR   21213             Broadway East
## 1018                        RYAN'S DAUGHTER   21212 Chinquapin Park-Belvedere
## 1022           saigon remembered restaurant   21212                Mid-Govans
## 1053                    SHIRLEY'S  HONEY HOLE   21213             Broadway East
## 1120                       STEEPLE CHASE II   21213             Biddle Street
## 1122                                  SUBWAY   21213                    Oliver
## 1153                                 TAM-TAM   21212                Mid-Govans
## 1155                                   TASTE   21212                Mid-Govans
## 1159                             TAYLORS EAST   21213                     Berea
## 1186                    THE EDGE BAR & LOUNGE   21213             Broadway East
## 1187 THE EDGE BAR & LOUNGE - KITCHEN AREA   21213             Broadway East
## 1198                   THE HOLLOW BAR & GRILL   21212                  Rosebank
## 1209                 THE NEW BUCKETT'S LOUNGE   21213             Broadway East
## 1232                             THREE  ACE'S   21213             Belair-Edison
## 1246                     TORAIN'S  HIDE-A-WAY   21213             Broadway East
## 1259                      TSUNAMI BALTIMORE   21213              Little Italy
## 1287                              VITO'S PIZZA   21212               Cedarcroft
## 1298 WENDY'S OLD FASHIONED HAMBURGERS #96   21212                  Homeland
## 1304                     WHITTEN'S  (4502-04)   21213         Claremont-Freedom
## 1312                               wozi lounge   21212                  Guilford
## 1319               YETI RESTAURANT & CARRYOUT   21212                  Rosebank
## 1320                         YORK CLUB TAVERN   21212                  Homeland
```

```
## 1323          ZEN WEST ROADSIDE CANTINA   21212                 Rosebank
## 1325                   ZINK'S CAF\u0090   21213            Belair-Edison
##      councilDistrict policeDistrict                         Location.1
## 29                11        CENTRAL      206 REDWOOD ST\nBaltimore, MD
## 39                12        EASTERN      1801 NORTH AVE\nBaltimore, MD
## 92                 4       NORTHERN   529 BELVEDERE AVE\nBaltimore, MD
## 111               12        EASTERN      1932 BELAIR RD\nBaltimore, MD
## 187                4       NORTHERN   438 BELVEDERE AVE\nBaltimore, MD
## 220                4       NORTHERN   529 BELVEDERE AVE\nBaltimore, MD
## 266               14   NORTHEASTERN       2701 ST LO DR\nBaltimore, MD
## 276               13        EASTERN     4217 ERDMAN AVE\nBaltimore, MD
## 289               13        EASTERN     4217 ERDMAN AVE\nBaltimore, MD
## 291               12        EASTERN      2101 NORTH AVE\nBaltimore, MD
## 362               13        EASTERN  2300 LAFAYETTE AVE\nBaltimore, MD
## 373                4       NORTHERN        5422 YORK RD\nBaltimore, MD
## 383               13        EASTERN 1203 COLLINGTON AVE\nBaltimore, MD
## 417                4       NORTHERN        5723 YORK RD\nBaltimore, MD
## 475                4       NORTHERN   527 BELVEDERE AVE\nBaltimore, MD
## 545               12        EASTERN      2135 NORTH AVE\nBaltimore, MD
## 604                4       NORTHERN        5847 YORK RD\nBaltimore, MD
## 616                4       NORTHERN   529 BELVEDERE AVE\nBaltimore, MD
## 620               13        EASTERN       2201 CHASE ST\nBaltimore, MD
## 626               13   NORTHEASTERN      3349 BELAIR RD\nBaltimore, MD
## 678                4       NORTHERN   529 BELVEDERE AVE\nBaltimore, MD
## 711                4       NORTHERN    201 HOMELAND AVE\nBaltimore, MD
## 763               12        EASTERN     1000 HOFFMAN ST\nBaltimore, MD
## 777                1   SOUTHEASTERN    1300 BANK STREET\nBaltimore, MD
## 779               12        EASTERN  1500 WASHINGTON ST\nBaltimore, MD
## 845               12        EASTERN       2001 BROADWAY\nBaltimore, MD
## 852                4       NORTHERN        5100 YORK RD\nBaltimore, MD
## 873                4       NORTHERN        4637 YORK RD\nBaltimore, MD
## 895               13        EASTERN      2301 BIDDLE ST\nBaltimore, MD
## 919                4       NORTHERN    6307 1 2 YORK RD\nBaltimore, MD
## 940                4       NORTHERN        6302 YORK RD\nBaltimore, MD
## 949                4       NORTHERN        5928 YORK RD\nBaltimore, MD
## 957                3   NORTHEASTERN      3539 BELAIR RD\nBaltimore, MD
## 976                4       NORTHERN        5002 YORK RD\nBaltimore, MD
## 994               12        EASTERN      2250 NORTH AVE\nBaltimore, MD
## 1017              12        EASTERN    1508 RUTLAND AVE\nBaltimore, MD
## 1018               4       NORTHERN   600 BELVEDERE AVE\nBaltimore, MD
## 1022               4       NORTHERN        5857 york rd\nBaltimore, MD
## 1053              13        EASTERN      2300 OLIVER ST\nBaltimore, MD
## 1120              13        EASTERN       2401 CHASE ST\nBaltimore, MD
## 1122              12        EASTERN      1400 NORTH AVE\nBaltimore, MD
## 1153               4       NORTHERN        5722 YORK RD\nBaltimore, MD
## 1155               4       NORTHERN   510 BELVEDERE AVE\nBaltimore, MD
## 1159              13        EASTERN     1201 POTOMAC ST\nBaltimore, MD
## 1186              12        EASTERN     2015 FEDERAL ST\nBaltimore, MD
```

```
## 1187                12        EASTERN     2015 FEDERAL ST\nBaltimore, MD
## 1198                 4       NORTHERN        5921 YORK RD\nBaltimore, MD
## 1209                13        EASTERN     1432 CHESTER ST\nBaltimore, MD
## 1232                 3   NORTHEASTERN      3534 belair RD\nBaltimore, MD
## 1246                12        EASTERN   1701 ELLSWORTH ST\nBaltimore, MD
## 1259                 1   SOUTHEASTERN        1300 BANK ST\nBaltimore, MD
## 1287                 4       NORTHERN        6304 YORK RD\nBaltimore, MD
## 1298                 4       NORTHERN        5615 YORK RD\nBaltimore, MD
## 1304                13   NORTHEASTERN     4502 ERDMAN AVE\nBaltimore, MD
## 1312                 4       NORTHERN        4515 YORK RD\nBaltimore, MD
## 1319                 4       NORTHERN        5926 YORK RD\nBaltimore, MD
## 1320                 4       NORTHERN        5407 YORK RD\nBaltimore, MD
## 1323                 4       NORTHERN        5916 YORK RD\nBaltimore, MD
## 1325                13   NORTHEASTERN   3300 LAWNVIEW AVE\nBaltimore, MD
##      X2010.Census.Neighborhoods X2010.Census.Wards.Precincts Zip.Codes
## 29                           NA                           NA        NA
## 39                           NA                           NA        NA
## 92                           NA                           NA        NA
## 111                          NA                           NA        NA
## 187                          NA                           NA        NA
## 220                          NA                           NA        NA
## 266                          NA                           NA        NA
## 276                          NA                           NA        NA
## 289                          NA                           NA        NA
## 291                          NA                           NA        NA
## 362                          NA                           NA        NA
## 373                          NA                           NA        NA
## 383                          NA                           NA        NA
## 417                          NA                           NA        NA
## 475                          NA                           NA        NA
## 545                          NA                           NA        NA
## 604                          NA                           NA        NA
## 616                          NA                           NA        NA
## 620                          NA                           NA        NA
## 626                          NA                           NA        NA
## 678                          NA                           NA        NA
## 711                          NA                           NA        NA
## 763                          NA                           NA        NA
## 777                          NA                           NA        NA
## 779                          NA                           NA        NA
## 845                          NA                           NA        NA
## 852                          NA                           NA        NA
## 873                          NA                           NA        NA
## 895                          NA                           NA        NA
## 919                          NA                           NA        NA
## 940                          NA                           NA        NA
## 949                          NA                           NA        NA
## 957                          NA                           NA        NA
```

```
## 976                          NA                           NA         NA
## 994                          NA                           NA         NA
## 1017                         NA                           NA         NA
## 1018                         NA                           NA         NA
## 1022                         NA                           NA         NA
## 1053                         NA                           NA         NA
## 1120                         NA                           NA         NA
## 1122                         NA                           NA         NA
## 1153                         NA                           NA         NA
## 1155                         NA                           NA         NA
## 1159                         NA                           NA         NA
## 1186                         NA                           NA         NA
## 1187                         NA                           NA         NA
## 1198                         NA                           NA         NA
## 1209                         NA                           NA         NA
## 1232                         NA                           NA         NA
## 1246                         NA                           NA         NA
## 1259                         NA                           NA         NA
## 1287                         NA                           NA         NA
## 1298                         NA                           NA         NA
## 1304                         NA                           NA         NA
## 1312                         NA                           NA         NA
## 1319                         NA                           NA         NA
## 1320                         NA                           NA         NA
## 1323                         NA                           NA         NA
## 1325                         NA                           NA         NA
```

Cross tabs

```r
data(UCBAdmissions) #This is a dataset included in R
DF <- as.data.frame(UCBAdmissions)
summary(DF)
```

```
##       Admit       Gender   Dept       Freq
##  Admitted:12   Male  :12   A:4   Min.   :  8.0
##  Rejected:12   Female:12   B:4   1st Qu.: 80.0
##                            C:4   Median :170.0
##                            D:4   Mean   :188.6
##                            E:4   3rd Qu.:302.5
##                            F:4   Max.   :512.0
```

```r
xt <- xtabs(Freq ~ Gender + Admit, data = DF) #susbet is optional
xt #Freq table of Gender to num of Admissions
```

```
##         Admit
## Gender   Admitted Rejected
##   Male       1198     1493
```

```
##   Female      557      1278
```

Flat tables

```r
#warpbreaks is another dataset included in R
warpbreaks$replicate <- rep(1:9, len = 54)
xt <- xtabs(breaks ~., data=warpbreaks)#Break by all(.) variables in dataset
xt
```

```
## , , replicate = 1
##
##     tension
## wool  L  M  H
##    A 26 18 36
##    B 27 42 20
##
## , , replicate = 2
##
##     tension
## wool  L  M  H
##    A 30 21 21
##    B 14 26 21
##
## , , replicate = 3
##
##     tension
## wool  L  M  H
##    A 54 29 24
##    B 29 19 24
##
## , , replicate = 4
##
##     tension
## wool  L  M  H
##    A 25 17 18
##    B 19 16 17
##
## , , replicate = 5
##
##     tension
## wool  L  M  H
##    A 70 12 10
##    B 29 39 13
##
## , , replicate = 6
##
##     tension
```

```
## wool  L  M  H
##    A 52 18 43
##    B 31 28 15
##
## , , replicate = 7
##
##     tension
## wool  L  M  H
##    A 51 35 28
##    B 41 21 15
##
## , , replicate = 8
##
##     tension
## wool  L  M  H
##    A 26 30 15
##    B 20 39 16
##
## , , replicate = 9
##
##     tension
## wool  L  M  H
##    A 67 36 26
##    B 44 29 28
```

```
ftable(xt)#summarizes data in a more compact form
```

```
##              replicate  1  2  3  4  5  6  7  8  9
## wool tension
## A    L                 26 30 54 25 70 52 51 26 67
##      M                 18 21 29 17 12 18 35 30 36
##      H                 36 21 24 18 10 43 28 15 26
## B    L                 27 14 29 19 29 31 41 20 44
##      M                 42 26 19 16 39 28 21 39 29
##      H                 20 21 24 17 13 15 15 16 28
```

Size of a data set

```
fakeData <- rnorm(1e5)
object.size(fakeData)
```

```
## 800048 bytes
```

```
print(object.size(fakeData), units = "Mb")#Change units
```

```
## 0.8 Mb
```

Creating New Variables

- Intro:
  - Often the raw data won't have a value you are looking for
  - You will need to transform the data to get the values you would like
  - Usually you will add those values to the data frames you're working with
- Common variables to create
  - Missingness indicators
  - "Cutting up" quantitative variables
  - Applying transforms
- Still using Baltimore restaurant data

```
head(restData) #should exist from previous section
```

```
##                    name zipCode neighborhood councilDistrict policeDistrict
## 1                   410   21206    Frankford               2   NORTHEASTERN
## 2                  1919   21231  Fells Point               1   SOUTHEASTERN
## 3                 SAUTE   21224       Canton               1   SOUTHEASTERN
## 4    #1 CHINESE KITCHEN   21211      Hampden              14       NORTHERN
## 5 #1 chinese restaurant   21223     Millhill               9   SOUTHWESTERN
## 6             19TH HOLE   21218 Clifton Park              14   NORTHEASTERN
##                         Location.1 X2010.Census.Neighborhoods
## 1   4509 BELAIR ROAD\nBaltimore, MD                        NA
## 2      1919 FLEET ST\nBaltimore, MD                        NA
## 3     2844 HUDSON ST\nBaltimore, MD                        NA
## 4    3998 ROLAND AVE\nBaltimore, MD                        NA
## 5 2481 frederick ave\nBaltimore, MD                        NA
## 6    2722 HARFORD RD\nBaltimore, MD                        NA
##   X2010.Census.Wards.Precincts Zip.Codes
## 1                           NA        NA
## 2                           NA        NA
## 3                           NA        NA
## 4                           NA        NA
## 5                           NA        NA
## 6                           NA        NA
```

- Creating sequences (`seq`)
  - Often used to index operations you're going to use on a data set
  - `by` - increases "by" the amount to UB

```
s1 <- seq(1,10, by = 2)
s1
```

```
## [1] 1 3 5 7 9
```

- `length` - Creates a sequence of specified "length", starting at LB and ending at UB

```
s2 <- seq(1,10, length = 3)
s2
```

```
## [1]  1.0  5.5 10.0
```

- `along` - Creates a vector with the same length of given variable

```
x <- c(2, 3, 8, 25, 100)
seq(along = x)
```

```
## [1] 1 2 3 4 5
```

- Subsetting variables
  - Indicates what subset another variable comes from

  - Below we create a new subset, `nearMe`

```
saveLoc <- paste0(getwd(), "/data/restaurants.csv")
restData <- read.csv(saveLoc)
#restData$nearMe <- restdata$neighborhood %in% c("Roland Park", "Homeland") #Assigns to DF its
#table(restData$nearMe)
```

- Creating bianry variables

```
restData$zipWrong <- ifelse(restData$zipCode < 0, TRUE, FALSE)
table(restData$zipWrong, restData$zipCode < 0)
```

```
##
##         FALSE TRUE
##   FALSE  1326    0
##   TRUE      0    1
```

- Creating categorical variables
  - `cut` - cuts subset into sections determined by `breaks` param

```
restData$zipGroups <- cut(restData$zipCode, breaks = quantile(restData$zipCode))
table(restData$zipGroups)
```

```
##
## (-2.123e+04,2.12e+04]  (2.12e+04,2.122e+04] (2.122e+04,2.123e+04]
##                   337                   375                   282
## (2.123e+04,2.129e+04]
##                   332
```

- Easier cutting
  - `cut2` - number of groups you want is determined by the `g` param

```
library(Hmisc)
```

```
## Loading required package: lattice
```

```
## Loading required package: survival
```

```
## Loading required package: Formula

## Loading required package: ggplot2

##
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:plyr':
##
##     is.discrete, summarize

## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
restData$zipGroups <- cut2(restData$zipCode, g = 4)
table(restData$zipGroups)
```

```
##
## [-21226,21205) [ 21205,21220) [ 21220,21227) [ 21227,21287]
##            338            375            300            314
```

Factor variables

- Creating factor variables

```
restData$zcf <- factor(restData$zipCode)
restData$zcf[1:10]
```

```
##  [1] 21206 21231 21224 21211 21223 21218 21205 21211 21205 21231
## 32 Levels: -21226 21201 21202 21205 21206 21207 21208 21209 21210 ... 21287
```

```
class(restData$zcf)
```

```
## [1] "factor"
```

- Levels of factor variables

```
yesno <- sample(c("yes", "no"), size = 10, replace = TRUE)
yesnofac <- factor(yesno, levels = c("yes", "no"))
relevel(yesnofac, ref = "yes")
```

```
##  [1] yes no  yes yes no  yes yes no  yes yes
## Levels: yes no
```

```
as.numeric(yesnofac)
```

```
##  [1] 1 2 1 1 2 1 1 2 1 1
```

- Cutting produced factor variables

```
library(Hmisc)
restData$zipGroups <- cut2(restData$zipCode, g=4)
table(restData$zipGroups)
```

```
##
## [-21226,21205) [ 21205,21220) [ 21220,21227) [ 21227,21287]
##            338           375           300           314
```

- Using the `mutate` function to create a new verison of a variable and apply it to the existing dataset

```r
library(Hmisc); library(plyr)
restData2 <- mutate(restData, zipGroups=cut2(zipCode, g=4))
table(restData2$zipGroups)
```

```
##
## [-21226,21205) [ 21205,21220) [ 21220,21227) [ 21227,21287]
##            338           375           300           314
```

Common transforms

- `abs(x)` - absolute value
- `sqrt(x)` - square root
- `ceiling(x)` - ceiling(3.475) is 4
- `floor(x)` - floor(3.475) is 3
- `round(x, digits = n)` - round(3.75, digits = 2) is 3.48
- `signif(x,digits = n)` - signif(3.475, digits = 2) is 3.5
- `cos(x), sin(x)`, etc - Trig functions
- `log(x)` - natrual logarithm
- `log2(x)`, `log10(x)` - other common logs
- `exp(x)` - exponentiating x
- **A tutorial from Hadley Wickham on plyr**

Reshaping Data

- Start with reshaping

```r
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
## The following objects are masked from 'package:data.table':
##
##     dcast, melt
```

```
head(mtcars)
```

```
##                    mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

- Melting data frames
  - Rownames as the carname is untidy, likewise gear & cyl refer more to the type of car. The variable of intrest is `mpg` or `hp`
  - `id` param is the *fixed variables*

  - `measure.vars` is the… measured variables

```
mtcars$carname <- rownames(mtcars)
carMelt <- melt(mtcars, id = c("carname", "gear", "cyl"), measure.vars = c("mpg", "hp"))
head(carMelt, n = 3)
```

```
##         carname gear cyl variable value
## 1     Mazda RX4    4   6      mpg  21.0
## 2 Mazda RX4 Wag    4   6      mpg  21.0
## 3    Datsun 710    4   4      mpg  22.8
```

```
tail(carMelt, n = 3)
```

```
##          carname gear cyl variable value
## 62  Ferrari Dino    5   6       hp   175
## 63 Maserati Bora    5   8       hp   335
## 64    Volvo 142E    4   4       hp   109
```

- Casting data frames

```
cylData <- dcast(carMelt, cyl ~ variable)
```

```
## Aggregation function missing: defaulting to length
```

```
cylData #Puts cyl in left, then variables in following cols
```

```
##   cyl mpg hp
## 1   4  11 11
## 2   6   7  7
## 3   8  14 14
```

```r
#Can compute transforms at this point
cylData <- dcast(carMelt, cyl ~ variable,mean)
cylData
```

```
##   cyl      mpg        hp
## 1   4 26.66364  82.63636
## 2   6 19.74286 122.28571
## 3   8 15.10000 209.21429
```

- Averaging values

```r
head(InsectSprays)
```

```
##   count spray
## 1    10     A
## 2     7     A
## 3    20     A
## 4    14     A
## 5    14     A
## 6    12     A
```

```r
#Take sum of count by the `spray`
tapply(InsectSprays$count, InsectSprays$spray, sum)
```

```
##   A   B   C   D   E   F
## 174 184  25  59  42 200
```

- Another way with `split` then `lapply`

```r
spIns <- split(InsectSprays$count, InsectSprays$spray)
spIns
```

```
## $A
##  [1] 10  7 20 14 14 12 10 23 17 20 14 13
##
## $B
##  [1] 11 17 21 11 16 14 17 17 19 21  7 13
##
## $C
##  [1] 0 1 7 2 3 1 2 1 3 0 1 4
##
## $D
##  [1]  3  5 12  6  4  3  5  5  5  5  2  4
##
## $E
##  [1] 3 5 3 5 3 6 1 1 3 2 6 4
##
## $F
##  [1] 11  9 15 22 15 16 13 10 26 26 24 13
```

```r
sprCount <- lapply(spIns, sum)
sprCount
```

```
## $A
## [1] 174
##
## $B
## [1] 184
##
## $C
## [1] 25
##
## $D
## [1] 59
##
## $E
## [1] 42
##
## $F
## [1] 200
```

- Another way - combine

```r
unlist(sprCount)
```

```
##   A   B   C   D   E   F
## 174 184  25  59  42 200
```

```r
sapply(spIns, sum)
```

```
##   A   B   C   D   E   F
## 174 184  25  59  42 200
```

- Another way with `plyr` package

```r
#. is needed so quotes aren't

#sapply(InsectSprays,.(spray), summarize, sum = sum(count))
```

- Creating a new variable

```r
# spraySums <- ddply(InsectSprays,.(spray), summarize, sum = ave(count, FUN = sum))
# dim(spraySums)#Same num as InsectSprays
# head(spraySums)
```

- Some other functions
  - `acast` - for casting as multi-dimensional arrays
  - `arrange` - for faster reordering without using `order()` commands
  - `mutate` - adding new variables

Managing Data Frames with dplyr - Intro

- Verbs to be covered:
  - `select` - return a subset of the columns of a data frame

  - `filter` - extract a subset of rows from a data frame based on logical conditions

  - `arrange` - reorder rows of a data frame

  - `rename` - rename variables in a data frame

  - `mutate` - add new variables/columns or transform existing variables

  - `summarise` / `summarize` - generate summary statistics of different variabels in the data frame, possibly within `strata`
  - `print` - prevents a lot of data printing to the console
- `dplyr` is used to work with data frames, the key structure in statistics and `R`
  - part of the *tidyverse*, as such it was developed by Hadley Wickham

  - An optimized & distilled version of the `plyr` package (Also by Hadley)
    - Does not provide any "new" functionality

  - Is **very** fast, as many key operations are coded in `C++`

- Arguments
  - The first argument is always a data frame.
  - The subsequent arguments describe what to do with it
    - You can refer to columns in the data frame directly, without the `$` operator (just use the names)

- Data must be properly formatted and annotated (tidy) for this to all be useful

- A new data.frame is always returned

Managing Data Frames with dplyr - Basic Tools

- When loading `dplyr` a few warning will appear telling of masked functions

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:Hmisc':
##
##     src, summarize

## The following objects are masked from 'package:plyr':
##
```

```
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

## The following objects are masked from 'package:data.table':
##
##     between, first, last

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
#Data used in examples
url <- "https://github.com/DataScienceSpecialization/courses/blob/master/03_GettingData/dplyr/c
dir <- "data"
filename <- "Chicago.rds"
ogDir <- getwd()
saveLoc <- paste(ogDir, dir, filename, sep = "/")
download.file(url, saveLoc, method = "curl", extra = "-L")

setwd(paste(ogDir, dir, sep = "/"))
chicago <- readRDS(file = saveLoc)
setwd(ogDir)
```

```r
#Checking out the data set
dim(chicago)
```

```
## [1] 6940    8
```

```r
str(chicago)
```

```
## 'data.frame':    6940 obs. of  8 variables:
##  $ city      : chr  "chic" "chic" "chic" "chic" ...
##  $ tmpd      : num  31.5 33 33 29 32 40 34.5 29 26.5 32.5 ...
##  $ dptp      : num  31.5 29.9 27.4 28.6 28.9 ...
##  $ date      : Date, format: "1987-01-01" "1987-01-02" ...
##  $ pm25tmean2: num  NA NA NA NA NA NA NA NA NA NA ...
##  $ pm10tmean2: num  34 NA 34.2 47 NA ...
##  $ o3tmean2  : num  4.25 3.3 3.33 4.38 4.75 ...
##  $ no2tmean2 : num  20 23.2 23.8 30.4 30.3 ...
```

```r
names(chicago)
```

```
## [1] "city"       "tmpd"       "dptp"       "date"       "pm25tmean2"
## [6] "pm10tmean2" "o3tmean2"   "no2tmean2"
```

```r
print(paste("This data is from ", chicago$date[1], " to ", chicago$date[length(chicago[,1])], s
```

```
## [1] "This data is from 1987-01-01 to 2005-12-31"
```

- `select` function

```r
#dplyr allows referencing to colnames as if they were ordinal objects
head(select(chicago, city:dptp))
```

```
##   city tmpd   dptp
## 1 chic 31.5 31.500
## 2 chic 33.0 29.875
## 3 chic 33.0 27.375
## 4 chic 29.0 28.625
## 5 chic 32.0 28.875
## 6 chic 40.0 35.125
```

```r
#One can also use the minus sign
head(select(chicago, -(city:dptp)))
```

```
##         date pm25tmean2 pm10tmean2 o3tmean2 no2tmean2
## 1 1987-01-01         NA   34.00000 4.250000  19.98810
## 2 1987-01-02         NA         NA 3.304348  23.19099
## 3 1987-01-03         NA   34.16667 3.333333  23.81548
## 4 1987-01-04         NA   47.00000 4.375000  30.43452
## 5 1987-01-05         NA         NA 4.750000  30.33333
## 6 1987-01-06         NA   48.00000 5.833333  25.77233
```

```r
#Base R equivelent
i <- match("city", names(chicago))
j <- match("dptp", names(chicago))
head(chicago[, -(i:j)])
```

```
##         date pm25tmean2 pm10tmean2 o3tmean2 no2tmean2
## 1 1987-01-01         NA   34.00000 4.250000  19.98810
## 2 1987-01-02         NA         NA 3.304348  23.19099
## 3 1987-01-03         NA   34.16667 3.333333  23.81548
## 4 1987-01-04         NA   47.00000 4.375000  30.43452
## 5 1987-01-05         NA         NA 4.750000  30.33333
## 6 1987-01-06         NA   48.00000 5.833333  25.77233
```

- `filter` function (keeps the `TRUE` parts of logical)

```r
chic.f <- filter(chicago, pm25tmean2 > 30)
head(chic.f, 10)
```

```
##    city tmpd dptp       date pm25tmean2 pm10tmean2  o3tmean2 no2tmean2
## 1  chic   23 21.9 1998-01-17      38.10   32.46154  3.180556  25.30000
## 2  chic   28 25.8 1998-01-23      33.95   38.69231  1.750000  29.37630
## 3  chic   55 51.3 1998-04-30      39.40   34.00000 10.786232  25.31310
## 4  chic   59 53.7 1998-05-01      35.40   28.50000 14.295125  31.42905
## 5  chic   57 52.0 1998-05-02      33.30   35.00000 20.662879  26.79861
## 6  chic   57 56.0 1998-05-07      32.10   34.50000 24.270422  33.99167
## 7  chic   75 65.8 1998-05-15      56.50   91.00000 38.573007  29.03261
## 8  chic   61 59.0 1998-06-09      33.80   26.00000 17.890810  25.49668
```

```
## 9  chic   73 60.3 1998-07-13      30.30   64.50000 37.018865  37.93056
## 10 chic   78 67.1 1998-07-14      41.40   75.00000 40.080902  32.59054
```

```
#Can accept more complex logicals
chic.f <- filter(chicago, pm25tmean2 > 30 & tmpd > 80)
head(chic.f)
```

```
##   city tmpd dptp       date pm25tmean2 pm10tmean2 o3tmean2 no2tmean2
## 1 chic   81 71.2 1998-08-23    39.6000       59.0 45.86364  14.32639
## 2 chic   81 70.4 1998-09-06    31.5000       50.5 50.66250  20.31250
## 3 chic   82 72.2 2001-07-20    32.3000       58.5 33.00380  33.67500
## 4 chic   84 72.9 2001-08-01    43.7000       81.5 45.17736  27.44239
## 5 chic   85 72.6 2001-08-08    38.8375       70.0 37.98047  27.62743
## 6 chic   84 72.6 2001-08-09    38.2000       66.0 36.73245  26.46742
```

- arrange function

```
chicago <- arrange(chicago, date)
head(chicago)
```

```
##   city tmpd   dptp       date pm25tmean2 pm10tmean2 o3tmean2 no2tmean2
## 1 chic 31.5 31.500 1987-01-01         NA   34.00000 4.250000  19.98810
## 2 chic 33.0 29.875 1987-01-02         NA         NA 3.304348  23.19099
## 3 chic 33.0 27.375 1987-01-03         NA   34.16667 3.333333  23.81548
## 4 chic 29.0 28.625 1987-01-04         NA   47.00000 4.375000  30.43452
## 5 chic 32.0 28.875 1987-01-05         NA         NA 4.750000  30.33333
## 6 chic 40.0 35.125 1987-01-06         NA   48.00000 5.833333  25.77233
```

```
chicago <- arrange(chicago, desc(date))
head(chicago)
```

```
##   city tmpd dptp       date pm25tmean2 pm10tmean2  o3tmean2 no2tmean2
## 1 chic   35 30.1 2005-12-31   15.00000       23.5  2.531250  13.25000
## 2 chic   36 31.0 2005-12-30   15.05714       19.2  3.034420  22.80556
## 3 chic   35 29.4 2005-12-29    7.45000       23.5  6.794837  19.97222
## 4 chic   37 34.5 2005-12-28   17.75000       27.5  3.260417  19.28563
## 5 chic   40 33.6 2005-12-27   23.56000       27.0  4.468750  23.50000
## 6 chic   35 29.6 2005-12-26    8.40000        8.5 14.041667  16.81944
```

- rename function

```
#For reproducability
if(any(colnames(chicago) == "pm25")){
  chicago <- rename(chicago, pm25tmean2 = pm25)}
if(any(colnames(chicago) == "dewpoint")){
  chicago <- rename(chicago, dptp = dewpoint)}

#Showing rename function
chicago <- rename(chicago, pm25 = pm25tmean2, dewpoint = dptp)
head(chicago)
```

```
##   city tmpd dewpoint       date     pm25 pm10tmean2  o3tmean2 no2tmean2
```

```
## 1 chic   35     30.1 2005-12-31 15.00000       23.5  2.531250  13.25000
## 2 chic   36     31.0 2005-12-30 15.05714       19.2  3.034420  22.80556
## 3 chic   35     29.4 2005-12-29  7.45000       23.5  6.794837  19.97222
## 4 chic   37     34.5 2005-12-28 17.75000       27.5  3.260417  19.28563
## 5 chic   40     33.6 2005-12-27 23.56000       27.0  4.468750  23.50000
## 6 chic   35     29.6 2005-12-26  8.40000        8.5 14.041667  16.81944
```

- mutate function

```
chicago <- mutate(chicago, pm25detrend = pm25-mean(pm25, na.rm = TRUE))
head(select(chicago, pm25, pm25detrend))
```

```
##       pm25 pm25detrend
## 1 15.00000   -1.230958
## 2 15.05714   -1.173815
## 3  7.45000   -8.780958
## 4 17.75000    1.519042
## 5 23.56000    7.329042
## 6  8.40000   -7.830958
```

- groupby function

```
#First we'll create a tempcat variable to show if the day was hot or cold
chicago <- mutate(chicago, tempcat = factor(1 * (tmpd > 80), labels = c("cold", "hot")))


hotcold <- group_by(chicago, tempcat)
```

```
## Warning: Factor `tempcat` contains implicit NA, consider using
## `forcats::fct_explicit_na`
```

```
#Now summarize will split info by tempcat factor
summarize(hotcold, pm25 = mean(pm25, na.rm = TRUE), o3 = max(o3tmean2), no2 = median(no2tmean2
```

```
## # A tibble: 3 x 4
##   tempcat  pm25    o3   no2
##   <fct>   <dbl> <dbl> <dbl>
## 1 cold     16.0 66.6   24.5
## 2 hot      26.5 63.0   24.9
## 3 <NA>     47.7  9.42  37.4
```

```
chicago <- mutate(chicago, year = as.POSIXlt(date)$year + 1900)
years <- group_by(chicago, year)
summarize(years, pm25 = mean(pm25, na.rm = TRUE), o3 = max(o3tmean2), no2 = median(no2tmean2))
```

```
## # A tibble: 19 x 4
##     year  pm25    o3   no2
##    <dbl> <dbl> <dbl> <dbl>
##  1  1987 NaN    63.0  23.5
##  2  1988 NaN    61.7  24.5
##  3  1989 NaN    59.7  26.1
##  4  1990 NaN    52.2  22.6
```

```
## 5   1991 NaN    63.1  21.4
## 6   1992 NaN    50.8  24.8
## 7   1993 NaN    44.3  25.8
## 8   1994 NaN    52.2  28.5
## 9   1995 NaN    66.6  27.3
## 10  1996 NaN    58.4  26.4
## 11  1997 NaN    56.5  25.5
## 12  1998  18.3  50.7  24.6
## 13  1999  18.5  57.5  24.7
## 14  2000  16.9  55.8  23.5
## 15  2001  16.9  51.8  25.1
## 16  2002  15.3  54.9  22.7
## 17  2003  15.2  56.2  24.6
## 18  2004  14.6  44.5  23.4
## 19  2005  16.2  58.8  22.6
```

- Chaining operations together with *pipe* `%>%`
  - Don't need to specify data frame with *pipe* because it's implied

  - Helps you not have to assign additional temp variables

```
chicago %>% mutate(month = as.POSIXlt(date)$mon + 1) %>% group_by(month) %>% summarize(pm25 = m
 na.rm = TRUE), o3 = max(o3tmean2), no2 = median(no2tmean2))
```

```
## # A tibble: 12 x 4
##    month  pm25    o3   no2
##    <dbl> <dbl> <dbl> <dbl>
##  1     1  17.8  28.2  25.4
##  2     2  20.4  37.4  26.8
##  3     3  17.4  39.0  26.8
##  4     4  13.9  47.9  25.0
##  5     5  14.1  52.8  24.2
##  6     6  15.9  66.6  25.0
##  7     7  16.6  59.5  22.4
##  8     8  16.9  54.0  23.0
##  9     9  15.9  57.5  24.5
## 10    10  14.2  47.1  24.2
## 11    11  15.2  29.5  23.6
## 12    12  17.5  27.7  24.5
```

- Additional benefits to `dplyr`
  - `dplyr` can work with other data frame "backends"

  - works with `data.table` which is for large data sets

  - allows one to interact with SQL interface for relational databases via the `DBI` package

Merging Data

- Example will use: Peer review data

```r
saveDir <- paste(getwd(), "/data", sep = "")
if (!file.exists(saveDir)) {dir.create(saveDir)}
reviewUrl <- "https://raw.githubusercontent.com/jtleek/dataanalysis/master/week2/007summarizing
solUrl <- "https://raw.githubusercontent.com/jtleek/dataanalysis/master/week2/007summarizingDa

saveLoc <- paste(saveDir, "/review.csv", sep = "")
download.file(reviewUrl, saveLoc, method = "curl")
reviews <- read.csv(saveLoc)

saveLoc <- paste(saveDir, "/solutions.csv", sep = "")
download.file(solUrl, saveLoc, method = "curl")
solutions <- read.csv(saveLoc)

rm(saveLoc)#Because it's value was somewhat ambiguious

head(reviews, 2)
```

```
##   id solution_id reviewer_id      start       stop time_left accept
## 1  1           3          27 1304095698 1304095758      1754      1
## 2  2           4          22 1304095188 1304095206      2306      1
```

```r
head(solutions, 2)
```

```
##   id problem_id subject_id      start       stop time_left answer
## 1  1        156         29 1304095119 1304095169      2343      B
## 2  2        269         25 1304095119 1304095183      2329      C
```

- Merging data frames with the `merge()` function
  - Important parameters:
  - `x` - First data frame
  - `y` - Second data frame
  - `by` - Default is to merge by columns with common names
  - `by.x` -
  - `by.y` -
  - `all` - Logical for if all names should be included
    - (If `x` has a name `y` does not then y will have `NA` for the values in that missing column)

```r
mergedData <- merge(reviews, solutions, by.x = "solution_id", by.y = "id", all = TRUE)
head(mergedData)
```

```
##   solution_id id reviewer_id    start.x     stop.x time_left.x accept
## 1           1  4          26 1304095267 1304095423        2089      1
## 2           2  6          29 1304095471 1304095513        1999      1
## 3           3  1          27 1304095698 1304095758        1754      1
## 4           4  2          22 1304095188 1304095206        2306      1
## 5           5  3          28 1304095276 1304095320        2192      1
```

```
## 6           6 16          22 1304095303 1304095471        2041      1
##   problem_id subject_id    start.y     stop.y time_left.y answer
## 1        156         29 1304095119 1304095169        2343      B
## 2        269         25 1304095119 1304095183        2329      C
## 3         34         22 1304095127 1304095146        2366      C
## 4         19         23 1304095127 1304095150        2362      D
## 5        605         26 1304095127 1304095167        2345      A
## 6        384         27 1304095131 1304095270        2242      C
```

- Default - merge all common column names

```r
intersect(names(solutions), names(reviews)) #Displays == names
```

```
## [1] "id"        "start"     "stop"      "time_left"
```

```r
mergedData2 <- merge(reviews, solutions, all = TRUE)
head(mergedData2) #Start and stop times were diffrent in datasets so they make diffrent rows wi
```

```
##   id      start       stop time_left solution_id reviewer_id accept problem_id
## 1  1 1304095119 1304095169      2343          NA          NA     NA        156
## 2  1 1304095698 1304095758      1754           3          27      1         NA
## 3  2 1304095119 1304095183      2329          NA          NA     NA        269
## 4  2 1304095188 1304095206      2306           4          22      1         NA
## 5  3 1304095127 1304095146      2366          NA          NA     NA         34
## 6  3 1304095276 1304095320      2192           5          28      1         NA
##   subject_id answer
## 1         29      B
## 2         NA   <NA>
## 3         25      C
## 4         NA   <NA>
## 5         22      C
## 6         NA   <NA>
```

- Using `join` in the `plyr` package
  - Faster, but less full featured - defaults to left join, see help file for more

```r
df1 <- data.frame(id = sample(1:10), x = rnorm(10))
df2 <- data.frame(id = sample(1:10), y = rnorm(10))
df3 <- data.frame(id = sample(1:10), z = rnorm(10))
dfList = list(df1, df2, df3)
join_all(dfList) #Merges datasets by common variable
```

```
## Joining by: id
## Joining by: id
```

```
##    id          x          y           z
## 1  10  0.2645152 -1.8963536  1.16579466
## 2   7  1.9296758 -0.1263317 -0.75025267
## 3   1 -0.7309874 -0.7388428  2.04580330
## 4   8  0.5428679  0.2464850  0.88137624
## 5   5  0.5144002  2.6998482  0.07436382
```

```
## 6   2 -0.5557001  1.5638946  0.92514533
## 7   4  1.1938073 -0.3393521 -2.99431792
## 8   3  0.4106490  0.9995930  0.64047066
## 9   9  1.2332349 -0.5516383 -1.06584640
## 10  6  0.6730770  1.9963144  0.93444780
```

- More on merging data
  - **The quick R data merging page**
  - **plyr information (Hadley's site)**
  - **Types of joins**

Lessons with `swirl()`

Manipulating Data with dplyr

- dplyr can work with: data tables, databases, and multidimensional arrays; in addition to, the prefered, data frames.
- This lesson work with a `csv` data set which I've saved and shall assign to `mydf` now

```
saveLoc <- paste(getwd(), "/data/CRANpackages.csv", sep = "")
mydf <- read.csv(saveLoc, stringsAsFactors = FALSE)
cran <- tbl_df(mydf)
```

- opening `dplyr` library and checking the version

```
library(dplyr)
packageVersion("dplyr")
```

```
## [1] '0.8.5'
```

- data frame tbl, `tbl_df()`
  - The main advantage to using a tbl_df over a regular data frame is the printing.
    - limits the volume of data that is outputted
    - highlights `NA` values
    - only prints as many columns as neatly fit in the console

```
cran
```

```
## # A tibble: 225,468 x 12
##      X.1     X date  time    size r_version r_arch r_os  package version country
##    <int> <int> <chr> <chr>  <int> <chr>     <chr>  <chr> <chr>   <chr>   <chr>
##  1     1     1 2014~ 00:5~ 8.06e4 3.1.0     x86_64 ming~ htmlto~ 0.2.4   US
##  2     2     2 2014~ 00:5~ 3.22e5 3.1.0     x86_64 ming~ tseries 0.10-32 US
##  3     3     3 2014~ 00:4~ 7.48e5 3.1.0     x86_64 linu~ party   1.0-15  US
##  4     4     4 2014~ 00:4~ 6.06e5 3.1.0     x86_64 linu~ Hmisc   3.14-4  US
##  5     5     5 2014~ 00:4~ 7.98e4 3.0.2     x86_64 linu~ digest  0.6.4   CA
##  6     6     6 2014~ 00:4~ 7.77e4 3.1.0     x86_64 linu~ random~ 4.6-7   US
##  7     7     7 2014~ 00:4~ 3.94e5 3.1.0     x86_64 linu~ plyr    1.8.1   US
##  8     8     8 2014~ 00:4~ 2.82e4 3.0.2     x86_64 linu~ whisker 0.3-2   US
##  9     9     9 2014~ 00:5~ 5.93e3 <NA>      <NA>   <NA>  Rcpp    0.10.4  CN
## 10    10    10 2014~ 00:1~ 2.21e6 3.0.2     x86_64 linu~ hfligh~ 0.1     US
```

```
## # ... with 225,458 more rows, and 1 more variable: ip_id <int>
```

- "The dplyr philosophy is to have small functions that each do one thing well." Specifically there are five 'verbs' that cover most fundamental data manipulation tasks:
  - `select()` - subset columns
    - knows it's parameters are not objects, but rather colnames. And will throw a fit if they aren't actually names in the data set
    - orders output's columns by the order they're passed into the function

    - one can also use the `:` operator to refer to a sequence of columns in either ascending or descending order

```r
select(cran, ip_id, package, country)
```

```
## # A tibble: 225,468 x 3
##    ip_id package      country
##    <int> <chr>        <chr>
##  1     1 htmltools    US
##  2     2 tseries      US
##  3     3 party        US
##  4     3 Hmisc        US
##  5     4 digest       CA
##  6     3 randomForest US
##  7     3 plyr         US
##  8     5 whisker      US
##  9     6 Rcpp         CN
## 10     7 hflights     US
## # ... with 225,458 more rows
```

```r
select(cran, r_arch:country) #Ascending
```

```
## # A tibble: 225,468 x 5
##    r_arch r_os      package      version country
##    <chr>  <chr>     <chr>        <chr>   <chr>
##  1 x86_64 mingw32   htmltools    0.2.4   US
##  2 x86_64 mingw32   tseries      0.10-32 US
##  3 x86_64 linux-gnu party        1.0-15  US
##  4 x86_64 linux-gnu Hmisc        3.14-4  US
##  5 x86_64 linux-gnu digest       0.6.4   CA
##  6 x86_64 linux-gnu randomForest 4.6-7   US
##  7 x86_64 linux-gnu plyr         1.8.1   US
##  8 x86_64 linux-gnu whisker      0.3-2   US
##  9 <NA>   <NA>      Rcpp         0.10.4  CN
## 10 x86_64 linux-gnu hflights     0.1     US
## # ... with 225,458 more rows
```

```r
select(cran, country:r_arch) #Decending
```

```
## # A tibble: 225,468 x 5
##    country version package      r_os      r_arch
```

```
##    <chr>   <chr>   <chr>        <chr>     <chr>
##  1 US      0.2.4   htmltools    mingw32   x86_64
##  2 US      0.10-32 tseries      mingw32   x86_64
##  3 US      1.0-15  party        linux-gnu x86_64
##  4 US      3.14-4  Hmisc        linux-gnu x86_64
##  5 CA      0.6.4   digest       linux-gnu x86_64
##  6 US      4.6-7   randomForest linux-gnu x86_64
##  7 US      1.8.1   plyr         linux-gnu x86_64
##  8 US      0.3-2   whisker      linux-gnu x86_64
##  9 CN      0.10.4  Rcpp         <NA>      <NA>
## 10 US      0.1     hflights     linux-gnu x86_64
## # ... with 225,458 more rows
```

```
select(cran, -time) #all but-
```

```
## # A tibble: 225,468 x 11
##      X.1     X date    size r_version r_arch r_os  package version country ip_id
##    <int> <int> <chr>  <int> <chr>     <chr>  <chr> <chr>   <chr>   <chr>   <int>
##  1     1     1 2014~ 8.06e4 3.1.0     x86_64 ming~ htmlto~ 0.2.4   US          1
##  2     2     2 2014~ 3.22e5 3.1.0     x86_64 ming~ tseries 0.10-32 US          2
##  3     3     3 2014~ 7.48e5 3.1.0     x86_64 linu~ party   1.0-15  US          3
##  4     4     4 2014~ 6.06e5 3.1.0     x86_64 linu~ Hmisc   3.14-4  US          3
##  5     5     5 2014~ 7.98e4 3.0.2     x86_64 linu~ digest  0.6.4   CA          4
##  6     6     6 2014~ 7.77e4 3.1.0     x86_64 linu~ random~ 4.6-7   US          3
##  7     7     7 2014~ 3.94e5 3.1.0     x86_64 linu~ plyr    1.8.1   US          3
##  8     8     8 2014~ 2.82e4 3.0.2     x86_64 linu~ whisker 0.3-2   US          5
##  9     9     9 2014~ 5.93e3 <NA>      <NA>   <NA>  Rcpp    0.10.4  CN          6
## 10    10    10 2014~ 2.21e6 3.0.2     x86_64 linu~ hfligh~ 0.1     US          7
## # ... with 225,458 more rows
```

```
select(cran, -(X:size)) #omitting a seq of cols
```

```
## # A tibble: 225,468 x 8
##      X.1 r_version r_arch r_os      package      version country ip_id
##    <int> <chr>     <chr>  <chr>     <chr>        <chr>   <chr>   <int>
##  1     1 3.1.0     x86_64 mingw32   htmltools    0.2.4   US          1
##  2     2 3.1.0     x86_64 mingw32   tseries      0.10-32 US          2
##  3     3 3.1.0     x86_64 linux-gnu party        1.0-15  US          3
##  4     4 3.1.0     x86_64 linux-gnu Hmisc        3.14-4  US          3
##  5     5 3.0.2     x86_64 linux-gnu digest       0.6.4   CA          4
##  6     6 3.1.0     x86_64 linux-gnu randomForest 4.6-7   US          3
##  7     7 3.1.0     x86_64 linux-gnu plyr         1.8.1   US          3
##  8     8 3.0.2     x86_64 linux-gnu whisker      0.3-2   US          5
##  9     9 <NA>      <NA>   <NA>      Rcpp         0.10.4  CN          6
## 10    10 3.0.2     x86_64 linux-gnu hflights     0.1     US          7
## # ... with 225,458 more rows
```

- `filter()` - subset rows by `Comparison` logicals (Use `?Comparison` to learn more)

```r
filter(cran, package == "swirl")
```

```
## # A tibble: 820 x 12
##      X.1     X date  time    size r_version r_arch r_os  package version country
##    <int> <int> <chr> <chr>  <int> <chr>     <chr>  <chr> <chr>   <chr>   <chr>
##  1    27    27 2014~ 00:1~ 105350 3.0.2     x86_64 ming~ swirl   2.2.9   US
##  2   156   156 2014~ 00:2~  41261 3.1.0     x86_64 linu~ swirl   2.2.9   US
##  3   358   358 2014~ 00:1~ 105335 2.15.2    x86_64 ming~ swirl   2.2.9   CA
##  4   593   593 2014~ 00:5~ 105465 3.1.0     x86_64 darw~ swirl   2.2.9   MX
##  5   831   831 2014~ 00:5~ 105335 3.0.3     x86_64 ming~ swirl   2.2.9   US
##  6   997   997 2014~ 00:3~  41261 3.1.0     x86_64 ming~ swirl   2.2.9   US
##  7  1023  1023 2014~ 00:3~ 106393 3.1.0     x86_64 ming~ swirl   2.2.9   BR
##  8  1144  1144 2014~ 00:0~ 106534 3.0.2     x86_64 linu~ swirl   2.2.9   US
##  9  1402  1402 2014~ 00:4~  41261 3.1.0     i386   ming~ swirl   2.2.9   US
## 10  1424  1424 2014~ 00:4~ 106393 3.1.0     x86_64 linu~ swirl   2.2.9   US
## # ... with 810 more rows, and 1 more variable: ip_id <int>
```

```r
#Multiple parameters are allowed, only all(TRUE) rows will be returned
filter(cran, r_version == "3.1.1", country == "US")
```

```
## # A tibble: 1,588 x 12
##      X.1     X date  time    size r_version r_arch r_os  package version country
##    <int> <int> <chr> <chr>  <int> <chr>     <chr>  <chr> <chr>   <chr>   <chr>
##  1  2216  2216 2014~ 00:4~ 3.85e5 3.1.1     x86_64 darw~ colors~ 1.2-4   US
##  2 17332 17332 2014~ 03:3~ 1.97e5 3.1.1     x86_64 darw~ httr    0.3     US
##  3 17465 17465 2014~ 03:2~ 2.33e4 3.1.1     x86_64 darw~ snow    0.3-13  US
##  4 18844 18844 2014~ 03:5~ 1.91e5 3.1.1     x86_64 darw~ maxLik  1.2-0   US
##  5 30182 30182 2014~ 04:1~ 7.77e4 3.1.1     i386   ming~ random~ 4.6-7   US
##  6 30193 30193 2014~ 04:0~ 2.35e6 3.1.1     i386   ming~ ggplot2 1.0.0   US
##  7 30195 30195 2014~ 04:0~ 2.99e5 3.1.1     i386   ming~ fExtre~ 3010.81 US
##  8 30217 30217 2014~ 04:3~ 5.68e5 3.1.1     i386   ming~ rJava   0.9-6   US
##  9 30245 30245 2014~ 04:1~ 5.27e5 3.1.1     i386   ming~ LPCM    0.44-8  US
## 10 30354 30354 2014~ 04:3~ 1.76e6 3.1.1     i386   ming~ mgcv    1.8-1   US
## # ... with 1,578 more rows, and 1 more variable: ip_id <int>
```

```r
#Demo of OR conditional
filter(cran, country == "US" | country == "IN")
```

```
## # A tibble: 95,283 x 12
##      X.1     X date  time    size r_version r_arch r_os  package version country
##    <int> <int> <chr> <chr>  <int> <chr>     <chr>  <chr> <chr>   <chr>   <chr>
##  1     1     1 2014~ 00:5~ 8.06e4 3.1.0     x86_64 ming~ htmlto~ 0.2.4   US
##  2     2     2 2014~ 00:5~ 3.22e5 3.1.0     x86_64 ming~ tseries 0.10-32 US
##  3     3     3 2014~ 00:4~ 7.48e5 3.1.0     x86_64 linu~ party   1.0-15  US
##  4     4     4 2014~ 00:4~ 6.06e5 3.1.0     x86_64 linu~ Hmisc   3.14-4  US
##  5     6     6 2014~ 00:4~ 7.77e4 3.1.0     x86_64 linu~ random~ 4.6-7   US
##  6     7     7 2014~ 00:4~ 3.94e5 3.1.0     x86_64 linu~ plyr    1.8.1   US
##  7     8     8 2014~ 00:4~ 2.82e4 3.0.2     x86_64 linu~ whisker 0.3-2   US
##  8    10    10 2014~ 00:1~ 2.21e6 3.0.2     x86_64 linu~ hfligh~ 0.1     US
```

```
## 9     11     11 2014~ 00:1~ 5.27e5 3.0.2     x86_64 linu~ LPCM    0.44-8  US
## 10    12     12 2014~ 00:1~ 2.35e6 2.14.1    x86_64 linu~ ggplot2 1.0.0   US
## # ... with 95,273 more rows, and 1 more variable: ip_id <int>
```

```
#Numerics don't need quote && AND conditionals can just use separate params
filter(cran, size > 100500, r_os == "linux-gnu")
```

```
## # A tibble: 33,683 x 12
##      X.1     X date  time    size r_version r_arch r_os  package version country
##    <int> <int> <chr> <chr>  <int> <chr>     <chr>  <chr> <chr>   <chr>   <chr>
##  1     3     3 2014~ 00:4~ 7.48e5 3.1.0     x86_64 linu~ party   1.0-15  US
##  2     4     4 2014~ 00:4~ 6.06e5 3.1.0     x86_64 linu~ Hmisc   3.14-4  US
##  3     7     7 2014~ 00:4~ 3.94e5 3.1.0     x86_64 linu~ plyr    1.8.1   US
##  4    10    10 2014~ 00:1~ 2.21e6 3.0.2     x86_64 linu~ hfligh~ 0.1     US
##  5    11    11 2014~ 00:1~ 5.27e5 3.0.2     x86_64 linu~ LPCM    0.44-8  US
##  6    12    12 2014~ 00:1~ 2.35e6 2.14.1    x86_64 linu~ ggplot2 1.0.0   US
##  7    14    14 2014~ 00:1~ 3.10e6 3.0.2     x86_64 linu~ Rcpp    0.9.7   VE
##  8    15    15 2014~ 00:1~ 5.68e5 3.1.0     x86_64 linu~ rJava   0.9-6   US
##  9    16    16 2014~ 00:1~ 1.60e6 3.1.0     x86_64 linu~ RSQLite 0.11.4  US
## 10    18    18 2014~ 00:2~ 1.87e5 3.1.0     x86_64 linu~ ipred   0.9-3   DE
## # ... with 33,673 more rows, and 1 more variable: ip_id <int>
```

```
#Filtering out NAs
filter(cran, !is.na(r_version))
```

```
## # A tibble: 207,205 x 12
##      X.1     X date  time    size r_version r_arch r_os  package version country
##    <int> <int> <chr> <chr>  <int> <chr>     <chr>  <chr> <chr>   <chr>   <chr>
##  1     1     1 2014~ 00:5~ 8.06e4 3.1.0     x86_64 ming~ htmlto~ 0.2.4   US
##  2     2     2 2014~ 00:5~ 3.22e5 3.1.0     x86_64 ming~ tseries 0.10-32 US
##  3     3     3 2014~ 00:4~ 7.48e5 3.1.0     x86_64 linu~ party   1.0-15  US
##  4     4     4 2014~ 00:4~ 6.06e5 3.1.0     x86_64 linu~ Hmisc   3.14-4  US
##  5     5     5 2014~ 00:4~ 7.98e4 3.0.2     x86_64 linu~ digest  0.6.4   CA
##  6     6     6 2014~ 00:4~ 7.77e4 3.1.0     x86_64 linu~ random~ 4.6-7   US
##  7     7     7 2014~ 00:4~ 3.94e5 3.1.0     x86_64 linu~ plyr    1.8.1   US
##  8     8     8 2014~ 00:4~ 2.82e4 3.0.2     x86_64 linu~ whisker 0.3-2   US
##  9    10    10 2014~ 00:1~ 2.21e6 3.0.2     x86_64 linu~ hfligh~ 0.1     US
## 10    11    11 2014~ 00:1~ 5.27e5 3.0.2     x86_64 linu~ LPCM    0.44-8  US
## # ... with 207,195 more rows, and 1 more variable: ip_id <int>
```

- `arrange()` - orders the rows of a dataset accoring to the values of a particular variable

```
#Demonstrate with a subset of cran
cran2 <- select(cran, size:ip_id)
arrange(cran2, ip_id)#Order rows by ip_id
```

```
## # A tibble: 225,468 x 8
##      size r_version r_arch r_os          package     version country ip_id
##     <int> <chr>     <chr>  <chr>         <chr>       <chr>   <chr>   <int>
##  1  80589 3.1.0     x86_64 mingw32       htmltools   0.2.4   US          1
##  2 180562 3.0.2     x86_64 mingw32       yaml        2.1.13  US          1
```

```
##  3 190120 3.1.0     i386   mingw32      babel      0.2-6   US            1
##  4 321767 3.1.0     x86_64 mingw32      tseries    0.10-32 US            2
##  5  52281 3.0.3     x86_64 darwin10.8.0 quadprog   1.5-5   US            2
##  6 876702 3.1.0     x86_64 linux-gnu    zoo        1.7-11  US            2
##  7 321764 3.0.2     x86_64 linux-gnu    tseries    0.10-32 US            2
##  8 876702 3.1.0     x86_64 linux-gnu    zoo        1.7-11  US            2
##  9 321768 3.1.0     x86_64 mingw32      tseries    0.10-32 US            2
## 10 784093 3.1.0     x86_64 linux-gnu    strucchange 1.5-0  US            2
## # ... with 225,458 more rows
```

```r
#Descending order
arrange(cran2, desc(ip_id))
```

```
## # A tibble: 225,468 x 8
##       size r_version r_arch r_os         package      version country ip_id
##      <int> <chr>     <chr>  <chr>        <chr>        <chr>   <chr>   <int>
##  1    5933 <NA>      <NA>   <NA>         CPE          1.4.2   CN      13859
##  2  569241 3.1.0     x86_64 mingw32      multcompView 0.1-5   US      13858
##  3  228444 3.1.0     x86_64 mingw32      tourr        0.5.3   NZ      13857
##  4  308962 3.1.0     x86_64 darwin13.1.0 ctv          0.7-9   CN      13856
##  5  950964 3.0.3     i386   mingw32      knitr        1.6     CA      13855
##  6   80185 3.0.3     i386   mingw32      htmltools    0.2.4   CA      13855
##  7 1431750 3.0.3     i386   mingw32      shiny        0.10.0  CA      13855
##  8 2189695 3.1.0     x86_64 mingw32      RMySQL       0.9-3   US      13854
##  9 4818024 3.1.0     i386   mingw32      igraph       0.7.1   US      13853
## 10  197495 3.1.0     x86_64 mingw32      coda         0.16-1  US      13852
## # ... with 225,458 more rows
```

```r
#arrange by multiple variables
arrange(cran2, package, ip_id) #order of params determines order of arrange
```

```
## # A tibble: 225,468 x 8
##     size r_version r_arch r_os         package version country ip_id
##    <int> <chr>     <chr>  <chr>        <chr>   <chr>   <chr>   <int>
##  1 71677 3.0.3     x86_64 darwin10.8.0 A3      0.9.2   CN       1003
##  2 71672 3.1.0     x86_64 linux-gnu    A3      0.9.2   US       1015
##  3 71677 3.1.0     x86_64 mingw32      A3      0.9.2   IN       1054
##  4 70438 3.0.1     x86_64 darwin10.8.0 A3      0.9.2   CN       1513
##  5 71677 <NA>      <NA>   <NA>         A3      0.9.2   BR       1526
##  6 71892 3.0.2     x86_64 linux-gnu    A3      0.9.2   IN       1542
##  7 71677 3.1.0     x86_64 linux-gnu    A3      0.9.2   ZA       2925
##  8 71672 3.1.0     x86_64 mingw32      A3      0.9.2   IL       3889
##  9 71677 3.0.3     x86_64 mingw32      A3      0.9.2   DE       3917
## 10 71672 3.1.0     x86_64 mingw32      A3      0.9.2   US       4219
## # ... with 225,458 more rows
```

- `mutate()` - create a new variable based on the value of one or more variables in a dataset

```r
#Creating a new subset to demo mutate
cran3 <- select(cran, ip_id, package, size)
```

```r
mutate(cran3, size_mb = size / 2^20)
```

```
## # A tibble: 225,468 x 4
##    ip_id package          size size_mb
##    <int> <chr>           <int>   <dbl>
##  1     1 htmltools       80589 0.0769
##  2     2 tseries        321767 0.307
##  3     3 party          748063 0.713
##  4     3 Hmisc          606104 0.578
##  5     4 digest          79825 0.0761
##  6     3 randomForest    77681 0.0741
##  7     3 plyr           393754 0.376
##  8     5 whisker         28216 0.0269
##  9     6 Rcpp             5928 0.00565
## 10     7 hflights      2206029 2.10
## # ... with 225,458 more rows
```

```r
#mutate can also use a value that is created within it's own call
mutate(cran3, size_mb = size / 2^20, size_gb = size_mb / 2^10)
```

```
## # A tibble: 225,468 x 5
##    ip_id package          size size_mb    size_gb
##    <int> <chr>           <int>   <dbl>      <dbl>
##  1     1 htmltools       80589 0.0769  0.0000751
##  2     2 tseries        321767 0.307   0.000300
##  3     3 party          748063 0.713   0.000697
##  4     3 Hmisc          606104 0.578   0.000564
##  5     4 digest          79825 0.0761  0.0000743
##  6     3 randomForest    77681 0.0741  0.0000723
##  7     3 plyr           393754 0.376   0.000367
##  8     5 whisker         28216 0.0269  0.0000263
##  9     6 Rcpp             5928 0.00565 0.00000552
## 10     7 hflights      2206029 2.10    0.00205
## # ... with 225,458 more rows
```

- `summarize()` - collapses the dataset to a single row

```r
#Demo with average size:
summarize(cran, avg_bytes = mean(size))
```

```
## # A tibble: 1 x 1
##   avg_bytes
##       <dbl>
## 1   844086.
```

Grouping and Chaining with dplyr

- This assignment will also be using data about CRAN packages

```
saveLoc <- paste(getwd(), "/data/CRANpackages.csv", sep = "")
mydf <- read.csv(saveLoc, stringsAsFactors = FALSE)
cran <- tbl_df(mydf)
```

- The main idea behind grouping data is that you want to break up your dataset into groups of rows based on the values of one or more variables.
- The group_by() function is used for this

```
by_package <- group_by(cran, package)
by_package
```

```
## # A tibble: 225,468 x 12
## # Groups:   package [6,023]
##      X.1     X date  time    size r_version r_arch r_os  package version country
##    <int> <int> <chr> <chr>  <int> <chr>     <chr>  <chr> <chr>   <chr>   <chr>
##  1     1     1 2014~ 00:5~ 8.06e4 3.1.0     x86_64 ming~ htmlto~ 0.2.4   US
##  2     2     2 2014~ 00:5~ 3.22e5 3.1.0     x86_64 ming~ tseries 0.10-32 US
##  3     3     3 2014~ 00:4~ 7.48e5 3.1.0     x86_64 linu~ party   1.0-15  US
##  4     4     4 2014~ 00:4~ 6.06e5 3.1.0     x86_64 linu~ Hmisc   3.14-4  US
##  5     5     5 2014~ 00:4~ 7.98e4 3.0.2     x86_64 linu~ digest  0.6.4   CA
##  6     6     6 2014~ 00:4~ 7.77e4 3.1.0     x86_64 linu~ random~ 4.6-7   US
##  7     7     7 2014~ 00:4~ 3.94e5 3.1.0     x86_64 linu~ plyr    1.8.1   US
##  8     8     8 2014~ 00:4~ 2.82e4 3.0.2     x86_64 linu~ whisker 0.3-2   US
##  9     9     9 2014~ 00:5~ 5.93e3 <NA>      <NA>   <NA>  Rcpp    0.10.4  CN
## 10    10    10 2014~ 00:1~ 2.21e6 3.0.2     x86_64 linu~ hfligh~ 0.1     US
## # ... with 225,458 more rows, and 1 more variable: ip_id <int>
```

- Now summarizing the mean will be much more informative as it will be grouped by package

```
summarize(by_package, mean(size))
```

```
## # A tibble: 6,023 x 2
##    package     `mean(size)`
##    <chr>              <dbl>
##  1 A3                62195.
##  2 abc             4826665
##  3 abcdeFBA         455980.
##  4 ABCExtremes       22904.
##  5 ABCoptim          17807.
##  6 ABCp2             30473.
##  7 abctools        2589394
##  8 abd              453631.
##  9 abf2              35693.
## 10 abind             32939.
## # ... with 6,013 more rows
```

- We can make multiple parameters by group in the summarize function

```
pack_sum <- summarize(by_package,
                      count = n(), #Number of observations in the group
```

```r
                        unique = n_distinct(ip_id), #Faster equivalent of ‘length(unique(ip_id))
                        countries = n_distinct(country),
                        avg_bytes = mean(size))
pack_sum
```

```
## # A tibble: 6,023 x 5
##    package     count unique countries avg_bytes
##    <chr>       <int>  <int>     <int>     <dbl>
##  1 A3             25     24        10    62195.
##  2 abc            29     25        16   4826665
##  3 abcdeFBA       15     15         9   455980.
##  4 ABCExtremes    18     17         9    22904.
##  5 ABCoptim       16     15         9    17807.
##  6 ABCp2          18     17        10    30473.
##  7 abctools       19     19        11   2589394
##  8 abd            17     16        10   453631.
##  9 abf2           13     13         9    35693.
## 10 abind         396    365        50    32939.
## # ... with 6,013 more rows
```

```r
#Find 99th percentile
quantile(pack_sum$count, probs = 0.99)
```

```
##    99%
## 679.56
```

```r
#filter the top 1% of packages
top_counts <- filter(pack_sum, count > 679)
top_counts #Only shows 10
```

```
## # A tibble: 61 x 5
##    package    count unique countries avg_bytes
##    <chr>      <int>  <int>     <int>     <dbl>
##  1 bitops      1549   1408        76    28715.
##  2 car         1008    837        64  1229122.
##  3 caTools      812    699        64   176589.
##  4 colorspace  1683   1433        80   357411.
##  5 data.table   680    564        59  1252721.
##  6 DBI         2599    492        48   206933.
##  7 devtools     769    560        55   212933.
##  8 dichromat   1486   1257        74   134732.
##  9 digest      2210   1894        83   120549.
## 10 doSNOW       740     75        24     8364.
## # ... with 51 more rows
```

- The `View()` function allows us to see all the rows of a `tbl_df` (Should be executed in `RStudio`)

```r
#Let’s be real, we want that data to be informative to look at
top_counts_sorted <- arrange(top_counts, desc(count))
#Commented out for knitted document# View(top_counts_sorted)
```

- *'piping'* (or *'chaining'*)

```r
#WITHOUT pipes:
result2 <-
  arrange(
    filter(
      summarize(
        group_by(cran,
                 package
        ),
        count = n(),
        unique = n_distinct(ip_id),
        countries = n_distinct(country),
        avg_bytes = mean(size)
      ),
      countries > 60
    ),
    desc(countries),
    avg_bytes
  )

print(result2)
```

```
## # A tibble: 46 x 5
##    package      count unique countries avg_bytes
##    <chr>        <int>  <int>     <int>     <dbl>
##  1 Rcpp          3195   2044        84  2512100.
##  2 digest        2210   1894        83   120549.
##  3 stringr       2267   1948        82    65277.
##  4 plyr          2908   1754        81   799123.
##  5 ggplot2       4602   1680        81  2427716.
##  6 colorspace    1683   1433        80   357411.
##  7 RColorBrewer  1890   1584        79    22764.
##  8 scales        1726   1408        77   126819.
##  9 bitops        1549   1408        76    28715.
## 10 reshape2      2032   1652        76   330128.
## # ... with 36 more rows
```

```r
#WITH pipes:
result3 <-
  cran %>%
  group_by(package) %>%
  summarize(count = n(),
            unique = n_distinct(ip_id),
            countries = n_distinct(country),
            avg_bytes = mean(size)
  ) %>%
  filter(countries > 60) %>%
  arrange(desc(countries), avg_bytes)
```

```r
# Print result to console
print(result3)
```

```
## # A tibble: 46 x 5
##    package      count unique countries avg_bytes
##    <chr>        <int>  <int>     <int>     <dbl>
##  1 Rcpp          3195   2044        84  2512100.
##  2 digest        2210   1894        83   120549.
##  3 stringr       2267   1948        82    65277.
##  4 plyr          2908   1754        81   799123.
##  5 ggplot2       4602   1680        81  2427716.
##  6 colorspace    1683   1433        80   357411.
##  7 RColorBrewer  1890   1584        79    22764.
##  8 scales        1726   1408        77   126819.
##  9 bitops        1549   1408        76    28715.
## 10 reshape2      2032   1652        76   330128.
## # ... with 36 more rows
```

Tidying Data with dplyr

- **Hadley Wickham's paper on tidy data** (You printed this out and read it already)
- The first part of this lesson talked about `gather`, however that has `lifecycle:retired` tagged in it.

```r
students <- data.frame (grade = as.factor(c("A", "B", "C", "D", "F")),
                        male = as.integer(c(5,4,8,4,5)),
                        female = as.integer(c(3,1,6,5,5)))
students
```

```
##   grade male female
## 1     A    5      3
## 2     B    4      1
## 3     C    8      6
## 4     D    4      5
## 5     F    5      5
```

```r
#gather(students, sex, count, -grade)
```

- The data argument, students, gives the name of the original dataset. The key and value arguments – sex and count, respectively – give the column names for our tidy dataset. The final argument, -grade, says that we want to gather all columns EXCEPT the grade column
- Ok, so.. all of these functions are being listed as retired. Just read Hadley's paper to understand what kinds of messy data we can get into. It'll be more informative.

Quiz Scribbles

1)

- *"The American Community Survey distributes downloadable data about United States communities. Download the 2006 microdata survey about housing for the state of Idaho."* **A code book describing the variable names**

```
url <- "https://d396qusza40orc.cloudfront.net/getdata%2Fdata%2Fss06hid.csv"
saveLoc <- paste(getwd(), "/data/IdahoHousing06.csv", sep = "")
download.file(url, saveLoc, "curl")
housing <- read.csv(saveLoc)
library(dplyr)
housing <- tbl_df(housing)
housing
```

```
## # A tibble: 6,496 x 188
##    RT    SERIALNO DIVISION  PUMA REGION    ST ADJUST  WGTP    NP  TYPE   ACR
##    <fct>    <int>    <int> <int>  <int> <int>  <int> <int> <int> <int> <int>
##  1 H          186        8   700      4    16 1.02e6    89     4     1     1
##  2 H          306        8   700      4    16 1.02e6   310     1     1    NA
##  3 H          395        8   100      4    16 1.02e6   106     2     1     1
##  4 H          506        8   700      4    16 1.02e6   240     4     1     1
##  5 H          835        8   800      4    16 1.02e6   118     4     1     2
##  6 H          989        8   700      4    16 1.02e6   115     4     1     1
##  7 H         1861        8   700      4    16 1.02e6     0     1     2    NA
##  8 H         2120        8   200      4    16 1.02e6    35     1     1     1
##  9 H         2278        8   400      4    16 1.02e6    47     2     1     1
## 10 H         2428        8   500      4    16 1.02e6    51     2     1     1
## # ... with 6,486 more rows, and 177 more variables: AGS <int>, BDS <int>,
## #   BLD <int>, BUS <int>, CONP <int>, ELEP <int>, FS <int>, FULP <int>,
## #   GASP <int>, HFL <int>, INSP <int>, KIT <int>, MHP <int>, MRGI <int>,
## #   MRGP <int>, MRGT <int>, MRGX <int>, PLM <int>, RMS <int>, RNTM <int>,
## #   RNTP <int>, SMP <int>, TEL <int>, TEN <int>, VACS <int>, VAL <int>,
## #   VEH <int>, WATP <int>, YBL <int>, FES <int>, FINCP <int>, FPARC <int>,
## #   GRNTP <int>, GRPIP <int>, HHL <int>, HHT <int>, HINCP <int>, HUGCL <int>,
## #   HUPAC <int>, HUPAOC <int>, HUPARC <int>, LNGI <int>, MV <int>, NOC <int>,
## #   NPF <int>, NPP <int>, NR <int>, NRC <int>, OCPIP <int>, PARTNER <int>,
## #   PSF <int>, R18 <int>, R60 <int>, R65 <int>, RESMODE <int>, SMOCP <int>,
## #   SMX <int>, SRNT <int>, SVAL <int>, TAXP <int>, WIF <int>, WKEXREL <int>,
## #   WORKSTAT <int>, FACRP <int>, FAGSP <int>, FBDSP <int>, FBLDP <int>,
## #   FBUSP <int>, FCONP <int>, FELEP <int>, FFSP <int>, FFULP <int>,
## #   FGASP <int>, FHFLP <int>, FINSP <int>, FKITP <int>, FMHP <int>,
## #   FMRGIP <int>, FMRGP <int>, FMRGTP <int>, FMRGXP <int>, FMVYP <int>,
## #   FPLMP <int>, FRMSP <int>, FRNTMP <int>, FRNTP <int>, FSMP <int>,
## #   FSMXHP <int>, FSMXSP <int>, FTAXP <int>, FTELP <int>, FTENP <int>,
## #   FVACSP <int>, FVALP <int>, FVEHP <int>, FWATP <int>, FYBLP <int>,
## #   wgtp1 <int>, wgtp2 <int>, wgtp3 <int>, ...
```

- *"Create a logical vector that identifies the households on greater than 10 acres who sold more than $10,000 worth of agriculture products. Assign that logical vector to the variable agricultureLogical."*

Relavent info from the Code Book: * ACR: + N/A - (GQ/not a one-family house or mobile home) + 1 - House on less than one acre + 2 - House on one to less than ten acres + 3 - House on ten or more acresAGS 1

- AGS:
  - N/A - (less than 1 acre/GQ/vacant/2 or more units in structure)
  - 1 - None
  - 2 - $ 1 - $ 999
  - 3 - $ 1000 - $ 2499
  - 4 - $ 2500 - $ 4999
  - 5 - $ 5000 - $ 9999
  - 6 - $10000+

```
library(dplyr)
slimHousing <-  housing %>% select(SERIALNO, ACR, AGS) %>%
mutate(Qualify = ACR == 3 & AGS == 6)
```

"Apply the which() function like this to identify the rows of the data frame where the logical vector is TRUE. What are the first 3 values that result?"

```
which(slimHousing$Qualify)[1:3]
```

```
## [1] 125 238 262
```

2)

- *"Using the jpeg package read in the following picture of your instructor into R"*

```
url <- "https://d396qusza40orc.cloudfront.net/getdata%2Fjeff.jpg"
saveLoc <- paste(getwd(), "/data/jeff.jpg", sep = "")
download.file(url, saveLoc, "curl", mode = "wb") #jpeg is a binary file
```

- *"Use the parameter native=TRUE. What are the 30th and 80th quantiles of the resulting data?"*

```
library(jpeg)
jeff <- readJPEG(saveLoc, native = TRUE)

# "(some Linux systems may produce an answer 638 different for the 30th quantile)"
quantile(jeff, probs = c(.3, .8))
```

```
##       30%       80%
## -15258512 -10575416
```

3)

- *"Load the Gross Domestic Product data for the 190 ranked countries in this data set:"*

```
url <- "https://d396qusza40orc.cloudfront.net/getdata%2Fdata%2FGDP.csv"
saveLoc <- paste(getwd(), "/data/GDP.csv", sep = "")
```

```r
download.file(url, saveLoc, "curl")
GDP <- read.csv(saveLoc)

#Cleaning
GDP <- GDP[5:235,]
library(dplyr)
GDP <- GDP %>% rename(CountryCode = X, Ranking = Gross.domestic.product.2012,
                      Long.Name = X.2, mil.US.dollars = X.3) %>%
              select(CountryCode, Long.Name, Ranking, mil.US.dollars) %>%
              filter(!is.na(Ranking) & Ranking != "")
```

- *"Load the educational data from this data set"*

```r
url <- "https://d396qusza40orc.cloudfront.net/getdata%2Fdata%2FEDSTATS_Country.csv"
saveLoc <- paste(getwd(), "/data/Edu.csv", sep = "")
download.file(url, saveLoc, "curl")
edu <- read.csv(saveLoc)
```

- *"Match the data based on the country shortcode. How many of the IDs match?"*

```r
library(dplyr)
mergedData <- merge(GDP, edu, by = "CountryCode", all = FALSE)
mergedData
```

```
##     CountryCode                  Long.Name.x Ranking mil.US.dollars
## 1           ABW                        Aruba     161          2,584
## 2           AFG                  Afghanistan     105         20,497
## 3           AGO                       Angola      60        114,147
## 4           ALB                      Albania     125         12,648
## 5           ARE         United Arab Emirates      32        348,595
## 6           ARG                    Argentina      26        475,502
## 7           ARM                      Armenia     133          9,951
## 8           ATG          Antigua and Barbuda     172          1,134
## 9           AUS                    Australia      12      1,532,408
## 10          AUT                      Austria      27        394,708
## 11          AZE                   Azerbaijan      68         66,605
## 12          BDI                      Burundi     162          2,472
## 13          BEL                      Belgium      25        483,262
## 14          BEN                        Benin     140          7,557
## 15          BFA                 Burkina Faso     128         10,441
## 16          BGD                   Bangladesh      59        116,355
## 17          BGR                     Bulgaria      76         50,972
## 18          BHR                      Bahrain      93         29,044
## 19          BHS                 Bahamas, The     138          8,149
## 20          BIH       Bosnia and Herzegovina     111         17,466
## 21          BLR                      Belarus      69         63,267
## 22          BLZ                       Belize     169          1,493
## 23          BMU                      Bermuda     149          5,474
## 24          BOL                      Bolivia      96         27,035
```

```
## 25           BRA                      Brazil      7     2,252,664
## 26           BRB                    Barbados    153         4,225
## 27           BRN           Brunei Darussalam    113        16,954
## 28           BTN                      Bhutan    167         1,780
## 29           BWA                    Botswana    117        14,504
## 30           CAF    Central African Republic    165         2,184
## 31           CAN                      Canada     11     1,821,424
## 32           CHE                 Switzerland     20       631,173
## 33           CHL                       Chile     36       269,869
## 34           CHN                       China      2     8,227,103
## 35           CIV             C\xf4te d'Ivoire     99        24,680
## 36           CMR                    Cameroon     98        25,322
## 37           COG                 Congo, Rep.    121        13,678
## 38           COL                    Colombia     30       369,606
## 39           COM                     Comoros    182           596
## 40           CPV                  Cape Verde    166         1,827
## 41           CRI                  Costa Rica     81        45,104
## 42           CUB                        Cuba     67        68,234
## 43           CYP                      Cyprus    102        22,767
## 44           CZE              Czech Republic     51       196,446
## 45           DEU                     Germany      4     3,428,131
## 46           DMA                    Dominica    183           480
## 47           DNK                     Denmark     33       314,887
## 48           DOM          Dominican Republic     72        59,047
## 49           DZA                     Algeria     48       205,789
## 50           ECU                     Ecuador     64        84,040
## 51           EGY            Egypt, Arab Rep.     38       262,832
## 52           ERI                     Eritrea    159         3,092
## 53           ESP                       Spain     13     1,322,965
## 54           EST                     Estonia    103        22,390
## 55           ETH                    Ethiopia     85        41,605
## 56           FIN                     Finland     43       247,546
## 57           FJI                        Fiji    155         3,908
## 58           FRA                      France      5     2,612,878
## 59           FSM       Micronesia, Fed. Sts.    185           326
## 60           GAB                       Gabon    109        18,377
## 61           GBR              United Kingdom      6     2,471,784
## 62           GEO                     Georgia    114        15,747
## 63           GHA                       Ghana     86        40,711
## 64           GIN                      Guinea    148         5,632
## 65           GMB                 Gambia, The    175           917
## 66           GNB               Guinea-Bissau    176           822
## 67           GNQ           Equatorial Guinea    110        17,697
## 68           GRC                      Greece     42       249,099
## 69           GRD                     Grenada    178           767
## 70           GTM                   Guatemala     77        50,234
## 71           GUY                      Guyana    160         2,851
## 72           HKG        Hong Kong SAR, China     37       263,259
```

```
## 73          HND                    Honduras   108     18,434
## 74          HRV                     Croatia    71     59,228
## 75          HTI                       Haiti   139      7,843
## 76          HUN                     Hungary    58    124,600
## 77          IDN                   Indonesia    16    878,043
## 78          IND                       India    10  1,841,710
## 79          IRL                     Ireland    46    210,771
## 80          IRN          Iran, Islamic Rep.    22    514,060
## 81          IRQ                        Iraq    47    210,280
## 82          ISL                     Iceland   122     13,579
## 83          ISR                      Israel    40    258,217
## 84          ITA                       Italy     9  2,014,670
## 85          JAM                     Jamaica   116     14,755
## 86          JOR                      Jordan    92     31,015
## 87          JPN                       Japan     3  5,959,718
## 88          KAZ                  Kazakhstan    50    203,521
## 89          KEN                       Kenya    87     40,697
## 90          KGZ             Kyrgyz Republic   145      6,475
## 91          KHM                    Cambodia   120     14,038
## 92          KIR                    Kiribati   189        175
## 93          KNA         St. Kitts and Nevis   178        767
## 94          KOR                 Korea, Rep.    15  1,129,598
## 95          KSV                      Kosovo   146      6,445
## 96          KWT                      Kuwait    56    160,913
## 97          LAO                     Lao PDR   136      9,418
## 98          LBN                     Lebanon    83     42,945
## 99          LBR                     Liberia   168      1,734
## 100         LCA                   St. Lucia   171      1,239
## 101         LKA                   Sri Lanka    70     59,423
## 102         LSO                     Lesotho   163      2,448
## 103         LTU                   Lithuania    84     42,344
## 104         LUX                  Luxembourg    74     55,178
## 105         LVA                      Latvia    94     28,373
## 106         MAC           Macao SAR, China    82     43,582
## 107         MAR                     Morocco    62     95,982
## 108         MCO                      Monaco   147      6,075
## 109         MDA                     Moldova   141      7,253
## 110         MDG                  Madagascar   132      9,975
## 111         MDV                    Maldives   164      2,222
## 112         MEX                      Mexico    14  1,178,126
## 113         MHL            Marshall Islands   188        182
## 114         MKD              Macedonia, FYR   135      9,613
## 115         MLI                        Mali   129     10,308
## 116         MLT                       Malta   137      8,722
## 117         MNE                  Montenegro   151      4,373
## 118         MNG                    Mongolia   130     10,271
## 119         MOZ                  Mozambique   118     14,244
## 120         MRT                  Mauritania   154      4,199
```

```
## 121         MUS                    Mauritius  127     10,486
## 122         MWI                       Malawi  152      4,264
## 123         MYS                     Malaysia   34    305,033
## 124         NAM                      Namibia  123     13,072
## 125         NER                        Niger  144      6,773
## 126         NGA                      Nigeria   39    262,597
## 127         NIC                    Nicaragua  126     10,507
## 128         NLD                  Netherlands   18    770,555
## 129         NOR                       Norway   23    499,667
## 130         NPL                        Nepal  107     18,963
## 131         NZL                  New Zealand   55    167,347
## 132         OMN                         Oman   66     69,972
## 133         PAK                     Pakistan   44    225,143
## 134         PAN                       Panama   89     36,253
## 135         PER                         Peru   49    203,790
## 136         PHL                  Philippines   41    250,182
## 137         PLW                        Palau  187        228
## 138         PNG             Papua New Guinea  115     15,654
## 139         POL                       Poland   24    489,795
## 140         PRI                  Puerto Rico   61    101,496
## 141         PRT                     Portugal   45    212,274
## 142         PRY                     Paraguay   97     25,502
## 143         QAT                        Qatar   54    171,476
## 144         ROM                      Romania   52    192,711
## 145         RUS           Russian Federation    8  2,014,775
## 146         RWA                       Rwanda  142      7,103
## 147         SAU                 Saudi Arabia   19    711,050
## 148         SDN                        Sudan   73     58,769
## 149         SEN                      Senegal  119     14,046
## 150         SGP                    Singapore   35    274,701
## 151         SLB              Solomon Islands  174      1,008
## 152         SLE                 Sierra Leone  157      3,796
## 153         SLV                  El Salvador  100     23,864
## 154         SRB                       Serbia   88     37,489
## 155         STP S\xe3o Tom\xe9 and Principe  186        263
## 156         SUR                     Suriname  150      5,012
## 157         SVK              Slovak Republic   63     91,149
## 158         SVN                     Slovenia   80     45,279
## 159         SWE                       Sweden   21    523,806
## 160         SWZ                    Swaziland  158      3,744
## 161         SYC                   Seychelles  173      1,129
## 162         SYR         Syrian Arab Republic   65     73,672
## 163         TCD                         Chad  124     12,887
## 164         TGO                         Togo  156      3,814
## 165         THA                     Thailand   31    365,966
## 166         TJK                   Tajikistan  143      6,972
## 167         TKM                 Turkmenistan   91     35,164
## 168         TMP                  Timor-Leste  170      1,293
```

```
## 169          TON                        Tonga      184        472
## 170          TTO          Trinidad and Tobago      101     23,320
## 171          TUN                      Tunisia       79     45,662
## 172          TUR                       Turkey       17    789,257
## 173          TUV                       Tuvalu      190         40
## 174          TZA                     Tanzania       95     28,242
## 175          UGA                       Uganda      106     19,881
## 176          UKR                      Ukraine       53    176,309
## 177          URY                      Uruguay       78     49,920
## 178          USA                United States        1 16,244,600
## 179          UZB                   Uzbekistan       75     51,113
## 180          VCT St. Vincent and the Grenadines      180        713
## 181          VEN                Venezuela, RB       29    381,286
## 182          VNM                      Vietnam       57    155,820
## 183          VUT                      Vanuatu      177        787
## 184          WSM                        Samoa      181        684
## 185          YEM                  Yemen, Rep.       90     35,646
## 186          ZAF                 South Africa       28    384,313
## 187          ZAR             Congo, Dem. Rep.      112     17,204
## 188          ZMB                       Zambia      104     20,678
## 189          ZWE                     Zimbabwe      134      9,802
##                                                              Long.Name.y
## 1                                                                  Aruba
## 2                                          Islamic State of Afghanistan
## 3                                           People's Republic of Angola
## 4                                                   Republic of Albania
## 5                                                  United Arab Emirates
## 6                                                    Argentine Republic
## 7                                                   Republic of Armenia
## 8                                                   Antigua and Barbuda
## 9                                             Commonwealth of Australia
## 10                                                  Republic of Austria
## 11                                               Republic of Azerbaijan
## 12                                                  Republic of Burundi
## 13                                                    Kingdom of Belgium
## 14                                                    Republic of Benin
## 15                                                         Burkina Faso
## 16                                      People's Republic of Bangladesh
## 17                                                 Republic of Bulgaria
## 18                                                   Kingdom of Bahrain
## 19                                          Commonwealth of The Bahamas
## 20                                               Bosnia and Herzegovina
## 21                                                  Republic of Belarus
## 22                                                               Belize
## 23                                                         The Bermudas
## 24                                         Plurinational State of Bolivia
## 25                                        Federative Republic of Brazil
## 26                                                             Barbados
```

```
## 27                                                      Brunei Darussalam
## 28                                                      Kingdom of Bhutan
## 29                                                   Republic of Botswana
## 30                                               Central African Republic
## 31                                                                 Canada
## 32                                                            Switzerland
## 33                                                      Republic of Chile
## 34                                             People's Republic of China
## 35                                           Republic of C\xf4te d'Ivoire
## 36                                                   Republic of Cameroon
## 37                                                      Republic of Congo
## 38                                                   Republic of Colombia
## 39                                                   Union of the Comoros
## 40                                                 Republic of Cape Verde
## 41                                                 Republic of Costa Rica
## 42                                                       Republic of Cuba
## 43                                                     Republic of Cyprus
## 44                                                         Czech Republic
## 45                                            Federal Republic of Germany
## 46                                               Commonwealth of Dominica
## 47                                                     Kingdom of Denmark
## 48                                                     Dominican Republic
## 49                                People's Democratic Republic of Algeria
## 50                                                    Republic of Ecuador
## 51                                                 Arab Republic of Egypt
## 52                                                       State of Eritrea
## 53                                                       Kingdom of Spain
## 54                                                    Republic of Estonia
## 55                                Federal Democratic Republic of Ethiopia
## 56                                                    Republic of Finland
## 57                                                       Republic of Fiji
## 58                                                        French Republic
## 59                                         Federated States of Micronesia
## 60                                                      Gabonese Republic
## 61                   United Kingdom of Great Britain and Northern Ireland
## 62                                                                Georgia
## 63                                                      Republic of Ghana
## 64                                                     Republic of Guinea
## 65                                                 Republic of The Gambia
## 66                                              Republic of Guinea-Bissau
## 67                                         Republic of Equatorial Guinea
## 68                                                      Hellenic Republic
## 69                                                                Grenada
## 70                                                  Republic of Guatemala
## 71                                                     Republic of Guyana
## 72 Hong Kong Special Administrative Region of the People's Republic of China
## 73                                                   Republic of Honduras
## 74                                                    Republic of Croatia
```

```
## 75                                                              Republic of Haiti
## 76                                                            Republic of Hungary
## 77                                                          Republic of Indonesia
## 78                                                              Republic of India
## 79                                                                        Ireland
## 80                                                     Islamic Republic of Iran
## 81                                                               Republic of Iraq
## 82                                                            Republic of Iceland
## 83                                                                State of Israel
## 84                                                               Italian Republic
## 85                                                                        Jamaica
## 86                                                  Hashemite Kingdom of Jordan
## 87                                                                          Japan
## 88                                                         Republic of Kazakhstan
## 89                                                              Republic of Kenya
## 90                                                                Kyrgyz Republic
## 91                                                            Kingdom of Cambodia
## 92                                                           Republic of Kiribati
## 93                                                            St. Kitts and Nevis
## 94                                                              Republic of Korea
## 95                                                             Republic of Kosovo
## 96                                                                State of Kuwait
## 97                                             Lao People's Democratic Republic
## 98                                                              Lebanese Republic
## 99                                                            Republic of Liberia
## 100                                                                     St. Lucia
## 101                                 Democratic Socialist Republic of Sri Lanka
## 102                                                            Kingdom of Lesotho
## 103                                                        Republic of Lithuania
## 104                                                    Grand Duchy of Luxembourg
## 105                                                            Republic of Latvia
## 106     Macao Special Administrative Region of the People's Republic of China
## 107                                                            Kingdom of Morocco
## 108                                                       Principality of Monaco
## 109                                                           Republic of Moldova
## 110                                                       Republic of Madagascar
## 111                                                         Republic of Maldives
## 112                                                        United Mexican States
## 113                                            Republic of the Marshall Islands
## 114                                       Former Yugoslav Republic of Macedonia
## 115                                                              Republic of Mali
## 116                                                             Republic of Malta
## 117                                                                    Montenegro
## 118                                                                      Mongolia
## 119                                                       Republic of Mozambique
## 120                                                 Islamic Republic of Mauritania
## 121                                                        Republic of Mauritius
## 122                                                            Republic of Malawi
```

```
## 123 Malaysia
## 124 Republic of Namibia
## 125 Republic of Niger
## 126 Federal Republic of Nigeria
## 127 Republic of Nicaragua
## 128 Kingdom of the Netherlands
## 129 Kingdom of Norway
## 130 Nepal
## 131 New Zealand
## 132 Sultanate of Oman
## 133 Islamic Republic of Pakistan
## 134 Republic of Panama
## 135 Republic of Peru
## 136 Republic of the Philippines
## 137 Republic of Palau
## 138 The Independent State of Papua New Guinea
## 139 Republic of Poland
## 140 Puerto Rico
## 141 Portuguese Republic
## 142 Republic of Paraguay
## 143 State of Qatar
## 144 Romania
## 145 Russian Federation
## 146 Republic of Rwanda
## 147 Kingdom of Saudi Arabia
## 148 Republic of the Sudan
## 149 Republic of Senegal
## 150 Republic of Singapore
## 151 Solomon Islands
## 152 Republic of Sierra Leone
## 153 Republic of El Salvador
## 154 Republic of Serbia
## 155 Democratic Republic of S\xe3o Tom\xe9 and Principe
## 156 Republic of Suriname
## 157 Slovak Republic
## 158 Republic of Slovenia
## 159 Kingdom of Sweden
## 160 Kingdom of Swaziland
## 161 Republic of Seychelles
## 162 Syrian Arab Republic
## 163 Republic of Chad
## 164 Republic of Togo
## 165 Kingdom of Thailand
## 166 Republic of Tajikistan
## 167 Turkmenistan
## 168 Democratic Republic of Timor-Leste
## 169 Kingdom of Tonga
## 170 Republic of Trinidad and Tobago
```

```
## 171                                                                   Republic of Tunisia
## 172                                                                    Republic of Turkey
## 173                                                                                Tuvalu
## 174                                                           United Republic of Tanzania
## 175                                                                    Republic of Uganda
## 176                                                                               Ukraine
## 177                                                          Oriental Republic of Uruguay
## 178                                                              United States of America
## 179                                                                Republic of Uzbekistan
## 180                                                        St. Vincent and the Grenadines
## 181                                               Rep\xfablica Bolivariana de Venezuela
## 182                                                       Socialist Republic of Vietnam
## 183                                                                   Republic of Vanuatu
## 184                                                                                 Samoa
## 185                                                                     Republic of Yemen
## 186                                                              Republic of South Africa
## 187                                                    Democratic Republic of the Congo
## 188                                                                    Republic of Zambia
## 189                                                                  Republic of Zimbabwe
##              Income.Group                     Region Lending.category
## 1   High income: nonOECD  Latin America & Caribbean
## 2             Low income                 South Asia              IDA
## 3    Lower middle income         Sub-Saharan Africa              IDA
## 4    Upper middle income      Europe & Central Asia             IBRD
## 5   High income: nonOECD Middle East & North Africa
## 6    Upper middle income  Latin America & Caribbean             IBRD
## 7    Lower middle income      Europe & Central Asia            Blend
## 8    Upper middle income  Latin America & Caribbean             IBRD
## 9      High income: OECD        East Asia & Pacific
## 10     High income: OECD      Europe & Central Asia
## 11   Upper middle income      Europe & Central Asia            Blend
## 12            Low income         Sub-Saharan Africa              IDA
## 13     High income: OECD      Europe & Central Asia
## 14            Low income         Sub-Saharan Africa              IDA
## 15            Low income         Sub-Saharan Africa              IDA
## 16            Low income                 South Asia              IDA
## 17   Upper middle income      Europe & Central Asia             IBRD
## 18  High income: nonOECD Middle East & North Africa
## 19  High income: nonOECD  Latin America & Caribbean
## 20   Upper middle income      Europe & Central Asia            Blend
## 21   Upper middle income      Europe & Central Asia             IBRD
## 22   Lower middle income  Latin America & Caribbean             IBRD
## 23  High income: nonOECD              North America
## 24   Lower middle income  Latin America & Caribbean            Blend
## 25   Upper middle income  Latin America & Caribbean             IBRD
## 26  High income: nonOECD  Latin America & Caribbean
## 27  High income: nonOECD        East Asia & Pacific
## 28   Lower middle income                 South Asia              IDA
```

```
## 29   Upper middle income         Sub-Saharan Africa             IBRD
## 30            Low income         Sub-Saharan Africa              IDA
## 31     High income: OECD              North America
## 32     High income: OECD      Europe & Central Asia
## 33   Upper middle income  Latin America & Caribbean             IBRD
## 34   Lower middle income        East Asia & Pacific             IBRD
## 35   Lower middle income         Sub-Saharan Africa              IDA
## 36   Lower middle income         Sub-Saharan Africa              IDA
## 37   Lower middle income         Sub-Saharan Africa              IDA
## 38   Upper middle income  Latin America & Caribbean             IBRD
## 39            Low income         Sub-Saharan Africa              IDA
## 40   Lower middle income         Sub-Saharan Africa            Blend
## 41   Upper middle income  Latin America & Caribbean             IBRD
## 42   Upper middle income  Latin America & Caribbean
## 43  High income: nonOECD      Europe & Central Asia
## 44     High income: OECD      Europe & Central Asia
## 45     High income: OECD      Europe & Central Asia
## 46   Upper middle income  Latin America & Caribbean            Blend
## 47     High income: OECD      Europe & Central Asia
## 48   Upper middle income  Latin America & Caribbean             IBRD
## 49   Upper middle income Middle East & North Africa             IBRD
## 50   Lower middle income  Latin America & Caribbean             IBRD
## 51   Lower middle income Middle East & North Africa             IBRD
## 52            Low income         Sub-Saharan Africa              IDA
## 53     High income: OECD      Europe & Central Asia
## 54  High income: nonOECD      Europe & Central Asia
## 55            Low income         Sub-Saharan Africa              IDA
## 56     High income: OECD      Europe & Central Asia
## 57   Upper middle income        East Asia & Pacific             IBRD
## 58     High income: OECD      Europe & Central Asia
## 59   Lower middle income        East Asia & Pacific             IBRD
## 60   Upper middle income         Sub-Saharan Africa             IBRD
## 61     High income: OECD      Europe & Central Asia
## 62   Lower middle income      Europe & Central Asia            Blend
## 63            Low income         Sub-Saharan Africa              IDA
## 64            Low income         Sub-Saharan Africa              IDA
## 65            Low income         Sub-Saharan Africa              IDA
## 66            Low income         Sub-Saharan Africa              IDA
## 67  High income: nonOECD         Sub-Saharan Africa             IBRD
## 68     High income: OECD      Europe & Central Asia
## 69   Upper middle income  Latin America & Caribbean            Blend
## 70   Lower middle income  Latin America & Caribbean             IBRD
## 71   Lower middle income  Latin America & Caribbean              IDA
## 72  High income: nonOECD        East Asia & Pacific
## 73   Lower middle income  Latin America & Caribbean              IDA
## 74  High income: nonOECD      Europe & Central Asia             IBRD
## 75            Low income  Latin America & Caribbean              IDA
## 76     High income: OECD      Europe & Central Asia
```

```
## 77   Lower middle income       East Asia & Pacific             IBRD
## 78   Lower middle income                South Asia            Blend
## 79     High income: OECD     Europe & Central Asia
## 80   Upper middle income Middle East & North Africa            IBRD
## 81   Lower middle income Middle East & North Africa            IBRD
## 82     High income: OECD     Europe & Central Asia
## 83     High income: OECD Middle East & North Africa
## 84     High income: OECD     Europe & Central Asia
## 85   Upper middle income  Latin America & Caribbean            IBRD
## 86   Lower middle income Middle East & North Africa            IBRD
## 87     High income: OECD       East Asia & Pacific
## 88   Upper middle income     Europe & Central Asia             IBRD
## 89            Low income        Sub-Saharan Africa              IDA
## 90            Low income     Europe & Central Asia              IDA
## 91            Low income       East Asia & Pacific              IDA
## 92   Lower middle income       East Asia & Pacific              IDA
## 93   Upper middle income  Latin America & Caribbean            IBRD
## 94     High income: OECD       East Asia & Pacific             IBRD
## 95   Lower middle income     Europe & Central Asia              IDA
## 96  High income: nonOECD Middle East & North Africa
## 97            Low income       East Asia & Pacific              IDA
## 98   Upper middle income Middle East & North Africa            IBRD
## 99            Low income        Sub-Saharan Africa              IDA
## 100  Upper middle income  Latin America & Caribbean           Blend
## 101  Lower middle income                South Asia              IDA
## 102  Lower middle income        Sub-Saharan Africa              IDA
## 103  Upper middle income     Europe & Central Asia
## 104    High income: OECD     Europe & Central Asia
## 105 High income: nonOECD     Europe & Central Asia
## 106 High income: nonOECD       East Asia & Pacific
## 107  Lower middle income Middle East & North Africa            IBRD
## 108 High income: nonOECD     Europe & Central Asia
## 109  Lower middle income     Europe & Central Asia              IDA
## 110           Low income        Sub-Saharan Africa              IDA
## 111  Lower middle income                South Asia              IDA
## 112  Upper middle income  Latin America & Caribbean            IBRD
## 113  Lower middle income       East Asia & Pacific             IBRD
## 114  Upper middle income     Europe & Central Asia             IBRD
## 115           Low income        Sub-Saharan Africa              IDA
## 116 High income: nonOECD Middle East & North Africa
## 117  Upper middle income     Europe & Central Asia             IBRD
## 118  Lower middle income       East Asia & Pacific              IDA
## 119           Low income        Sub-Saharan Africa              IDA
## 120           Low income        Sub-Saharan Africa              IDA
## 121  Upper middle income        Sub-Saharan Africa             IBRD
## 122           Low income        Sub-Saharan Africa              IDA
## 123  Upper middle income       East Asia & Pacific             IBRD
## 124  Upper middle income        Sub-Saharan Africa             IBRD
```

```
## 125           Low income         Sub-Saharan Africa                IDA
## 126  Lower middle income         Sub-Saharan Africa                IDA
## 127  Lower middle income  Latin America & Caribbean                IDA
## 128    High income: OECD      Europe & Central Asia
## 129    High income: OECD      Europe & Central Asia
## 130           Low income                 South Asia                IDA
## 131    High income: OECD        East Asia & Pacific
## 132 High income: nonOECD Middle East & North Africa
## 133  Lower middle income                 South Asia              Blend
## 134  Upper middle income  Latin America & Caribbean               IBRD
## 135  Upper middle income  Latin America & Caribbean               IBRD
## 136  Lower middle income        East Asia & Pacific               IBRD
## 137  Upper middle income        East Asia & Pacific               IBRD
## 138  Lower middle income        East Asia & Pacific              Blend
## 139    High income: OECD      Europe & Central Asia               IBRD
## 140 High income: nonOECD  Latin America & Caribbean
## 141    High income: OECD      Europe & Central Asia
## 142  Lower middle income  Latin America & Caribbean               IBRD
## 143 High income: nonOECD Middle East & North Africa
## 144  Upper middle income      Europe & Central Asia               IBRD
## 145  Upper middle income      Europe & Central Asia               IBRD
## 146           Low income         Sub-Saharan Africa                IDA
## 147 High income: nonOECD Middle East & North Africa
## 148  Lower middle income         Sub-Saharan Africa                IDA
## 149  Lower middle income         Sub-Saharan Africa                IDA
## 150 High income: nonOECD        East Asia & Pacific
## 151           Low income        East Asia & Pacific                IDA
## 152           Low income         Sub-Saharan Africa                IDA
## 153  Lower middle income  Latin America & Caribbean               IBRD
## 154  Upper middle income      Europe & Central Asia               IBRD
## 155  Lower middle income         Sub-Saharan Africa                IDA
## 156  Upper middle income  Latin America & Caribbean               IBRD
## 157    High income: OECD      Europe & Central Asia
## 158    High income: OECD      Europe & Central Asia
## 159    High income: OECD      Europe & Central Asia
## 160  Lower middle income         Sub-Saharan Africa               IBRD
## 161  Upper middle income         Sub-Saharan Africa               IBRD
## 162  Lower middle income Middle East & North Africa               IBRD
## 163           Low income         Sub-Saharan Africa                IDA
## 164           Low income         Sub-Saharan Africa                IDA
## 165  Lower middle income        East Asia & Pacific               IBRD
## 166           Low income      Europe & Central Asia                IDA
## 167  Lower middle income      Europe & Central Asia               IBRD
## 168  Lower middle income        East Asia & Pacific                IDA
## 169  Lower middle income        East Asia & Pacific                IDA
## 170 High income: nonOECD  Latin America & Caribbean               IBRD
## 171  Lower middle income Middle East & North Africa               IBRD
## 172  Upper middle income      Europe & Central Asia               IBRD
```

```
## 173  Lower middle income        East Asia & Pacific
## 174           Low income         Sub-Saharan Africa              IDA
## 175           Low income         Sub-Saharan Africa              IDA
## 176  Lower middle income      Europe & Central Asia             IBRD
## 177  Upper middle income  Latin America & Caribbean             IBRD
## 178    High income: OECD              North America
## 179  Lower middle income      Europe & Central Asia            Blend
## 180  Upper middle income  Latin America & Caribbean            Blend
## 181  Upper middle income  Latin America & Caribbean             IBRD
## 182  Lower middle income        East Asia & Pacific            Blend
## 183  Lower middle income        East Asia & Pacific              IDA
## 184  Lower middle income        East Asia & Pacific              IDA
## 185  Lower middle income Middle East & North Africa              IDA
## 186  Upper middle income         Sub-Saharan Africa             IBRD
## 187           Low income         Sub-Saharan Africa              IDA
## 188           Low income         Sub-Saharan Africa              IDA
## 189           Low income         Sub-Saharan Africa            Blend
##     Other.groups                           Currency.Unit
## 1                                          Aruban florin
## 2           HIPC                          Afghan afghani
## 3                                         Angolan kwanza
## 4                                           Albanian lek
## 5                                          U.A.E. dirham
## 6                                         Argentine peso
## 7                                          Armenian dram
## 8                                 East Caribbean dollar
## 9                                     Australian dollar
## 10     Euro area                                    Euro
## 11                                       New Azeri manat
## 12          HIPC                           Burundi franc
## 13     Euro area                                    Euro
## 14          HIPC                               CFA franc
## 15          HIPC                               CFA franc
## 16                                      Bangladeshi taka
## 17                                         Bulgarian lev
## 18                                        Bahraini dinar
## 19                                       Bahamian dollar
## 20              Bosnia and Herzegovina convertible mark
## 21                                      Belarusian rubel
## 22                                         Belize dollar
## 23                                        Bermuda dollar
## 24          HIPC                     Bolivian Boliviano
## 25                                        Brazilian real
## 26                                       Barbados dollar
## 27                                         Brunei dollar
## 28                                   Bhutanese ngultrum
## 29                                         Botswana pula
## 30          HIPC                               CFA franc
```

```
## 31                                    Canadian dollar
## 32                                        Swiss franc
## 33                                       Chilean peso
## 34                                       Chinese yuan
## 35         HIPC                             CFA franc
## 36         HIPC                             CFA franc
## 37         HIPC                             CFA franc
## 38                                     Colombian peso
## 39         HIPC                        Comorian franc
## 40                                  Cape Verde escudo
## 41                                  Costa Rican colon
## 42                                         Cuban peso
## 43    Euro area                                  Euro
## 44                                       Czech koruna
## 45    Euro area                                  Euro
## 46                              East Caribbean dollar
## 47                                       Danish krone
## 48                                     Dominican peso
## 49                                     Algerian dinar
## 50                                        U.S. dollar
## 51                                     Egyptian pound
## 52         HIPC                        Eritrean nakfa
## 53    Euro area                                  Euro
## 54                                     Estonian kroon
## 55         HIPC                        Ethiopian birr
## 56    Euro area                                  Euro
## 57                                      Fijian dollar
## 58    Euro area                                  Euro
## 59                                        U.S. dollar
## 60                                          CFA franc
## 61                                     Pound sterling
## 62                                      Georgian lari
## 63         HIPC                     New Ghanaian cedi
## 64         HIPC                         Guinean franc
## 65         HIPC                        Gambian dalasi
## 66         HIPC                             CFA franc
## 67                                          CFA franc
## 68    Euro area                                  Euro
## 69                              East Caribbean dollar
## 70                                  Guatemalan quetzal
## 71         HIPC                         Guyana dollar
## 72                                   Hong Kong dollar
## 73         HIPC                       Honduran lempira
## 74                                      Croatian kuna
## 75         HIPC                        Haitian gourde
## 76                                   Hungarian forint
## 77                                  Indonesian rupiah
## 78                                       Indian rupee
```

```
## 79     Euro area                                    Euro
## 80                                          Iranian rial
## 81                                           Iraqi dinar
## 82                                          Iceland krona
## 83                                    Israeli new shekel
## 84     Euro area                                    Euro
## 85                                        Jamaican dollar
## 86                                        Jordanian dinar
## 87                                          Japanese yen
## 88                                          Kazakh tenge
## 89                                       Kenyan shilling
## 90          HIPC                              Kyrgyz som
## 91                                        Cambodian riel
## 92                                     Australian dollar
## 93                                  East Caribbean dollar
## 94                                            Korean won
## 95                                                  Euro
## 96                                          Kuwaiti dinar
## 97                                               Lao kip
## 98                                        Lebanese pound
## 99          HIPC                         Liberian dollar
## 100                                 East Caribbean dollar
## 101                                      Sri Lankan rupee
## 102                                          Lesotho loti
## 103                                     Lithuanian litas
## 104    Euro area                                    Euro
## 105                                         Latvian lats
## 106                                         Macao pataca
## 107                                       Moroccan dirham
## 108                                                 Euro
## 109                                         Moldovan leu
## 110         HIPC                         Malagasy ariary
## 111                                     Maldivian rufiyaa
## 112                                         Mexican peso
## 113                                          U.S. dollar
## 114                                      Macedonian denar
## 115         HIPC                               CFA franc
## 116    Euro area    Euro (data reported in Maltese liri)
## 117                                                 Euro
## 118                                      Mongolian tugrik
## 119         HIPC                  New Mozambican metical
## 120         HIPC                    Mauritanian ouguiya
## 121                                       Mauritian rupee
## 122         HIPC                            Malawi kwacha
## 123                                     Malaysian ringgit
## 124                                       Namibian dollar
## 125         HIPC                               CFA franc
## 126                                       Nigerian naira
```

```
## 127        HIPC                Nicaraguan gold cordoba
## 128   Euro area                                   Euro
## 129                                  Norwegian krone
## 130                                   Nepalese rupee
## 131                               New Zealand dollar
## 132                                       Rial Omani
## 133                                  Pakistani rupee
## 134                                Panamanian balboa
## 135                                 Peruvian new sol
## 136                                  Philippine peso
## 137                                      U.S. dollar
## 138                            Papua New Guinea kina
## 139                                     Polish zloty
## 140                                      U.S. dollar
## 141   Euro area                                   Euro
## 142                               Paraguayan guarani
## 143                                     Qatari riyal
## 144                                 New Romanian leu
## 145                                    Russian ruble
## 146        HIPC                          Rwandan franc
## 147                              Saudi Arabian riyal
## 148        HIPC                         Sudanese pound
## 149        HIPC                              CFA franc
## 150                                 Singapore dollar
## 151                           Solomon Islands dollar
## 152        HIPC                    Sierra Leonean leone
## 153                                      U.S. dollar
## 154                                    Serbian dinar
## 155        HIPC       S\xe3o Tom\xe9 and Principe dobra
## 156                                  Suriname dollar
## 157   Euro area                                   Euro
## 158   Euro area                                   Euro
## 159                                    Swedish krona
## 160                              Swaziland lilangeni
## 161                                 Seychelles rupee
## 162                                     Syrian pound
## 163        HIPC                              CFA franc
## 164        HIPC                              CFA franc
## 165                                        Thai baht
## 166                                     Tajik somoni
## 167                                New Turkmen manat
## 168                                      U.S. dollar
## 169                                  Tongan pa'anga
## 170                      Trinidad and Tobago dollar
## 171                                   Tunisian dinar
## 172                                 New Turkish lira
## 173                                Australian dollar
## 174        HIPC                     Tanzanian shilling
```

```
## 175         HIPC                          Ugandan shilling
## 176                                      Ukrainian hryvnia
## 177                                         Uruguayan peso
## 178                                            U.S. dollar
## 179                                              Uzbek sum
## 180                                   East Caribbean dollar
## 181                               Venezuelan bolivar fuerte
## 182                                        Vietnamese dong
## 183                                           Vanuatu vatu
## 184                                            Samoan tala
## 185                                            Yemeni rial
## 186                                      South African rand
## 187         HIPC                           Congolese franc
## 188         HIPC                            Zambian kwacha
## 189                                        Zimbabwe dollar
##     Latest.population.census    Latest.household.survey
## 1                       2000
## 2                       1979                 MICS, 2003
## 3                       1970   MICS, 2001, MIS, 2006/07
## 4                       2001                 MICS, 2005
## 5                       2005
## 6                       2001
## 7                       2001                  DHS, 2005
## 8                       2001
## 9                       2006
## 10                      2001
## 11                      2009                  DHS, 2006
## 12                      1990                 MICS, 2005
## 13                      2001
## 14                      2002                  DHS, 2006
## 15                      2006                 MICS, 2006
## 16                      2001                  DHS, 2007
## 17                      2001
## 18                      2001
## 19                      2000
## 20                      1991                 MICS, 2006
## 21                      1999                 MICS, 2005
## 22                      2000                 MICS, 2006
## 23                      2000
## 24                      2001                  DHS, 2008
## 25                      2000                  DHS, 1996
## 26                      2000
## 27                      2001
## 28                      2005
## 29                      2001                 MICS, 2000
## 30                      2003                 MICS, 2006
## 31                      2006
## 32                      2000
```

```
## 33                       2002
## 34                       2000                      NSS, 2007
## 35                       1998                     MICS, 2006
## 36                       1987                     MICS, 2006
## 37                       1996                      DHS, 2005
## 38                       2005                      DHS, 2005
## 39                       2003                     MICS, 2000
## 40                       2000
## 41                       2000                      RHS, 1993
## 42                       2002                     MICS, 2006
## 43                       2001
## 44                       2001                      RHS, 1993
## 45                       2001
## 46                       2001
## 47                       2001
## 48                       2002                      DHS, 2007
## 49                       2008                     MICS, 2006
## 50                       2001                      RHS, 2004
## 51                       2006                      DHS, 2008
## 52                       1984                      DHS, 2002
## 53                       2001
## 54                       2000
## 55                       2007                      DHS, 2005
## 56                       2000
## 57                       2007
## 58             2006 (rolling)
## 59                       2000
## 60                       2003                      DHS, 2000
## 61                       2001
## 62                       2002     MICS, 2005, RHS, 2005
## 63                       2000                      DHS, 2008
## 64                       1996                      DHS, 2005
## 65                       2003                  MICS, 2005/06
## 66                       2009                     MICS, 2006
## 67                       2002
## 68                       2001
## 69                       2001
## 70                       2002                      RHS, 2002
## 71                       2002                     MICS, 2006
## 72                       2006
## 73                       2001                   DHS, 2005/06
## 74                       2001
## 75                       2003                   DHS, 2005/06
## 76                       2001
## 77                       2000                      DHS, 2007
## 78                       2001                   DHS, 2005/06
## 79                       2006
## 80                       2006                      DHS, 2000
```

```
## 81                1997              MICS, 2006
## 82      Register based
## 83                2008
## 84                2001
## 85                2001               MICS 2005
## 86                2004               DHS, 2007
## 87                2005
## 88                1999              MICS, 2006
## 89                1999    DHS, 2003, SPA, 2004
## 90                2009            MICS 2005/06
## 91                2008               DHS, 2005
## 92                2005
## 93                2001
## 94                2005
## 95                1981
## 96                2005               FHS, 1996
## 97                2005              MICS, 2006
## 98                1970              MICS, 2000
## 99                2008 DHS, 2007, MIS, 2008/09
## 100               2001
## 101               2001               DHS, 1987
## 102               2006               DHS, 2004
## 103               2001
## 104               2001
## 105               2000
## 106               2006
## 107               2004              MICS, 2006
## 108               2008
## 109               2004               DHS, 2005
## 110               1993            DHS, 2003/04
## 111               2006              MICS, 2001
## 112               2005              ENPF, 1995
## 113               1999
## 114               2002              MICS, 2005
## 115               1998               DHS, 2006
## 116               2005
## 117               2003           MICS, 2005/06
## 118               2000              MICS, 2005
## 119               2007               DHS, 2003
## 120               2000              MICS, 2007
## 121               2000
## 122               2008               MICS 2006
## 123               2000
## 124               2001            DHS, 2006/07
## 125               2001          DHS/MICS, 2006
## 126               2006               DHS, 2008
## 127               2005            RHS, 2006/07
## 128               2001
```

```
## 129                2001
## 130                2001                    DHS, 2006
## 131                2006
## 132                2003                    FHS, 1995
## 133                1998                 DHS, 2006/07
## 134                2000                   LSMS, 2003
## 135                2007                 DHS, 2007/08
## 136                2007                    DHS, 2008
## 137                2005
## 138                2000                    DHS, 1996
## 139                2002
## 140                2000                 RHS, 1995/96
## 141                2001
## 142                2002                    RHS, 2004
## 143                2004
## 144                2002                    RHS, 1999
## 145                2002                    RHS, 1996
## 146                2002                 DHS, 2007/08
## 147                2004   Demographic survey, 2007
## 148                2008            MICS-PAPFAM 2006
## 149                2002    DHS, 2005, MIS, 2008-09
## 150                2000   General  household, 2005
## 151                1999
## 152                2004                     DHS 2008
## 153                2007                    RHS, 2008
## 154                2002                MICS, 2005-06
## 155                2001
## 156                2004                   MICS, 2000
## 157                2001
## 158                2002
## 159      Register based
## 160                2007                 DHS, 2006/07
## 161                2002
## 162                2004                   MICS, 2006
## 163                1993                    DHS, 2004
## 164                1981                   MICS, 2006
## 165                2000                 MICS 2005/06
## 166                2000                   MICS, 2005
## 167                1995                    MICS,2006
## 168                2004                   DGHS, 2003
## 169                2006
## 170                2000                   MICS, 2006
## 171                2004                   MICS, 2006
## 172                2000                    DHS, 2003
## 173
## 174                2002 DHS, 2004/05, AIS, 2007/08
## 175                2002       DHS, 2006, SPA, 2007
## 176                2001                    DHS, 2007
```

```
## 177                          2004
## 178                          2000                 CPS (monthly)
## 179                          1989                    MICS, 2006
## 180                          2001
## 181                          2001                    MICS, 2000
## 182                          2009                    MICS, 2006
## 183                          2009
## 184                          2006
## 185                          2004                    MICS, 2006
## 186                          2001                     DHS, 2003
## 187                          1984                      DHS 2007
## 188                          2000                     DHS, 2007
## 189                          2002                 DHS, 2005/06
##
## 1
## 2
## 3
## 4
## 5
## 6
## 7
## 8
## 9
## 10
## 11
## 12
## 13
## 14
## 15
## 16
## 17
## 18
## 19
## 20
## 21
## 22
## 23
## 24
## 25
## 26
## 27
## 28
## 29
## 30
## 31
## 32
## 33
## 34
```

```
## 35
## 36
## 37
## 38
## 39
## 40
## 41
## 42
## 43
## 44
## 45
## 46
## 47
## 48
## 49
## 50
## 51
## 52
## 53
## 54
## 55
## 56
## 57
## 58
## 59
## 60
## 61
## 62
## 63
## 64
## 65
## 66
## 67
## 68
## 69
## 70
## 71
## 72
## 73
## 74
## 75
## 76
## 77
## 78
## 79
## 80
## 81
## 82
```

```
## 83
## 84
## 85
## 86
## 87
## 88
## 89
## 90
## 91
## 92
## 93
## 94
## 95
## 96
## 97
## 98
## 99
## 100
## 101
## 102
## 103
## 104
## 105
## 106
## 107
## 108
## 109
## 110
## 111
## 112
## 113
## 114
## 115
## 116
## 117
## 118
## 119
## 120
## 121
## 122
## 123
## 124
## 125
## 126
## 127
## 128
## 129
## 130
```

```
## 131
## 132
## 133
## 134
## 135
## 136
## 137
## 138
## 139
## 140
## 141
## 142
## 143
## 144
## 145
## 146
## 147
## 148
## 149
## 150
## 151
## 152
## 153
## 154 Montenegro declared independence from Serbia and Montenegro on June 3, 2006. Where avail
## 155
## 156
## 157
## 158
## 159
## 160
## 161
## 162
## 163
## 164
## 165
## 166
## 167
## 168
## 169
## 170
## 171
## 172
## 173
## 174
## 175
## 176
## 177
## 178
```

```
## 179
## 180
## 181
## 182
## 183
## 184
## 185
## 186
## 187
## 188
## 189
##
## 1
## 2
## 3
## 4
## 5
## 6
## 7
## 8
## 9
## 10
## 11
## 12
## 13
## 14
## 15
## 16
## 17
## 18
## 19
## 20
## 21
## 22
## 23
## 24
## 25
## 26
## 27
## 28
## 29
## 30
## 31
## 32
## 33
## 34
## 35
## 36
```

```
## 37
## 38
## 39
## 40
## 41
## 42
## 43
## 44
## 45
## 46
## 47
## 48
## 49
## 50
## 51
## 52
## 53
## 54
## 55
## 56
## 57
## 58
## 59
## 60
## 61
## 62
## 63
## 64
## 65
## 66
## 67
## 68
## 69
## 70
## 71
## 72
## 73
## 74
## 75
## 76
## 77
## 78
## 79
## 80
## 81
## 82
## 83
## 84
```

```
## 85
## 86
## 87
## 88
## 89
## 90
## 91
## 92
## 93
## 94
## 95
## 96
## 97
## 98
## 99
## 100
## 101
## 102
## 103
## 104
## 105
## 106
## 107
## 108
## 109
## 110
## 111
## 112
## 113
## 114
## 115
## 116
## 117
## 118
## 119
## 120
## 121
## 122
## 123
## 124
## 125
## 126
## 127
## 128
## 129
## 130
## 131
## 132
```

```
## 133
## 134
## 135
## 136
## 137
## 138
## 139
## 140
## 141
## 142
## 143
## 144
## 145
## 146
## 147
## 148 1981/82 (Reporting period switch from fiscal year to calendar year from 1996. Pre-1996
## 149
## 150
## 151
## 152
## 153
## 154
## 155
## 156
## 157
## 158
## 159
## 160
## 161
## 162
## 163
## 164
## 165
## 166
## 167
## 168
## 169
## 170
## 171
## 172
## 173
## 174
## 175
## 176
## 177
## 178
## 179
## 180
```

```
## 181
## 182
## 183
## 184
## 185
## 186
## 187
## 188
## 189
##     National.accounts.reference.year System.of.National.Accounts
## 1                                  NA                          NA
## 2                                  NA                          NA
## 3                                  NA                          NA
## 4                                1996                        1993
## 5                                  NA                          NA
## 6                                  NA                        1993
## 7                                1996                        1993
## 8                                  NA                          NA
## 9                                2007                        1993
## 10                                 NA                        1993
## 11                               2003                        1993
## 12                                 NA                          NA
## 13                                 NA                        1993
## 14                                 NA                          NA
## 15                                 NA                          NA
## 16                                 NA                        1993
## 17                               2002                        1993
## 18                                 NA                          NA
## 19                                 NA                        1993
## 20                               1996                        1993
## 21                               2000                        1993
## 22                                 NA                        1993
## 23                                 NA                          NA
## 24                                 NA                        1993
## 25                                 NA                        1993
## 26                                 NA                          NA
## 27                                 NA                          NA
## 28                                 NA                        1993
## 29                                 NA                        1993
## 30                                 NA                          NA
## 31                                 NA                        1993
## 32                                 NA                          NA
## 33                                 NA                        1993
## 34                                 NA                        1993
## 35                                 NA                          NA
## 36                                 NA                        1993
## 37                                 NA                          NA
## 38                                 NA                        1993
```

```
## 39                                            NA                         NA
## 40                                            NA                         NA
## 41                                            NA                       1993
## 42                                            NA                         NA
## 43                                          2000                         NA
## 44                                          1995                       1993
## 45                                            NA                       1993
## 46                                            NA                       1993
## 47                                            NA                       1993
## 48                                            NA                         NA
## 49                                            NA                         NA
## 50                                            NA                       1993
## 51                                            NA                         NA
## 52                                            NA                         NA
## 53                                            NA                       1993
## 54                                            NA                       1993
## 55                                            NA                       1993
## 56                                            NA                       1993
## 57                                            NA                         NA
## 58                                          2000                       1993
## 59                                            NA                         NA
## 60                                            NA                         NA
## 61                                            NA                       1993
## 62                                          1996                       1993
## 63                                            NA                         NA
## 64                                            NA                         NA
## 65                                            NA                         NA
## 66                                            NA                         NA
## 67                                            NA                         NA
## 68                                          2000                         NA
## 69                                            NA                         NA
## 70                                            NA                       1993
## 71                                            NA                         NA
## 72                                            NA                       1993
## 73                                            NA                       1993
## 74                                          1997                       1993
## 75                                            NA                         NA
## 76                                          2000                       1993
## 77                                            NA                         NA
## 78                                            NA                       1993
## 79                                            NA                       1993
## 80                                            NA                         NA
## 81                                            NA                         NA
## 82                                            NA                         NA
## 83                                            NA                       1993
## 84                                            NA                       1993
## 85                                            NA                         NA
## 86                                            NA                         NA
```

```
## 87                                NA                          NA
## 88                              1995                        1993
## 89                                NA                        1993
## 90                              1995                        1993
## 91                                NA                          NA
## 92                                NA                          NA
## 93                                NA                        1993
## 94                                NA                        1993
## 95                                NA                          NA
## 96                                NA                          NA
## 97                                NA                          NA
## 98                                NA                          NA
## 99                                NA                          NA
## 100                               NA                          NA
## 101                               NA                          NA
## 102                               NA                        1993
## 103                               NA                        1993
## 104                             2000                          NA
## 105                               NA                        1993
## 106                               NA                          NA
## 107                               NA                          NA
## 108                               NA                          NA
## 109                             1996                        1993
## 110                               NA                          NA
## 111                               NA                          NA
## 112                               NA                        1993
## 113                               NA                          NA
## 114                             1995                        1993
## 115                               NA                          NA
## 116                               NA                          NA
## 117                               NA                        1993
## 118                               NA                        1993
## 119                               NA                          NA
## 120                               NA                          NA
## 121                               NA                          NA
## 122                               NA                          NA
## 123                               NA                          NA
## 124                               NA                        1993
## 125                               NA                          NA
## 126                               NA                          NA
## 127                               NA                        1993
## 128                             2000                        1993
## 129                             2000                        1993
## 130                               NA                          NA
## 131                               NA                          NA
## 132                               NA                          NA
## 133                               NA                        1993
## 134                               NA                        1993
```

```
## 135                                     NA                           NA
## 136                                     NA                           NA
## 137                                     NA                           NA
## 138                                     NA                           NA
## 139                                   2002                         1993
## 140                                     NA                           NA
## 141                                     NA                         1993
## 142                                     NA                           NA
## 143                                     NA                           NA
## 144                                   2005                         1993
## 145                                     NA                         1993
## 146                                     NA                           NA
## 147                                     NA                           NA
## 148                                   1996                           NA
## 149                                   1987                         1993
## 150                                     NA                         1993
## 151                                     NA                           NA
## 152                                     NA                         1993
## 153                                     NA                           NA
## 154                                   2002                         1993
## 155                                     NA                           NA
## 156                                     NA                         1993
## 157                                   1995                         1993
## 158                                   2000                         1993
## 159                                   2000                           NA
## 160                                     NA                           NA
## 161                                     NA                           NA
## 162                                     NA                           NA
## 163                                     NA                         1993
## 164                                     NA                           NA
## 165                                     NA                           NA
## 166                                   2000                         1993
## 167                                   2007                         1993
## 168                                     NA                           NA
## 169                                     NA                           NA
## 170                                     NA                         1993
## 171                                     NA                           NA
## 172                                     NA                           NA
## 173                                     NA                           NA
## 174                                     NA                           NA
## 175                                     NA                           NA
## 176                                   2003                         1993
## 177                                     NA                           NA
## 178                                   2000                           NA
## 179                                   1997                         1993
## 180                                     NA                           NA
## 181                                     NA                           NA
## 182                                     NA                         1993
```

```
## 183                             NA                          NA
## 184                             NA                          NA
## 185                             NA                          NA
## 186                             NA                        1993
## 187                             NA                        1993
## 188                             NA                          NA
## 189                             NA                          NA
##     SNA.price.valuation Alternative.conversion.factor PPP.survey.year
## 1                                                                  NA
## 2                   VAB                                            NA
## 3                   VAP                       1991-96            2005
## 4                   VAB                                          2005
## 5                   VAB                                            NA
## 6                   VAB                       1971-84            2005
## 7                   VAB                       1990-95            2005
## 8                   VAB                                            NA
## 9                   VAB                                          2005
## 10                  VAB                                          2005
## 11                  VAB                       1992-95            2005
## 12                  VAB                                          2005
## 13                  VAB                                          2005
## 14                  VAP                          1992            2005
## 15                  VAB                       1992-93            2005
## 16                  VAB                                          2005
## 17                  VAB              1978-89, 1991-92            2005
## 18                  VAP                                          2005
## 19                  VAB                                            NA
## 20                  VAB                                          2005
## 21                  VAB                       1990-95            2005
## 22                  VAB                                            NA
## 23                  VAB                                            NA
## 24                  VAB                       1960-85            2005
## 25                  VAB                                          2005
## 26                  VAB                                            NA
## 27                  VAP                                          2005
## 28                  VAB                                          2005
## 29                  VAB                                          2005
## 30                  VAB                                          2005
## 31                  VAB                                          2005
## 32                  VAB                                          2005
## 33                  VAB                                          2005
## 34                  VAP                       1978-93            2005
## 35                  VAP                                          2005
## 36                  VAB                                          2005
## 37                  VAP                          1993            2005
## 38                  VAB                       1992-94            2005
## 39                  VAP                                          2005
## 40                  VAP                                          2005
```

```
## 41                   VAB                                   NA
## 42                   VAP                                   NA
## 43                   VAB                                 2005
## 44                   VAB                                 2005
## 45                   VAB                                 2005
## 46                   VAB                                   NA
## 47                   VAB                                 2005
## 48                   VAB                                   NA
## 49                   VAB                                   NA
## 50                   VAB                                 2005
## 51                   VAB                                 2005
## 52                   VAB                                   NA
## 53                   VAB                                 2005
## 54                   VAB                 1987-95         2005
## 55                   VAB                                 2005
## 56                   VAB                                 2005
## 57                   VAB                                 2005
## 58                   VAB                                 2005
## 59                   VAB                                   NA
## 60                   VAP                    1993         2005
## 61                   VAB                                 2005
## 62                   VAB                 1990-95         2005
## 63                   VAP                 1973-87         2005
## 64                   VAB                                 2005
## 65                   VAB                                 2005
## 66                   VAB                                 2005
## 67                   VAB                 1965-84         2005
## 68                   VAB                                 2005
## 69                   VAB                                   NA
## 70                   VAB                                   NA
## 71                   VAB                                   NA
## 72                   VAB                                 2005
## 73                   VAB                 1988-89           NA
## 74                   VAB                                 2005
## 75                   VAB                    1991           NA
## 76                   VAB                                 2005
## 77                   VAP                                 2005
## 78                   VAB                                 2005
## 79                   VAB                                 2005
## 80                   VAB                 1980-02         2005
## 81                   VAB              1997, 2004         2005
## 82                   VAB                                 2005
## 83                   VAP                                 2005
## 84                   VAB                                 2005
## 85                   VAB                                   NA
## 86                   VAB                                 2005
## 87                   VAB                                 2005
## 88                   VAB                 1987-95         2005
```

```
## 89                  VAB                                        2005
## 90                  VAB                      1990-95           2005
## 91                  VAB                                        2005
## 92                  VAB                                          NA
## 93                  VAB                                          NA
## 94                  VAB                                        2005
## 95                                                               NA
## 96                  VAP                                        2005
## 97                  VAB                                        2005
## 98                  VAB                                        2005
## 99                  VAP                                        2005
## 100                 VAB                                          NA
## 101                 VAP                                        2005
## 102                 VAB                                        2005
## 103                 VAB                      1990-95           2005
## 104                 VAB                                        2005
## 105                 VAB                      1987-95           2005
## 106                 VAB                                        2005
## 107                 VAB                                        2005
## 108                                                              NA
## 109                 VAB                      1990-95           2005
## 110                 VAB                                        2005
## 111                 VAB                                        2005
## 112                 VAB                                        2005
## 113                 VAB                                          NA
## 114                 VAB                                        2005
## 115                 VAB                                        2005
## 116                 VAB                                        2005
## 117                 VAB                                        2005
## 118                 VAB                                        2005
## 119                 VAB                      1992-95           2005
## 120                 VAB                                        2005
## 121                 VAB                                        2005
## 122                 VAB                                        2005
## 123                 VAP                                        2005
## 124                 VAB                                        2005
## 125                 VAP                         1993           2005
## 126                 VAB                      1971-98           2005
## 127                 VAB                      1965-95             NA
## 128                 VAB                                        2005
## 129                 VAB                                        2005
## 130                 VAB                                        2005
## 131                 VAB                                        2005
## 132                 VAP                                        2005
## 133                 VAB                                        2005
## 134                 VAB                                          NA
## 135                 VAB                      1985-90           2005
## 136                 VAP                                        2005
```

```
## 137            VAB                                NA
## 138            VAB                  1989          NA
## 139            VAB                              2005
## 140            VAP                                NA
## 141            VAB                              2005
## 142            VAP                              2005
## 143            VAP                              2005
## 144            VAB        1987-89, 1992         2005
## 145            VAB              1987-95         2005
## 146            VAP                 1994         2005
## 147            VAP                              2005
## 148            VAB                              2005
## 149            VAB                              2005
## 150            VAB                              2005
## 151            VAB                                NA
## 152            VAB                              2005
## 153            VAB                                NA
## 154            VAB                              2005
## 155            VAP                              2005
## 156            VAB                                NA
## 157            VAB                              2005
## 158            VAB                              2005
## 159            VAB                              2005
## 160            VAB                              2005
## 161            VAP                                NA
## 162            VAB              1970-08         2005
## 163            VAB                              2005
## 164            VAP                              2005
## 165            VAP                              2005
## 166            VAB              1990-95         2005
## 167            VAB     1987-95, 1997-07           NA
## 168            VAP                                NA
## 169            VAB                                NA
## 170            VAB                                NA
## 171            VAP                              2005
## 172            VAB                              2005
## 173                                               NA
## 174            VAB                              2005
## 175            VAB                              2005
## 176            VAB              1987-95         2005
## 177            VAB                              2005
## 178            VAB                              2005
## 179            VAB              1990-95           NA
## 180            VAB                                NA
## 181            VAB                              2005
## 182            VAP                 1991         2005
## 183            VAP                                NA
## 184            VAB                                NA
```

```
## 185                 VAP                          1990-96            2005
## 186                 VAB                                             2005
## 187                 VAB                          1999-01            2005
## 188                 VAB                          1990-92            2005
## 189                 VAB                       1991, 1998            2005
##     Balance.of.Payments.Manual.in.use External.debt.Reporting.status
## 1
## 2                                                          Actual
## 3                                BPM5                      Actual
## 4                                BPM5                      Actual
## 5                                BPM4
## 6                                BPM5                      Actual
## 7                                BPM5                      Actual
## 8                                BPM5
## 9                                BPM5
## 10                               BPM5
## 11                               BPM5                      Actual
## 12                               BPM5                      Actual
## 13                               BPM5
## 14                               BPM5                 Preliminary
## 15                               BPM4                      Actual
## 16                               BPM5                 Preliminary
## 17                               BPM5                      Actual
## 18                               BPM5
## 19                               BPM5
## 20                               BPM5                      Actual
## 21                               BPM5                      Actual
## 22                               BPM5                      Actual
## 23
## 24                               BPM5                      Actual
## 25                               BPM5                      Actual
## 26                               BPM5
## 27
## 28                                                         Actual
## 29                               BPM5                 Preliminary
## 30                               BPM4                 Preliminary
## 31                               BPM5
## 32                               BPM5
## 33                               BPM5                      Actual
## 34                               BPM5                 Preliminary
## 35                               BPM5                      Actual
## 36                               BPM5                      Actual
## 37                               BPM5                 Preliminary
## 38                               BPM5                      Actual
## 39                                                    Preliminary
## 40                               BPM5                      Actual
## 41                               BPM5                      Actual
## 42
```

```
## 43                                    BPM5
## 44                                    BPM5
## 45                                    BPM5
## 46                                    BPM5                       Actual
## 47                                    BPM5
## 48                                    BPM5                       Actual
## 49                                    BPM5                       Actual
## 50                                    BPM5                       Actual
## 51                                    BPM5                       Actual
## 52                                    BPM4                       Actual
## 53                                    BPM5
## 54                                    BPM5
## 55                                    BPM5                       Actual
## 56                                    BPM5
## 57                                    BPM4                       Actual
## 58                                    BPM5
## 59
## 60                                    BPM5                  Preliminary
## 61                                    BPM5
## 62                                    BPM5                       Actual
## 63                                    BPM5                       Actual
## 64                                    BPM5                     Estimate
## 65                                    BPM5                     Estimate
## 66                                    BPM5                       Actual
## 67
## 68                                    BPM5
## 69                                    BPM5                       Actual
## 70                                    BPM5                       Actual
## 71                                    BPM5                       Actual
## 72                                    BPM5
## 73                                    BPM5                       Actual
## 74                                    BPM5
## 75                                    BPM5                  Preliminary
## 76                                    BPM5
## 77                                    BPM5                       Actual
## 78                                    BPM5                       Actual
## 79                                    BPM5
## 80                                    BPM5                       Actual
## 81                                    BPM5
## 82                                    BPM5
## 83                                    BPM5
## 84                                    BPM5
## 85                                    BPM5                       Actual
## 86                                    BPM5                       Actual
## 87                                    BPM5
## 88                                    BPM5                       Actual
## 89                                    BPM5                       Actual
## 90                                    BPM5                       Actual
```

```
## 91                               BPM5                         Actual
## 92
## 93                               BPM5                    Preliminary
## 94                               BPM5
## 95
## 96                               BPM5
## 97                               BPM5                    Preliminary
## 98                               BPM5                         Actual
## 99                               BPM5                       Estimate
## 100                              BPM5                         Actual
## 101                              BPM5                         Actual
## 102                              BPM5                         Actual
## 103                              BPM5                         Actual
## 104                              BPM5
## 105                              BPM5                         Actual
## 106                              BPM5
## 107                              BPM5                         Actual
## 108
## 109                              BPM5                         Actual
## 110                              BPM5                         Actual
## 111                              BPM5                         Actual
## 112                              BPM5                         Actual
## 113
## 114                              BPM5                         Actual
## 115                              BPM4                         Actual
## 116                              BPM5
## 117                                                           Actual
## 118                              BPM5                       Estimate
## 119                              BPM5                         Actual
## 120                              BPM4                         Actual
## 121                              BPM5                         Actual
## 122                              BPM5                         Actual
## 123                              BPM5                       Estimate
## 124                              BPM5
## 125                              BPM5                    Preliminary
## 126                              BPM5                    Preliminary
## 127                              BPM5                         Actual
## 128                              BPM5
## 129                              BPM5
## 130                              BPM5                         Actual
## 131                              BPM5
## 132                              BPM5
## 133                              BPM5                         Actual
## 134                              BPM5                         Actual
## 135                              BPM5                         Actual
## 136                              BPM5                         Actual
## 137
## 138                              BPM5                         Actual
```

```
## 139                        BPM5               Actual
## 140
## 141                        BPM5
## 142                        BPM5               Actual
## 143
## 144                        BPM5               Actual
## 145                        BPM5          Preliminary
## 146                        BPM5             Estimate
## 147                        BPM4
## 148                        BPM5               Actual
## 149                        BPM5               Actual
## 150                        BPM5
## 151                        BPM5               Actual
## 152                        BPM5          Preliminary
## 153                        BPM5               Actual
## 154                                           Actual
## 155                                      Preliminary
## 156                        BPM5
## 157                        BPM5
## 158                        BPM5
## 159                        BPM5
## 160                                      Preliminary
## 161                        BPM5               Actual
## 162                        BPM5
## 163                        BPM5               Actual
## 164                        BPM5               Actual
## 165                        BPM5             Estimate
## 166                        BPM5          Preliminary
## 167                        BPM5             Estimate
## 168
## 169                        BPM5               Actual
## 170                        BPM5
## 171                        BPM5               Actual
## 172                        BPM5               Actual
## 173
## 174                        BPM5               Actual
## 175                        BPM5               Actual
## 176                        BPM5               Actual
## 177                        BPM5               Actual
## 178                        BPM5
## 179                        BPM5               Actual
## 180                        BPM5          Preliminary
## 181                        BPM5               Actual
## 182                        BPM4             Estimate
## 183                        BPM5             Estimate
## 184                        BPM5          Preliminary
## 185                        BPM5               Actual
## 186                        BPM5          Preliminary
```

```
## 187                            BPM5                          Estimate
## 188                            BPM5                       Preliminary
## 189                            BPM5                            Actual
##     System.of.trade Government.Accounting.concept
## 1           Special
## 2           General                  Consolidated
## 3           Special
## 4           General                  Consolidated
## 5           General                  Consolidated
## 6           Special                  Consolidated
## 7           Special                  Consolidated
## 8           General
## 9           General                  Consolidated
## 10          Special                  Consolidated
## 11          General                  Consolidated
## 12          Special                  Consolidated
## 13          Special                  Consolidated
## 14          Special                     Budgetary
## 15          General                     Budgetary
## 16          General                  Consolidated
## 17          General                  Consolidated
## 18          General                  Consolidated
## 19          General                     Budgetary
## 20          General                  Consolidated
## 21          General                  Consolidated
## 22          General                     Budgetary
## 23
## 24          Special                  Consolidated
## 25          Special                  Consolidated
## 26          General                  Consolidated
## 27          General
## 28                                   Consolidated
## 29          General                     Budgetary
## 30          Special                     Budgetary
## 31          General                  Consolidated
## 32          Special                  Consolidated
## 33          Special                  Consolidated
## 34          Special                     Budgetary
## 35          Special                  Consolidated
## 36          Special                  Consolidated
## 37          Special                  Consolidated
## 38          Special                     Budgetary
## 39
## 40          Special
## 41          Special                  Consolidated
## 42          General
## 43          General                  Consolidated
## 44          General                  Consolidated
```

```
## 45          Special                   Consolidated
## 46          General
## 47          General                   Consolidated
## 48          General                   Consolidated
## 49          Special                      Budgetary
## 50          Special                      Budgetary
## 51          Special                      Budgetary
## 52          General
## 53          Special                   Consolidated
## 54          General                   Consolidated
## 55          General                   Consolidated
## 56          General                   Consolidated
## 57          General                      Budgetary
## 58          Special                   Consolidated
## 59
## 60          Special
## 61          General                   Consolidated
## 62          General                   Consolidated
## 63          General                      Budgetary
## 64          Special                   Consolidated
## 65          General                   Consolidated
## 66          General
## 67
## 68          Special                   Consolidated
## 69          General                      Budgetary
## 70          Special                      Budgetary
## 71          Special
## 72          General                   Consolidated
## 73          Special                      Budgetary
## 74          General                   Consolidated
## 75          General
## 76          Special                   Consolidated
## 77          Special                   Consolidated
## 78          General                   Consolidated
## 79          General                   Consolidated
## 80          General                   Consolidated
## 81          Special
## 82          General                   Consolidated
## 83          Special                   Consolidated
## 84          Special                   Consolidated
## 85          General                   Consolidated
## 86          General                      Budgetary
## 87          General                   Consolidated
## 88          General                   Consolidated
## 89          General                      Budgetary
## 90          General                   Consolidated
## 91          General                   Consolidated
## 92          General
```

```
## 93          General                    Consolidated
## 94          Special                    Consolidated
## 95
## 96          Special                    Consolidated
## 97          General
## 98          General                       Budgetary
## 99
## 100         General
## 101         General                       Budgetary
## 102         General                    Consolidated
## 103         General                    Consolidated
## 104         Special                    Consolidated
## 105         Special                    Consolidated
## 106         General                    Consolidated
## 107         Special                    Consolidated
## 108
## 109         General                    Consolidated
## 110         Special                    Consolidated
## 111         General                    Consolidated
## 112         General                    Consolidated
## 113
## 114         General
## 115         General                       Budgetary
## 116         General                    Consolidated
## 117
## 118         Special                    Consolidated
## 119         Special
## 120         General
## 121         General                    Consolidated
## 122         General
## 123         General                    Consolidated
## 124         General                       Budgetary
## 125         Special
## 126         General
## 127         Special                       Budgetary
## 128         Special                    Consolidated
## 129         General                    Consolidated
## 130         Special                    Consolidated
## 131         General                    Consolidated
## 132         General                       Budgetary
## 133         General                    Consolidated
## 134         Special                    Consolidated
## 135         Special                    Consolidated
## 136         General                       Budgetary
## 137
## 138         General                       Budgetary
## 139         Special                    Consolidated
## 140         General
```

```
## 141        Special                 Consolidated
## 142        Special                 Consolidated
## 143        General                    Budgetary
## 144        Special                 Consolidated
## 145        General                 Consolidated
## 146        General                 Consolidated
## 147        General
## 148        General                    Budgetary
## 149        Special                    Budgetary
## 150        General                 Consolidated
## 151
## 152        Special                    Budgetary
## 153        Special                 Consolidated
## 154        Special                 Consolidated
## 155        Special
## 156        General
## 157        General                 Consolidated
## 158        Special                 Consolidated
## 159        General                 Consolidated
## 160        General                 Consolidated
## 161        General                 Consolidated
## 162        Special                 Consolidated
## 163        Special
## 164        Special                    Budgetary
## 165        General                 Consolidated
## 166        General                 Consolidated
## 167        General
## 168
## 169
## 170        Special                 Consolidated
## 171        General                 Consolidated
## 172        Special                    Budgetary
## 173
## 174        Special
## 175        General                    Budgetary
## 176        General                 Consolidated
## 177        Special                 Consolidated
## 178        General                 Consolidated
## 179        General
## 180        General                 Consolidated
## 181        General                 Consolidated
## 182        General                 Consolidated
## 183                                Consolidated
## 184        General
## 185        General                    Budgetary
## 186        General                 Consolidated
## 187        Special                 Consolidated
## 188        General                    Budgetary
```

```
## 189         General                  Consolidated
##     IMF.data.dissemination.standard
## 1
## 2                               GDDS
## 3                               GDDS
## 4                               GDDS
## 5                               GDDS
## 6                               SDDS
## 7                               SDDS
## 8                               GDDS
## 9                               SDDS
## 10                              SDDS
## 11                              GDDS
## 12
## 13                              SDDS
## 14                              GDDS
## 15                              GDDS
## 16                              GDDS
## 17                              SDDS
## 18                              GDDS
## 19                              GDDS
## 20
## 21                              SDDS
## 22                              GDDS
## 23
## 24                              GDDS
## 25                              SDDS
## 26                              GDDS
## 27                              GDDS
## 28
## 29                              GDDS
## 30                              GDDS
## 31                              SDDS
## 32                              SDDS
## 33                              SDDS
## 34                              GDDS
## 35                              GDDS
## 36                              GDDS
## 37                              GDDS
## 38                              SDDS
## 39
## 40                              GDDS
## 41                              SDDS
## 42
## 43                              SDDS
## 44                              SDDS
## 45                              SDDS
## 46                              GDDS
```

```
## 47                                   SDDS
## 48                                   GDDS
## 49                                   GDDS
## 50                                   SDDS
## 51                                   SDDS
## 52
## 53                                   SDDS
## 54                                   SDDS
## 55                                   GDDS
## 56                                   SDDS
## 57                                   GDDS
## 58                                   SDDS
## 59
## 60                                   GDDS
## 61                                   SDDS
## 62                                   GDDS
## 63                                   GDDS
## 64                                   GDDS
## 65                                   GDDS
## 66                                   GDDS
## 67
## 68                                   SDDS
## 69                                   GDDS
## 70                                   GDDS
## 71
## 72                                   SDDS
## 73                                   GDDS
## 74                                   SDDS
## 75
## 76                                   SDDS
## 77                                   SDDS
## 78                                   SDDS
## 79                                   SDDS
## 80
## 81                                   GDDS
## 82                                   SDDS
## 83                                   SDDS
## 84                                   SDDS
## 85                                   GDDS
## 86                                   SDDS
## 87                                   SDDS
## 88                                   SDDS
## 89                                   GDDS
## 90                                   SDDS
## 91                                   GDDS
## 92                                   GDDS
## 93                                   GDDS
## 94                                   SDDS
```

```
## 95
## 96                            GDDS
## 97
## 98                            GDDS
## 99                            GDDS
## 100                           GDDS
## 101                           GDDS
## 102                           GDDS
## 103                           SDDS
## 104                           SDDS
## 105                           SDDS
## 106                           GDDS
## 107                           SDDS
## 108
## 109                           SDDS
## 110                           GDDS
## 111
## 112                           SDDS
## 113
## 114                           GDDS
## 115                           GDDS
## 116                           SDDS
## 117
## 118                           GDDS
## 119                           GDDS
## 120                           GDDS
## 121                           GDDS
## 122                           GDDS
## 123                           SDDS
## 124                           GDDS
## 125                           GDDS
## 126                           GDDS
## 127                           GDDS
## 128                           SDDS
## 129                           SDDS
## 130                           GDDS
## 131
## 132                           GDDS
## 133                           GDDS
## 134                           GDDS
## 135                           SDDS
## 136                           SDDS
## 137
## 138
## 139                           SDDS
## 140
## 141                           SDDS
## 142                           GDDS
```

```
## 143                             GDDS
## 144                             SDDS
## 145                             SDDS
## 146                             GDDS
## 147                             GDDS
## 148                             GDDS
## 149                             GDDS
## 150                             SDDS
## 151
## 152                             GDDS
## 153                             SDDS
## 154                             GDDS
## 155                             GDDS
## 156                             GDDS
## 157                             SDDS
## 158                             SDDS
## 159                             SDDS
## 160                             GDDS
## 161                             SDDS
## 162                             GDDS
## 163                             GDDS
## 164                             GDDS
## 165                             SDDS
## 166                             GDDS
## 167
## 168
## 169                             GDDS
## 170                             GDDS
## 171                             SDDS
## 172                             SDDS
## 173
## 174                             GDDS
## 175                             GDDS
## 176                             SDDS
## 177                             SDDS
## 178                             SDDS
## 179
## 180                             GDDS
## 181                             GDDS
## 182                             GDDS
## 183                             GDDS
## 184
## 185                             GDDS
## 186                             SDDS
## 187                             GDDS
## 188                             GDDS
## 189                             GDDS
##     Source.of.most.recent.Income.and.expenditure.data
```

```
## 1
## 2
## 3                                           IHS, 2000
## 4                                          LSMS, 2005
## 5
## 6                                           IHS, 2006
## 7                                           IHS, 2007
## 8
## 9                                         ES/BS, 1994
## 10                                            IS 2000
## 11                                        ES/BS, 2005
## 12                                         CWIQ, 2006
## 13                                          IHS, 2000
## 14                                         CWIQ, 2003
## 15                                         CWIQ, 2003
## 16                                          IHS, 2005
## 17                                        ES/BS, 2003
## 18
## 19
## 20                                         LSMS, 2007
## 21                                         ES/BS 2007
## 22                                         ES/BS 1995
## 23
## 24                                          IHS, 2007
## 25                                          LFS, 2007
## 26
## 27
## 28                                          IHS, 2003
## 29                                     ES/BS, 1993/94
## 30                                           PS, 2003
## 31                                          LFS, 2000
## 32                                        ES/BS, 2000
## 33                                          IHS, 2006
## 34                                          IHS, 2005
## 35                                          IHS, 2002
## 36                                           PS, 2001
## 37                                     CWIQ/ PS, 2005
## 38                                          IHS, 2006
## 39                                          IHS, 2004
## 40                                        ES/BS, 2001
## 41                                          LFS, 2007
## 42
## 43
## 44                                            IS 1996
## 45                                          IHS, 2000
## 46
## 47                                           ITR 1997
## 48                                          IHS, 2005
```

```
## 49                                         IHS, 1995
## 50                                         LFS, 2005
## 51                                    ES/BS, 2004-05
## 52
## 53                                         IHS, 2000
## 54                                       ES/BS, 2004
## 55                                       ES/BS, 2005
## 56                                          IS, 2000
## 57
## 58                                    ES/BS, 1994/95
## 59
## 60                                   CWIQ/ IHS, 2005
## 61                                          IS, 1999
## 62                                         IHS, 2007
## 63                                        LSMS, 2006
## 64                                       CWIQ/, 2003
## 65                                         IHS, 2003
## 66                                        CWIQ, 2002
## 67
## 68                                         IHS, 2000
## 69
## 70                                        LSMS, 2006
## 71                                         IHS, 1998
## 72
## 73                                         IHS, 2006
## 74                                       ES/BS, 2005
## 75                                         IHS, 2001
## 76                                       ES/BS, 2004
## 77                                         IHS, 2007
## 78                                      IHS, 2004/05
## 79                                         IHS, 2000
## 80                                       ES/BS, 2005
## 81
## 82
## 83                                       ES/BS, 2001
## 84                                       ES/BS, 2000
## 85                                        LSMS, 2004
## 86                                       ES/BS, 2006
## 87                                          IS, 1993
## 88                                       ES/BS, 2007
## 89                                      IHS, 2005-06
## 90                                       ES/BS, 2007
## 91                                         IHS, 2007
## 92
## 93
## 94                                       ES/BS, 1998
## 95
## 96
```

```
## 97                                           ES/BS, 2002-03
## 98
## 99                                                CWIQ 2007
## 100                                               IHS, 1995
## 101                                             ES/BS, 2002
## 102                                          ES/BS, 2002-03
## 103                                             ES/BS, 2004
## 104
## 105                                               IHS, 2007
## 106
## 107                                             ES/BS, 2007
## 108
## 109                                             ES/BS, 2007
## 110                                                 PS 2005
## 111
## 112                                               LFS, 2008
## 113
## 114                                             ES/BS, 2006
## 115                                               IHS, 2006
## 116
## 117                                              ES/BS 2007
## 118                                           LSMS, 2006-08
## 119                                          ES/BS, 2002/03
## 120                                               IHS, 2000
## 121
## 122                                           LSMS, 2004-05
## 123                                             ES/BS, 2004
## 124                                          ES/BS, 1993/94
## 125                                            QWIC/PS 2005
## 126                                            IHS, 2003-04
## 127                                              LSMS, 2005
## 128                                               IHS, 1999
## 129                                                IS, 2000
## 130                                           LSMS, 2003/04
## 131                                                IS, 1997
## 132
## 133                                           LSMS, 2004/05
## 134                                               LFS, 2006
## 135                                              LSMS, 2007
## 136                                             ES/BS, 2006
## 137
## 138                                               IHS, 1996
## 139                                             ES/BS, 2005
## 140
## 141                                                IS, 1997
## 142                                               IHS, 2007
## 143
## 144                                               LFS, 2007
```

```
## 145                                            IHS, 2007
## 146                                            IHS, 2000
## 147
## 148
## 149                                              PS 2005
## 150
## 151
## 152                                            IHS, 2003
## 153                                            IHS, 2007
## 154
## 155                                          PS 2000-01
## 156                                          ES/BS, 1999
## 157                                             IS, 1996
## 158                                          ES/BS, 2004
## 159                                             IS, 2000
## 160                                       ES/BS, 2000/01
## 161
## 162
## 163                                          PS, 2002-03
## 164                                           CWIQ, 2006
## 165                                            IHS, 2004
## 166                                           LSMS, 2004
## 167                                           LSMS, 1998
## 168                                           LSMS, 2007
## 169
## 170                                            IHS, 1992
## 171                                            IHS, 2000
## 172                                            LFS, 2006
## 173
## 174                                       ES/BS, 2000/01
## 175                                             PS, 2005
## 176                                          ES/BS, 2008
## 177                                            IHS, 2007
## 178                                             LFS 2000
## 179                                          ES/BS, 2003
## 180
## 181                                            IHS, 2003
## 182                                            IHS, 2006
## 183
## 184
## 185                                          ES/BS, 2005
## 186                                          ES/BS, 2000
## 187                                       1-2-3, 2005-06
## 188                                         IHS, 2004-05
## 189
##     Vital.registration.complete     Latest.agricultural.census
## 1
## 2
```

```
## 3                                                1964-65
## 4                              Yes                  1998
## 5                                                   1998
## 6                              Yes                  2002
## 7                              Yes
## 8                              Yes
## 9                              Yes                  2001
## 10                             Yes             1999-2000
## 11                             Yes
## 12
## 13                             Yes 1999-2000 (conducted annually)
## 14                                                  1992
## 15                                                  1993
## 16                                                  2005
## 17                             Yes
## 18                             Yes
## 19
## 20                             Yes
## 21                             Yes                  1994
## 22
## 23                             Yes
## 24                                             1984-1988
## 25                                                  1996
## 26                             Yes
## 27                             Yes
## 28                                                  2000
## 29                                                  1993
## 30                                                  1985
## 31                             Yes             1996/2001
## 32                             Yes                  2000
## 33                             Yes                  1997
## 34                                                  1997
## 35                                                  2001
## 36                                                  1984
## 37                                             1985-1986
## 38                                                  2001
## 39
## 40                             Yes                  2004
## 41                             Yes                  1973
## 42                             Yes
## 43                             Yes
## 44                             Yes                  2000
## 45                             Yes             1999-2000
## 46                             Yes
## 47                             Yes             1999-2000
## 48                                                  1971
## 49                                                  2001
## 50                                             1999-2000
```

```
## 51                          Yes                      1999-2000
## 52
## 53                          Yes                           1999
## 54                          Yes                           2001
## 55                                                   2001-2002
## 56                          Yes                      1999-2000
## 57                          Yes
## 58                          Yes                      1999-2000
## 59
## 60                                                     1974-75
## 61                          Yes 1999-2000 (conducted annually)
## 62                          Yes                           2004
## 63                                                        1984
## 64                                                   2000-2001
## 65                                                   2001-2002
## 66                                                        1988
## 67
## 68                          Yes                      1999-2000
## 69
## 70                          Yes                           2003
## 71
## 72                          Yes
## 73                                                        1993
## 74                          Yes                           2003
## 75                                                        1971
## 76                          Yes                           2000
## 77                                                        2003
## 78                                         1995-1996/2000-2001
## 79                          Yes                           2000
## 80                          Yes                           2003
## 81                                                        1981
## 82                          Yes
## 83                          Yes                           1981
## 84                          Yes                           2000
## 85                                                        1996
## 86                                                        1997
## 87                          Yes                           2000
## 88                          Yes
## 89                                                   1977-1979
## 90                          Yes                           2002
## 91
## 92
## 93
## 94                          Yes                           2000
## 95
## 96                          Yes                           1970
## 97                                                   1998-1999
## 98                                                   1998-1999
```

```
## 99
## 100                          Yes
## 101                          Yes                        2002
## 102                                                1999-2000
## 103                          Yes                        2003
## 104                          Yes 1999-2000 (conducted annually)
## 105                          Yes                        2001
## 106                          Yes
## 107                                                     1996
## 108
## 109                          Yes
## 110                                                     2004
## 111                          Yes
## 112                                                     1991
## 113
## 114                          Yes                        1994
## 115                                                     1984
## 116                          Yes                        2001
## 117                          Yes
## 118                          Yes
## 119                                                1999-2000
## 120                                                1984-1985
## 121                          Yes
## 122                                                     1993
## 123                          Yes
## 124                                                1996-1997
## 125                                                     1980
## 126                                                     1960
## 127                                                     2001
## 128                          Yes 1999-2000 (conducted annually)
## 129                          Yes                        1999
## 130                                                     2002
## 131                          Yes                        2002
## 132                                                1978-1979
## 133                                                     2000
## 134                                                     2001
## 135                                                     1994
## 136                          Yes                        2002
## 137                          Yes
## 138
## 139                          Yes                   1996/2002
## 140                          Yes                   1997/2002
## 141                          Yes                        1999
## 142                                                     1991
## 143                          Yes                   2000-2001
## 144                          Yes                        2002
## 145                          Yes                     1994-95
## 146                                                     1984
```

```
## 147                                                  1999
## 148
## 149                                             1998-1999
## 150                          Yes
## 151
## 152                                             1984-1985
## 153                          Yes                  1970-71
## 154                          Yes
## 155
## 156                          Yes
## 157                          Yes                     2001
## 158                          Yes                     2000
## 159                          Yes                1999-2000
## 160                                                  2003
## 161                          Yes                     1998
## 162                                                  1981
## 163
## 164                                                  1996
## 165                                                  2003
## 166                                                  1994
## 167                          Yes
## 168
## 169                          Yes                     2001
## 170                          Yes                     2004
## 171                                                  2004
## 172                                                  2001
## 173
## 174                                             2002-2003
## 175                                                  1991
## 176                          Yes
## 177                          Yes                     2000
## 178                          Yes                1997/2002
## 179                          Yes
## 180                          Yes
## 181                          Yes                     1997
## 182                                                  2001
## 183
## 184                                                  1999
## 185                                                  2002
## 186                                                  2000
## 187                                                  1990
## 188                                                  1990
## 189                                                  1960
##     Latest.industrial.data Latest.trade.data Latest.water.withdrawal.data
## 1                       NA              2008                           NA
## 2                       NA              2008                         2000
## 3                       NA              1991                         2000
## 4                     2005              2008                         2000
```

```
## 5                         NA               2008                         2005
## 6                       2001               2008                         2000
## 7                         NA               2008                         2000
## 8                         NA               2007                         1990
## 9                       2004               2008                         2000
## 10                      2004               2008                         2000
## 11                      2005               2008                         2005
## 12                        NA               2008                         2000
## 13                      2004               2008                           NA
## 14                        NA               2005                         2001
## 15                        NA               2005                         2000
## 16                      1997               2007                         2000
## 17                      2005               2008                         2000
## 18                        NA               2007                         2003
## 19                      1997               2008                           NA
## 20                        NA               2008                           NA
## 21                        NA               2008                         2000
## 22                        NA               2008                         2000
## 23                        NA               2008                           NA
## 24                      2000               2008                         2000
## 25                      2004               2008                         2000
## 26                        NA               2008                         2000
## 27                        NA               2006                           NA
## 28                        NA               2008                         2000
## 29                      2005               2008                         2000
## 30                        NA               2005                         2000
## 31                      2001               2008                         2000
## 32                        NA               2008                         2000
## 33                      2005               2008                         2000
## 34                      2005               2008                         2000
## 35                        NA               2008                           NA
## 36                        NA               2006                         2000
## 37                        NA               1995                         2002
## 38                      2004               2008                         2000
## 39                        NA               2007                           NA
## 40                        NA               2008                           NA
## 41                        NA               2008                         2000
## 42                        NA               2006                         2000
## 43                      2005               2008                         2000
## 44                      2005               2008                         2000
## 45                      2004               2008                         2000
## 46                        NA               2008                           NA
## 47                      2004               2008                         2000
## 48                        NA               2008                         2000
## 49                        NA               2007                         2000
## 50                      2004               2008                         2000
## 51                      2001               2008                         2000
## 52                      2005               2003                         2004
```

```
## 53                         2004              2008                         2000
## 54                         2005              2008                         2000
## 55                         2005              2008                         2002
## 56                         2004              2008                         2000
## 57                         2003              2007                         2000
## 58                         2004              2008                         2000
## 59                           NA                NA                           NA
## 60                           NA              2006                         2000
## 61                         2004              2008                         2000
## 62                         2005              2008                         2005
## 63                         2002              2008                         2000
## 64                           NA              2008                         2000
## 65                           NA              2008                         2000
## 66                           NA              2005                         2000
## 67                           NA                NA                         2000
## 68                         2003              2008                         2000
## 69                           NA              2008                           NA
## 70                           NA              2008                         2000
## 71                           NA              2008                         2000
## 72                           NA              2008                           NA
## 73                           NA              2007                         2000
## 74                           NA              2008                           NA
## 75                           NA              1997                         2000
## 76                         2004              2008                         2000
## 77                         2004              2008                         2000
## 78                         2003              2008                         2000
## 79                         2004              2008                         2000
## 80                         2004              2006                         2004
## 81                         1996              2008                         2000
## 82                         2004              2008                         2000
## 83                         2004              2008                         2004
## 84                         2004              2008                         2000
## 85                           NA              2008                         2000
## 86                         2005              2008                         2005
## 87                         2004              2008                         2000
## 88                           NA              2008                         2000
## 89                         2005              2008                         2003
## 90                         2004              2007                         2000
## 91                         1999              2004                         2000
## 92                           NA              2005                           NA
## 93                           NA              2007                           NA
## 94                         2005              2008                         2000
## 95                           NA                NA                           NA
## 96                           NA              2007                         2002
## 97                         1998              1975                         2000
## 98                         1997              2008                         2005
## 99                           NA              1985                         2000
## 100                          NA              2008                           NA
```

```
## 101                   2005              2008                        2000
## 102                     NA              2004                        2000
## 103                   2005              2008                        2000
## 104                   2004              2008                          NA
## 105                   2005              2008                        2000
## 106                     NA              2008                          NA
## 107                   2005              2008                        2000
## 108                     NA                NA                          NA
## 109                   2004              2008                        2000
## 110                   2005              2008                        2000
## 111                     NA              2008                          NA
## 112                   1999              2008                        2000
## 113                     NA                NA                          NA
## 114                   2000              2008                          NA
## 115                     NA              2008                        2000
## 116                   2004              2008                        2000
## 117                     NA                NA                          NA
## 118                   1999              2007                        2000
## 119                     NA              2008                        2000
## 120                     NA              2008                        2000
## 121                   2003              2008                        2003
## 122                   2000              2008                        2000
## 123                   2004              2008                        2000
## 124                     NA              2008                        2000
## 125                     NA              2008                        2000
## 126                     NA              2008                        2000
## 127                     NA              2007                        2000
## 128                   2004              2008                          NA
## 129                   2003              2008                        2000
## 130                   2001              2002                        2000
## 131                   2003              2008                        2000
## 132                   2005              2008                        2003
## 133                     NA              2008                        2000
## 134                   2000              2008                        2000
## 135                   2006              2008                        2000
## 136                   2004              2008                        2000
## 137                     NA                NA                          NA
## 138                     NA              2004                        2000
## 139                   2004              2008                        2000
## 140                     NA                NA                          NA
## 141                   2004              2008                        2000
## 142                     NA              2008                        2000
## 143                   2005              2008                        2005
## 144                   2005              2008                        2000
## 145                   2005              2008                        2000
## 146                   1998              2008                        2000
## 147                   2005              2007                        2006
## 148                   2000              2008                        2000
```

```
## 149                 2001             2008                            2002
## 150                 2005             2008                              NA
## 151                   NA             2007                              NA
## 152                   NA             2002                            2000
## 153                   NA             2008                            2000
## 154                   NA             2008                              NA
## 155                   NA             2008                              NA
## 156                   NA             2008                            2000
## 157                 2004             2008                              NA
## 158                 2005             2008                              NA
## 159                 2004             2008                            2000
## 160                   NA             2007                            2000
## 161                   NA             2008                            2003
## 162                   NA             2007                            2003
## 163                   NA             1996                            2000
## 164                   NA             2007                            2002
## 165                 1999             2008                            2000
## 166                   NA             2000                            2000
## 167                   NA             2000                            2000
## 168                   NA             2005                              NA
## 169                   NA             2007                              NA
## 170                 2005             2008                            2000
## 171                   NA             2008                            2000
## 172                 2000             2008                            2003
## 173                   NA               NA                              NA
## 174                   NA             2007                            2002
## 175                 2001             2008                              NA
## 176                   NA             2008                            2000
## 177                 2004             2008                            2000
## 178                 2004             2008                            2000
## 179                   NA               NA                            2000
## 180                   NA             2008                              NA
## 181                   NA             2008                              NA
## 182                 1999             2008                            2000
## 183                   NA             2007                              NA
## 184                   NA             2008                              NA
## 185                 2005             2008                            2000
## 186                 2005             2008                            2000
## 187                   NA             1986                            2000
## 188                   NA             2008                            2000
## 189                 1995             2008                            2002
##     X2.alpha.code WB.2.code                     Table.Name
## 1              AW        AW                          Aruba
## 2              AF        AF                    Afghanistan
## 3              AO        AO                         Angola
## 4              AL        AL                        Albania
## 5              AE        AE           United Arab Emirates
## 6              AR        AR                      Argentina
```

```
## 7                  AM        AM                     Armenia
## 8                  AG        AG         Antigua and Barbuda
## 9                  AU        AU                   Australia
## 10                 AT        AT                     Austria
## 11                 AZ        AZ                  Azerbaijan
## 12                 BI        BI                     Burundi
## 13                 BE        BE                     Belgium
## 14                 BJ        BJ                       Benin
## 15                 BF        BF                Burkina Faso
## 16                 BD        BD                  Bangladesh
## 17                 BG        BG                    Bulgaria
## 18                 BH        BH                     Bahrain
## 19                 BS        BS                Bahamas, The
## 20                 BA        BA      Bosnia and Herzegovina
## 21                 BY        BY                     Belarus
## 22                 BZ        BZ                      Belize
## 23                 BM        BM                     Bermuda
## 24                 BO        BO                     Bolivia
## 25                 BR        BR                      Brazil
## 26                 BB        BB                    Barbados
## 27                 BN        BN           Brunei Darussalam
## 28                 BT        BT                      Bhutan
## 29                 BW        BW                    Botswana
## 30                 CF        CF    Central African Republic
## 31                 CA        CA                      Canada
## 32                 CH        CH                 Switzerland
## 33                 CL        CL                       Chile
## 34                 CN        CN                       China
## 35                 CI        CI             C\xf4te d'Ivoire
## 36                 CM        CM                    Cameroon
## 37                 CG        CG                 Congo, Rep.
## 38                 CO        CO                    Colombia
## 39                 KM        KM                     Comoros
## 40                 CV        CV                  Cape Verde
## 41                 CR        CR                  Costa Rica
## 42                 CU        CU                        Cuba
## 43                 CY        CY                      Cyprus
## 44                 CZ        CZ              Czech Republic
## 45                 DE        DE                     Germany
## 46                 DM        DM                    Dominica
## 47                 DK        DK                     Denmark
## 48                 DO        DO          Dominican Republic
## 49                 DZ        DZ                     Algeria
## 50                 EC        EC                     Ecuador
## 51                 EG        EG            Egypt, Arab Rep.
## 52                 ER        ER                     Eritrea
## 53                 ES        ES                       Spain
## 54                 EE        EE                     Estonia
```

```
## 55            ET      ET                   Ethiopia
## 56            FI      FI                    Finland
## 57            FJ      FJ                       Fiji
## 58            FR      FR                     France
## 59            FM      FM      Micronesia, Fed. Sts.
## 60            GA      GA                      Gabon
## 61            GB      GB             United Kingdom
## 62            GE      GE                    Georgia
## 63            GH      GH                      Ghana
## 64            GN      GN                     Guinea
## 65            GM      GM                Gambia, The
## 66            GW      GW              Guinea-Bissau
## 67            GQ      GQ          Equatorial Guinea
## 68            GR      GR                     Greece
## 69            GD      GD                    Grenada
## 70            GT      GT                  Guatemala
## 71            GY      GY                     Guyana
## 72            HK      HK       Hong Kong SAR, China
## 73            HN      HN                   Honduras
## 74            HR      HR                    Croatia
## 75            HT      HT                      Haiti
## 76            HU      HU                    Hungary
## 77            ID      ID                  Indonesia
## 78            IN      IN                      India
## 79            IE      IE                    Ireland
## 80            IR      IR         Iran, Islamic Rep.
## 81            IQ      IQ                       Iraq
## 82            IS      IS                    Iceland
## 83            IL      IL                     Israel
## 84            IT      IT                      Italy
## 85            JM      JM                    Jamaica
## 86            JO      JO                     Jordan
## 87            JP      JP                      Japan
## 88            KZ      KZ                 Kazakhstan
## 89            KE      KE                      Kenya
## 90            KG      KG            Kyrgyz Republic
## 91            KH      KH                   Cambodia
## 92            KI      KI                   Kiribati
## 93            KN      KN        St. Kitts and Nevis
## 94            KR      KR                Korea, Rep.
## 95                    KV                     Kosovo
## 96            KW      KW                     Kuwait
## 97            LA      LA                    Lao PDR
## 98            LB      LB                    Lebanon
## 99            LR      LR                    Liberia
## 100           LC      LC                  St. Lucia
## 101           LK      LK                  Sri Lanka
## 102           LS      LS                    Lesotho
```

```
## 103       LT    LT              Lithuania
## 104       LU    LU             Luxembourg
## 105       LV    LV                 Latvia
## 106       MO    MO       Macao SAR, China
## 107       MA    MA                Morocco
## 108       MC    MC                 Monaco
## 109       MD    MD                Moldova
## 110       MG    MG             Madagascar
## 111       MV    MV               Maldives
## 112       MX    MX                 Mexico
## 113       MH    MH       Marshall Islands
## 114       MK    MK        Macedonia, FYR
## 115       ML    ML                   Mali
## 116       MT    MT                  Malta
## 117       ME    ME             Montenegro
## 118       MN    MN               Mongolia
## 119       MZ    MZ             Mozambique
## 120       MR    MR             Mauritania
## 121       MU    MU              Mauritius
## 122       MW    MW                 Malawi
## 123       MY    MY               Malaysia
## 124     <NA>  <NA>                Namibia
## 125       NE    NE                  Niger
## 126       NG    NG                Nigeria
## 127       NI    NI              Nicaragua
## 128       NL    NL            Netherlands
## 129       NO    NO                 Norway
## 130       NP    NP                  Nepal
## 131       NZ    NZ            New Zealand
## 132       OM    OM                   Oman
## 133       PK    PK               Pakistan
## 134       PA    PA                 Panama
## 135       PE    PE                   Peru
## 136       PH    PH            Philippines
## 137       PW    PW                  Palau
## 138       PG    PG       Papua New Guinea
## 139       PL    PL                 Poland
## 140       PR    PR            Puerto Rico
## 141       PT    PT               Portugal
## 142       PY    PY               Paraguay
## 143       QA    QA                  Qatar
## 144       RO    RO                Romania
## 145       RU    RU     Russian Federation
## 146       RW    RW                 Rwanda
## 147       SA    SA           Saudi Arabia
## 148       SD    SD                  Sudan
## 149       SN    SN                Senegal
## 150       SG    SG              Singapore
```

```
## 151            SB        SB                   Solomon Islands
## 152            SL        SL                      Sierra Leone
## 153            SV        SV                       El Salvador
## 154            RS        YF                            Serbia
## 155            ST        ST   S\xe3o Tom\xe9 and Principe
## 156            SR        SR                          Suriname
## 157            SK        SK                   Slovak Republic
## 158            SI        SI                          Slovenia
## 159            SE        SE                            Sweden
## 160            SZ        SZ                         Swaziland
## 161            SC        SC                        Seychelles
## 162            SY        SY              Syrian Arab Republic
## 163            TD        TD                              Chad
## 164            TG        TG                              Togo
## 165            TH        TH                          Thailand
## 166            TJ        TJ                        Tajikistan
## 167            TM        TM                      Turkmenistan
## 168            TL        TP                       Timor-Leste
## 169            TO        TO                             Tonga
## 170            TT        TT               Trinidad and Tobago
## 171            TN        TN                           Tunisia
## 172            TR        TR                            Turkey
## 173            TV        TV                            Tuvalu
## 174            TZ        TZ                          Tanzania
## 175            UG        UG                            Uganda
## 176            UA        UA                           Ukraine
## 177            UY        UY                           Uruguay
## 178            US        US                     United States
## 179            UZ        UZ                        Uzbekistan
## 180            VC        VC St. Vincent and the Grenadines
## 181            VE        VE                     Venezuela, RB
## 182            VN        VN                           Vietnam
## 183            VU        VU                           Vanuatu
## 184            WS        WS                             Samoa
## 185            YE        RY                      Yemen, Rep.
## 186            ZA        ZA                      South Africa
## 187            CD        ZR                  Congo, Dem. Rep.
## 188            ZM        ZM                            Zambia
## 189            ZW        ZW                          Zimbabwe
##                   Short.Name
## 1                      Aruba
## 2                Afghanistan
## 3                     Angola
## 4                    Albania
## 5       United Arab Emirates
## 6                  Argentina
## 7                    Armenia
## 8        Antigua and Barbuda
```

```
## 9                       Australia
## 10                        Austria
## 11                     Azerbaijan
## 12                        Burundi
## 13                        Belgium
## 14                          Benin
## 15                   Burkina Faso
## 16                     Bangladesh
## 17                       Bulgaria
## 18                        Bahrain
## 19                    The Bahamas
## 20         Bosnia and Herzegovina
## 21                        Belarus
## 22                         Belize
## 23                        Bermuda
## 24                        Bolivia
## 25                         Brazil
## 26                       Barbados
## 27                         Brunei
## 28                         Bhutan
## 29                       Botswana
## 30       Central African Republic
## 31                         Canada
## 32                    Switzerland
## 33                          Chile
## 34                          China
## 35               C\xf4te d'Ivoire
## 36                       Cameroon
## 37                          Congo
## 38                       Colombia
## 39                        Comoros
## 40                     Cape Verde
## 41                     Costa Rica
## 42                           Cuba
## 43                         Cyprus
## 44                 Czech Republic
## 45                        Germany
## 46                       Dominica
## 47                        Denmark
## 48             Dominican Republic
## 49                        Algeria
## 50                        Ecuador
## 51                          Egypt
## 52                        Eritrea
## 53                          Spain
## 54                        Estonia
## 55                       Ethiopia
## 56                        Finland
```

```
## 57                              Fiji
## 58                            France
## 59                        Micronesia
## 60                             Gabon
## 61                    United Kingdom
## 62                           Georgia
## 63                             Ghana
## 64                            Guinea
## 65                        The Gambia
## 66                     Guinea-Bissau
## 67                 Equatorial Guinea
## 68                            Greece
## 69                           Grenada
## 70                         Guatemala
## 71                            Guyana
## 72              Hong Kong SAR, China
## 73                          Honduras
## 74                           Croatia
## 75                             Haiti
## 76                           Hungary
## 77                         Indonesia
## 78                             India
## 79                           Ireland
## 80                              Iran
## 81                              Iraq
## 82                           Iceland
## 83                            Israel
## 84                             Italy
## 85                           Jamaica
## 86                            Jordan
## 87                             Japan
## 88                        Kazakhstan
## 89                             Kenya
## 90                   Kyrgyz Republic
## 91                          Cambodia
## 92                          Kiribati
## 93               St. Kitts and Nevis
## 94                             Korea
## 95                            Kosovo
## 96                            Kuwait
## 97                           Lao PDR
## 98                           Lebanon
## 99                           Liberia
## 100                        St. Lucia
## 101                        Sri Lanka
## 102                          Lesotho
## 103                        Lithuania
## 104                       Luxembourg
```

```
## 105                  Latvia
## 106      Macao SAR, China
## 107                 Morocco
## 108                  Monaco
## 109                 Moldova
## 110              Madagascar
## 111                Maldives
## 112                  Mexico
## 113        Marshall Islands
## 114               Macedonia
## 115                    Mali
## 116                   Malta
## 117              Montenegro
## 118                Mongolia
## 119              Mozambique
## 120              Mauritania
## 121               Mauritius
## 122                  Malawi
## 123                Malaysia
## 124                 Namibia
## 125                   Niger
## 126                 Nigeria
## 127               Nicaragua
## 128             Netherlands
## 129                  Norway
## 130                   Nepal
## 131             New Zealand
## 132                    Oman
## 133                Pakistan
## 134                  Panama
## 135                    Peru
## 136             Philippines
## 137                   Palau
## 138        Papua New Guinea
## 139                  Poland
## 140             Puerto Rico
## 141                Portugal
## 142                Paraguay
## 143                   Qatar
## 144                 Romania
## 145                  Russia
## 146                  Rwanda
## 147            Saudi Arabia
## 148                   Sudan
## 149                 Senegal
## 150               Singapore
## 151         Solomon Islands
## 152            Sierra Leone
```

```
## 153                         El Salvador
## 154                              Serbia
## 155    S\xe3o Tom\xe9 and Principe
## 156                            Suriname
## 157                     Slovak Republic
## 158                            Slovenia
## 159                              Sweden
## 160                           Swaziland
## 161                          Seychelles
## 162                Syrian Arab Republic
## 163                                Chad
## 164                                Togo
## 165                            Thailand
## 166                          Tajikistan
## 167                        Turkmenistan
## 168                         Timor-Leste
## 169                               Tonga
## 170                 Trinidad and Tobago
## 171                             Tunisia
## 172                              Turkey
## 173                              Tuvalu
## 174                            Tanzania
## 175                              Uganda
## 176                             Ukraine
## 177                             Uruguay
## 178                       United States
## 179                          Uzbekistan
## 180 St. Vincent and the Grenadines
## 181                           Venezuela
## 182                             Vietnam
## 183                             Vanuatu
## 184                               Samoa
## 185                               Yemen
## 186                        South Africa
## 187                     Dem. Rep. Congo
## 188                              Zambia
## 189                            Zimbabwe
```

```r
count(mergedData)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   189
```

- *"Sort the data frame in descending order by GDP rank (so United States is last). What is the 13th country in the resulting data frame?"*

```r
sortedData <- arrange(mergedData, desc(as.numeric(as.character(Ranking))))
sortedData[13,]
```

```
##    CountryCode         Long.Name.x Ranking mil.US.dollars         Long.Name.y
## 13         KNA St. Kitts and Nevis     178            767  St. Kitts and Nevis
##           Income.Group                     Region Lending.category Other.groups
## 13 Upper middle income Latin America & Caribbean             IBRD
##            Currency.Unit Latest.population.census Latest.household.survey
## 13 East Caribbean dollar                     2001
##    Special.Notes National.accounts.base.year National.accounts.reference.year
## 13                                      1990                               NA
##    System.of.National.Accounts SNA.price.valuation
## 13                        1993                 VAB
##    Alternative.conversion.factor PPP.survey.year
## 13                                            NA
##    Balance.of.Payments.Manual.in.use External.debt.Reporting.status
## 13                              BPM5                    Preliminary
##    System.of.trade Government.Accounting.concept
## 13         General                  Consolidated
##    IMF.data.dissemination.standard
## 13                            GDDS
##    Source.of.most.recent.Income.and.expenditure.data
## 13
##    Vital.registration.complete Latest.agricultural.census
## 13
##    Latest.industrial.data Latest.trade.data Latest.water.withdrawal.data
## 13                     NA              2007                           NA
##    X2.alpha.code WB.2.code          Table.Name          Short.Name
## 13            KN        KN St. Kitts and Nevis St. Kitts and Nevis
```

4)

- *"What is the average GDP ranking for the"High income: OECD" and "High income: nonOECD" group? "*

```r
groupedData <- group_by(sortedData, Income.Group)
summarize(groupedData, mean(as.numeric(as.character(Ranking)), na.rm = TRUE))
```

```
## # A tibble: 5 x 2
##   Income.Group         `mean(as.numeric(as.character(Ranking)), na.rm = TRUE)`
##   <fct>                                                                  <dbl>
## 1 High income: nonOECD                                                    91.9
## 2 High income: OECD                                                       33.0
## 3 Low income                                                             134.
## 4 Lower middle income                                                    108.
## 5 Upper middle income                                                     92.1
```

5) *"Cut the GDP ranking into 5 separate quantile groups. Make a table versus Income.Group. How many countries are Lower middle income but among the 38 nations with highest GDP?"*

```r
quantile(as.numeric(as.character(sortedData$Ranking)), probs = c(seq(0.2:1, by=0.2)))
```

```
##   20%   40%   60%   80%  100%
##  38.6  76.2 113.8 152.4 190.0
```

```r
bestOfTheWorst <- filter(sortedData,
                    as.numeric(as.character(Ranking)) <= 38, as.character(Income.Group) == "Lo
count(bestOfTheWorst)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1     5
```

Text and Date Manipulation in R

Editing Text Variables

- Using Baltimore automated Speed Cameras data

```r
cameraData <- read.csv(paste(getwd(), "/data/cameras.csv", sep = ""))
names(cameraData)
```

```
## [1] "address"                      "direction"
## [3] "street"                       "crossStreet"
## [5] "intersection"                 "Location.1"
## [7] "X2010.Census.Neighborhoods"   "X2010.Census.Wards.Precincts"
## [9] "Zip.Codes"
```

- `tolower` function (There's also a `toupper` function)

```r
tolower(names(cameraData))#making all lower case helps reduce your own errors in typing
```

```
## [1] "address"                      "direction"
## [3] "street"                       "crossstreet"
## [5] "intersection"                 "location.1"
## [7] "x2010.census.neighborhoods"   "x2010.census.wards.precincts"
## [9] "zip.codes"
```

- Separated character vectors by a token with `strsplit`

```r
#Have to use '//' when refering to reserved chars
splitNames <- strsplit(names(cameraData), "\\.")
splitNames[[6]]
```

```
## [1] "Location" "1"
```

```r
#strsplit returns a list
splitNames
```

```
## [[1]]
```

```
## [1] "address"
##
## [[2]]
## [1] "direction"
##
## [[3]]
## [1] "street"
##
## [[4]]
## [1] "crossStreet"
##
## [[5]]
## [1] "intersection"
##
## [[6]]
## [1] "Location" "1"
##
## [[7]]
## [1] "X2010"         "Census"        "Neighborhoods"
##
## [[8]]
## [1] "X2010"     "Census"    "Wards"     "Precincts"
##
## [[9]]
## [1] "Zip"   "Codes"
```

- Quick aside on lists

```
mylist <- list(letters = c("A", "b", "c"), numbers = 1:3, matrix(1:25, ncol = 5))
head(mylist)
```

```
## $letters
## [1] "A" "b" "c"
##
## $numbers
## [1] 1 2 3
##
## [[3]]
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    6   11   16   21
## [2,]    2    7   12   17   22
## [3,]    3    8   13   18   23
## [4,]    4    9   14   19   24
## [5,]    5   10   15   20   25
```

```
#Returns first list and it's name
class(mylist[1])
```

```
## [1] "list"
```

```r
mylist[1]
```

```
## $letters
## [1] "A" "b" "c"
```

```r
#Returns vector of given name
class(mylist$letters)
```

```
## [1] "character"
```

```r
mylist$letters
```

```
## [1] "A" "b" "c"
```

```r
#Returns first vector
class(mylist[[1]])
```

```
## [1] "character"
```

```r
mylist[[1]]
```

```
## [1] "A" "b" "c"
```

- Fixing character vectors with `sapply`

```r
splitNames[[6]][1]
```

```
## [1] "Location"
```

```r
firstElement <- function(x){x[1]}
sapply(splitNames, firstElement)
```

```
## [1] "address"      "direction"    "street"       "crossStreet"  "intersection"
## [6] "Location"     "X2010"        "X2010"        "Zip"
```

- Now using peer review experiment data

```r
reviews <- read.csv(paste(getwd(), "/data/review.csv", sep =""))
solutions <- read.csv(paste(getwd(), "/data/solutions.csv", sep =""))
head(reviews)
```

```
##   id solution_id reviewer_id      start       stop time_left accept
## 1  1           3          27 1304095698 1304095758      1754      1
## 2  2           4          22 1304095188 1304095206      2306      1
## 3  3           5          28 1304095276 1304095320      2192      1
## 4  4           1          26 1304095267 1304095423      2089      1
## 5  5          10          29 1304095456 1304095469      2043      1
## 6  6           2          29 1304095471 1304095513      1999      1
```

```r
head(solutions)
```

```
##   id problem_id subject_id      start       stop time_left answer
## 1  1        156         29 1304095119 1304095169      2343      B
## 2  2        269         25 1304095119 1304095183      2329      C
## 3  3         34         22 1304095127 1304095146      2366      C
```

```
## 4  4         19         23 1304095127 1304095150      2362      D
## 5  5        605         26 1304095127 1304095167      2345      A
## 6  6        384         27 1304095131 1304095270      2242      C
```

- Using the `sub()` function
  - substitues `pattern` with `replacement` in the given vector, `x` in the first instance

```
names(reviews)
```

```
## [1] "id"          "solution_id" "reviewer_id" "start"       "stop"
## [6] "time_left"   "accept"
```

```
sub("_", "", names(reviews),)
```

```
## [1] "id"         "solutionid" "reviewerid" "start"      "stop"
## [6] "timeleft"   "accept"
```

- Using the `gsub()` function to replace all of a certain character

```
testName <- "this_is_a_test"
sub("_", "", testName)
```

```
## [1] "thisis_a_test"
```

```
gsub("_", "", testName)
```

```
## [1] "thisisatest"
```

Finding specific characters

```
#Looking at subset in range of result
cameraData$intersection[61:80]
```

```
##  [1] Caton Ave & Benson Ave              Cold Spring  & Hillen Road
##  [3] Russell \n & Hamburg St             Eastern  & Kane St
##  [5] E 33rd  & The Alameda               North Ave  & Howard St
##  [7] Reistertown Rd\n  & Druid Lake Drive Gwynns Falls \n  & Garrison Blvd
##  [9] The Alameda  & 33rd St              Madison  & Caroline St
## [11] Northern Pkwy  & Greenspring Ave    Erdman  & Macon St
## [13] Wilkens  & DeSoto                   Northern Pkwy  & Falls Road
## [15] Reisterstown \n & Menlo Drive       Perring Pkwy\n  & Belvedere Ave
## [17] Wilkens Ave & Pine Heights          Monroe\n & Lafayette
## [19] Harford \n & The Alameda            Caton Ave & Benson Ave
## 74 Levels: \nPulaski Hwy \n & Moravia Park Drive  & ... York Rd \n & Gitting Ave
```

```
#Search for a particular string withing the vectors
grep("Alameda", cameraData$intersection)
```

```
## [1] 65 69 79
```

```
#Counting how many times a particular string appears
table(grepl("Alameda", cameraData$intersection))
```

```
##
## FALSE  TRUE
##    77     3
```

```
#Using grepl to subset a certain string to remove
cameraData2 <- cameraData[!grepl("Alameda", cameraData$intersection),]
cameraData2[61:80,]
```

```
##                                   address direction            street
## 61             S CATON AVE & BENSON AVE       S/B         Caton Ave
## 62         E COLD SPRING LN & HILLEN RD       W/B       Cold Spring
## 63            RUSSELL ST & W HAMBURG ST       N/B        Russell \n
## 64                EASTERN AVE & KANE ST       E/B           Eastern
## 66            W NORTH AVE & N HOWARD ST       W/B         North Ave
## 67      REISTERSTOWN RD & DRUID PARK DR       S/B Reistertown Rd\n
## 68           GWYNNS FLS & GARRISON BLVD       E/B   Gwynns Falls \n
## 70         E MADISON ST & N CAROLINE ST       W/B           Madison
## 71   W NORTHERN PKWY & GREENSPRING AVE       E/B     Northern Pkwy
## 72               ERDMAN AVE & N MACON ST       E/B            Erdman
## 73               WILKENS AVE & DESOTO RD       E/B           Wilkens
## 74           W NORTHERN PKWY & FALLS RD       W/B     Northern Pkwy
## 75           REISTERSTOWN RD & MENLO DR       N/B   Reisterstown \n
## 76        PERRING PKWY & E BELVEDERE AVE       S/B    Perring Pkwy\n
## 77        WILKENS AVE & PINE HEIGHTS AVE       E/B       Wilkens Ave
## 78         N MONROE ST & W LAFAYETTE AVE       S/B          Monroe\n
## 80             S CATON AVE & BENSON AVE       N/B         Caton Ave
## NA                                   <NA>      <NA>              <NA>
## NA.1                                 <NA>      <NA>              <NA>
## NA.2                                 <NA>      <NA>              <NA>
##            crossStreet                       intersection
## 61          Benson Ave             Caton Ave & Benson Ave
## 62         Hillen Road           Cold Spring  & Hillen Road
## 63          Hamburg St              Russell \n & Hamburg St
## 64             Kane St                   Eastern  & Kane St
## 66           Howard St              North Ave  & Howard St
## 67   Druid Lake Drive Reistertown Rd\n  & Druid Lake Drive
## 68       Garrison Blvd     Gwynns Falls \n  & Garrison Blvd
## 70         Caroline St              Madison  & Caroline St
## 71     Greenspring Ave     Northern Pkwy  & Greenspring Ave
## 72            Macon St                   Erdman  & Macon St
## 73              DeSoto                    Wilkens  & DeSoto
## 74          Falls Road          Northern Pkwy  & Falls Road
## 75         Menlo Drive        Reisterstown \n & Menlo Drive
## 76       Belvedere Ave      Perring Pkwy\n  & Belvedere Ave
## 77        Pine Heights         Wilkens Ave & Pine Heights
## 78           Lafayette                 Monroe\n & Lafayette
## 80          Benson Ave             Caton Ave & Benson Ave
## NA                <NA>                                <NA>
```

```
## NA.1             <NA>                                    <NA>
## NA.2             <NA>                                    <NA>
##                   Location.1 X2010.Census.Neighborhoods
## 61    (39.269316, -76.66897)                        268
## 62   (39.345907, -76.585927)                        161
## 63   (39.279786, -76.623754)                        228
## 64   (39.287763, -76.537102)                        126
## 66   (39.311087, -76.619307)                         44
## 67   (39.325287, -76.657711)                        187
## 68   (39.313579, -76.676225)                        164
## 70   (39.299326, -76.597676)                         88
## 71   (39.355024, -76.660459)                         42
## 72   (39.306805, -76.559317)                          6
## 73   (39.274904, -76.668163)                        100
## 74    (39.361413, -76.64622)                        214
## 75   (39.351985, -76.696376)                         89
## 76   (39.354963, -76.575726)                         64
## 77   (39.272025, -76.676961)                        220
## 78   (39.298743, -76.647517)                        153
## 80   (39.269378, -76.668819)                        268
## NA                      <NA>                         NA
## NA.1                    <NA>                         NA
## NA.2                    <NA>                         NA
##      X2010.Census.Wards.Precincts Zip.Codes
## 61                            166     27944
## 62                            222     28570
## 63                              1     27953
## 64                            135     27935
## 66                            160     27307
## 67                             40     27295
## 68                             45     27297
## 70                            127     13987
## 71                            251     27295
## 72                            145     13987
## 73                            164     27632
## 74                            254     14004
## 75                            246     27295
## 76                            224     28564
## 77                            166     27950
## 78                             88     27301
## 80                            166     27632
## NA                             NA        NA
## NA.1                           NA        NA
## NA.2                           NA        NA
```

- More on `grep`

```r
#Return values rather than index
grep("Alameda", cameraData$intersection, value = TRUE)
```

```
## [1] "E 33rd  & The Alameda"    "The Alameda  & 33rd St"
## [3] "Harford \n & The Alameda"
```

```r
#Finding something that doesn't appear
grep("JeffStreet", cameraData$intersection)
```

```
## integer(0)
```

```r
length(grep("JeffStreet", cameraData$intersection))
```

```
## [1] 0
```

- Other useful string functions

```r
library(stringr)
nchar("Jeffrey Leek") #Num of characters
```

```
## [1] 12
```

```r
substr("Jeffrey Leek", 1,7) #Subset part of string (1st to 7th letters)
```

```
## [1] "Jeffrey"
```

```r
paste("Jeffrey", "Leek")#I already use this all the time
```

```
## [1] "Jeffrey Leek"
```

```r
#BUT I ALWAYS WASTE MY TIME TYPING: `sep = ""`
paste0("Jeffrey", "Leek")
```

```
## [1] "JeffreyLeek"
```

```r
#Trim out spaces
str_trim("Jeff          ")
```

```
## [1] "Jeff"
```

Important points about text in data sets

- Names of variables should be:
  - All lower case when possible
  - Descriptive (Diagnosis versus Dx)
  - Not duplicated
  - Not have underscores, dots, or white spaces
- Variables with character values:
  - Should usually be made into factor variables (depends on application)
  - Should be descriptive

    * use `TRUE`/`FALSE` instead of `0`/`1` and `Male`/`Female` versus either `0`/`1` or `M`/`F`

Regular Expressions 1

- Regular expressions can be thought of as a combination fo literals and metacharacters

- An analogy with natural language: think of literal text forming the words of this language, and the metacharacters as defining its grammer

- Regular expressions have a rich set of metacharacters

- Simplest pattern consists only of literals, such as a particular word; a match occurs if the sewuence of literals occurs anywhere in the text being tested
- What if we only want the word "Obama"? Or sentences that end in the word "Clinton", "clinton", or "clinto"?

- We need a way to express:
  - whitespace word boundaries
  - sets of literals
  - the beginning and end of a line
  - alternatives ("war" or"peace") This is where we get the aid of…

Metacharacters

- Some meta characters (`^`) represent the start of a line:
  - `^i think` will match with the lines:
    * *i think we all rule for participating*
    * *i think i have been outed*
    * *i think this will be quite fun actually*
    * *i think i need to go to work*
    * *i think i first saw zombo in 1999*
  - However, it will not match if *i think* appears in the middle of the line
- `$` represents the end of a line
  - `morning$` will match with the lines:
    * *well they had somethin this morning*

    - *then had to catch a tram home in the morning*

    - *dog obedience school in the morning*

    - *and yes happy birthday I forgot to say it earlier this morning*

    - *I walked in the rain this morning*

    - *good morning*
- We can list a set of characters we will accept at a given point in the match
  - `[Bb] [Uu] [Ss] [Hh]` will match with the lines: (Any verison of the word *bush*)
    - The democrats are playing, "Name the worst thing about **Bush**!"

    - I smelled the desert creosote **bush**, brownies, BBQ chicken

    - BBQ and **bush**walking at Molonglo Gorge

    - **Bush** TOLD you that North Korea is part of the Axis of Evil

    - I'm listening to **Bush** - Hurricane (Album Version)
- Combing these features
  - `^[Ii] am` will match with:
    - **i am** so angry at my boyfriend I can't even bear to look at him

    - **i am** boycotting the apple sotre

    - **I am** twittering from iPhone

    - **I am** a very vengeful person when you ruin my sweetheart.

    - **I am** so over this. I need food. Mmmm bacon...
- You can also specify a range of letters [a-z] or [a-zA-Z] for upper or lower case
  - notice the order doesn't matter

  - So `^[0-9] [a-zA-Z]` will match with:
    - **7t**h inning stretch

    - **2n**d half soon to begin. OSU did just win something

    - **3a**m - can't sleep - too hot still.. :(

    - **5f**t 7 send from heaven

    - **1s**t sign of starvation
- When used at the beginning of a character class, the `^` is also a metacharacter and indicates matching chracters NOT in the indicated class
  - `[^?.]$` will match any lines that do NOT end in a `?` or `.` such as:
    - I like basketballs

    * 6 and 9

    * don't worry… we all die anyway!

    * Not in Baghdad

    * helicopter under water? hmmm

Regular Expressions 2: More Metacharacters

- The `.` is used to refer to any character.
  - So `9.11` will match anything with a `9`, any character, then an `11`:
    * it's stupid the post **9-11** rules

    * if any 1 of us did **9/11** we would have been caught in days.

    * NetBios: scanning ip 203.16**9.11**4.66

    * Front Door **9:11**:46 AM

    * Sings: 0118999881**9911**9725…3 !
- The `|` is used like an "or" operator and can be used to combine two expressions, the subexpressions are called *alternatives*
  - So `flood|fire` will match with any line that contains `flood` or `fire`:
    * is **fire**wire like usb on none macs?

    * the global **flood** makes sense within the context of the bible

    * yeah I've ahd the **fire** on tongiht

    * … and the **flood**s, hurricanes, killer heatwaves, readnecks, gun nuts, etc.

  - Multiple characters can also be put in one line `flood|earthquake|hurricane|coldfire`
- The alternatives can be real expressions and not just literals
  - `^[Gg]ood|[Bb]ad` will match with
    * **good** to hear some good news from someone here

    * **Good** afternoon fellow american infidels!

    * **good** on you- what do you drive?

    * Katie… guess they had **bad** experiences…

    * my middle name is trouble, Miss **Bad** News
- As such paratheses should be used if one wish to extend the `^`
  - `^([Gg]ood|[Bb]ad)` will match:

        - **bad** habbit

        - **bad** coordination today

        - **good**, because there is nothing worse...

        - **Bad**cop, it's because people want to use drugs

        - **Good** Monday Holiday

        - **Good** riddance to Limey
- The `?` (Question mark) indicates that the indicated expression is optional
    - So `[Gg]eorge( [Ww]\.)? [Bb]ush` will match the lines:
        - I bet I can spell better than you and **george bush** combined

        - BBC reported that President **George W. Bush** claimed God told him to invade..
        - a bird in the hand is worth two **george bush**es
- The `*` and `+` signs are metacharacters used to indicate repetition;
    - `*` means "any number, including none, of the item"
    - `+` means "at least one of the item"

    - So `(.*)` is searching for some phrase inbetween paratheses
    - And `[0-9]+ (.*)[0-9]+` will look for any combination of numbers that are separated by something
- `{ and }` are refferred to as interval quantifiers; they let us specify the minimum and mzimum number of matches of an expression
    - So `[Bb]ush( +[^ ]+ +){1,5} debate` will match any line that has *bush* followed by *1 to 5* words then the word *debate*
- Numbers
    - m,n means at least m but not more than n matches

    - *m* means exactly m matches

    - *m,* means at least m matches
- In most implementations of regualr expressions, the parentheses not only limit the scope of alternatives divided by a "|", byt also can eb used to "remember" text matched by the subexpression enclsoed
    - We refer to the matched text with `\1`, `\2`, etc. (Escaped numbers)
    - So `+( [a-zA-Z]+) +\1 +` is looking for *a space* + *some number of, but at least 1, characters* + *a space* + *The same set of characters previously seen* + *a space*; The following lines will match
        - time for bed, **night night** twitter!

        - blah **blah blah** blah

        - my tattoo is **so so** itchy today

        - I was standing **all all** alone against the world outside...

    * hi **anybody anybody** at home

    * estudiando **css css** css css… que desastritooooo
- The `*` is "greedy" so it always matches the *longest* possible string that satisfies the regular expression
  - So `^s(.*)s` matches with:
    * **sitting at starbucks**

    * **setting up mysql and rails**

    * **studying s**tuff for the exam

    * **stop fighting with crackers**

    * **sore shoulders are s**tupid
  - The "greediness" of `*` can be turned off with the `?` as in: `^s(.*?)s$` which will match with:
    * **sitting at s**tarbucks

    * **setting up mys**ql and rails

    * **studying s**tuff for the exam

    * **stop fighting with crackers**

    * **sore s**houlders are stupid
- Summary
  - Regular expressions are used in many different languages; not unique to R

  - Regular expressions are composed of literals and metacharacters that represent sets or classes of characters/words

  - Text processing via regular expressions is a very powerful way to extract data from "unifriendly" sources

  - Used with the functions `grep`, `grepl`, `sub`, `gsub` and others that involve searching for text strings

Working with Dates

- Starting simple

```
d1 <- date()
d1
```

```
## [1] "Sat Mar 28 20:46:35 2020"
```

```r
class(d1)
```

```
## [1] "character"
```

- Date class

```r
d2 <- Sys.Date()
d2
```

```
## [1] "2020-03-28"
```

```r
class(d2)
```

```
## [1] "Date"
```

Formatting dates

- `%d` = day as number (0-31)
- `%a` = abbreviated weekday
- `%A` = unabbreviated weekday
- `%m` = month (00-12)
- `%b` = abbreviated month (Jan, Feb, etc.)
- `%B` = unabbreviated month (January, Febuary, etc.)
- `%y` = two digit year
- `%Y` = four digit year

```r
d2 <- Sys.Date()
format(d2, "%a %b %d")
```

```
## [1] "Sat Mar 28"
```

Creating dates with `as.Date` function

```r
x <- c("1jan1960", "2jan1960", "31mar1960", "30jul1960")
z <- as.Date(x, "%d%b%Y")
z
```

```
## [1] "1960-01-01" "1960-01-02" "1960-03-31" "1960-07-30"
```

```r
#Manipulating these dates
z[1] - z[2]
```

```
## Time difference of -1 days
```

```r
as.numeric(z[1]-z[2])
```

```
## [1] -1
```

- Converting to Julian

```r
d2 <- Sys.Date()
weekdays(d2, abbreviate = FALSE)
```

```
## [1] "Saturday"
```

```r
months(d2, abbreviate = FALSE)
```

```
## [1] "March"
```

```r
#Reports number of days since an orgin
julian(d2)
```

```
## [1] 18349
## attr(,"origin")
## [1] "1970-01-01"
```

Lubridate

- Another Hadley Wickham package, **Read more about it here**
- Easily converts common standard formats of dates into `Date` class

```r
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following object is masked from 'package:plyr':
##
##     here

## The following objects are masked from 'package:data.table':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year

## The following object is masked from 'package:base':
##
##     date
```

```r
ymd("20140108")
```

```
## [1] "2014-01-08"
```

```r
mdy("08/04/2013")
```

```
## [1] "2013-08-04"
```

```r
dmy("03-04-2013")
```

```
## [1] "2013-04-03"
```

- Also allows one to deal with times

```r
ymd_hms("2011-08-03 10:15:03")
```

```
## [1] "2011-08-03 10:15:03 UTC"
```

```r
#Including timezones
ymd_hms("2011-08-03 10:15:03", tz = "Pacific/Auckland")
```

```
## [1] "2011-08-03 10:15:03 NZST"
```

```r
#Finding your System's tz (kinda)
Sys.timezone()
```

```
## [1] "America/New_York"
```

- Some functions in `lubridate` have a slightly different syntax

```r
x <- dmy(c("1jan2013", "2jan2013", "31mar2013", "30jul2013"))

#Returns a numeric by default
wday(x[1])
```

```
## [1] 3
```

```r
#Returns a "ordered" with a "factor"
wday(x[1], label=TRUE)
```

```
## [1] Tue
## Levels: Sun < Mon < Tue < Wed < Thu < Fri < Sat
```

- Ultimately you want your date and times as class "`Date`" or the classes "`POSIXct`" or "`POSIXlt`"

Data Resources (Where to find cattle)

Open Government Sites

- **United Nations**

- **U. S.**

- **The Nethelands**
- **United Kingsom**

- **France**

- **Ghana**

- **Australia**
- **Germany**
- **Hong Kong**
- **Japan**
- ***Many more***

Other Sites

- **Gapminder** - Development in human health
- **Survey data from the United States** - Info on how to access the surveys
- **Infochimps Marketplace** - Some are free, some cost money
- **Kaggle** - Company that offers data science competitions

Collections by data scientists

- **Hilary Mason (Dead Link, website provided instead)**
- **Peter Skomoroch**
- **Jeff Hammerbacher**
- **Gregory Piatetsky-Shapiro**
- **Many more**

More specialized collections

- **Stanford Large Network Data**
- **UCI Machine Learning**
- **KDD Nugets Datasets**
- **CMU Statlib**
- **Gene expression omnibus**
- **ArXiv Data**
- **Public Data Sets on Amazon Web Services**

Some API's with R interfaces

- **twitter** and **twitteR** package
- **figshare** and [rfigshare]**https://cran.r-project.org/web/packages/rfigshare/index.html)**
- **PLoS** and **rplos**
- **rOpenSci**
- **Facebook** and **RFacebook**
- **Google maps** and **RGoogleMaps**

Lessons with `swirl()`

Tidying Data with tidyr

- This lesson just covered the `lubridate` package

```
#Sometimes one needs to be more specific in a call
ymd("192012")
```

```
## Warning: All formats failed to parse. No formats found.
```

```
## [1] NA
```

```
ymd("1920-1-2")
```

```
## [1] "1920-01-02"
```

- **Complete list of valid time zones for use with lubridate**
- The `with_tz()` function returns a date-time as it would appear in another time zone

Me looking at some data on my own

```
saveLoc <- paste0(getwd(), "/data/debate_transcripts_v3_2020-02-26.csv")
trans <- read.csv(saveLoc)
library(dplyr)
trans <- mutate(trans, word_count = sapply(sapply(as.character(trans$speech), strsplit, split =
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale

## Warning in FUN(X[[i]], ...): input string 1 is invalid in this locale
```

```
cutoffIndices <- grep("-$", trans$speech)
```

```
## Warning in grep("-$", trans$speech): input string 5 is invalid in this locale

## Warning in grep("-$", trans$speech): input string 6 is invalid in this locale

## Warning in grep("-$", trans$speech): input string 8 is invalid in this locale

## Warning in grep("-$", trans$speech): input string 9 is invalid in this locale

## Warning in grep("-$", trans$speech): input string 10 is invalid in this locale
```

```
trans <- mutate(trans, got_cutoff = FALSE, cutoff_opponet = FALSE)
trans$got_cutoff[cutoffIndices] <- TRUE
trans$cutoff_opponet[cutoffIndices+1] <- TRUE
trans <- group_by(trans, debate_name)
names(trans)
```

```
## [1] "date"                   "debate_name"            "debate_section"
## [4] "speaker"                "speech"                 "speaking_time_seconds"
## [7] "word_count"             "got_cutoff"             "cutoff_opponet"
```

```
bernie <- trans %>% filter(as.character(speaker) == "Bernie Sanders")
biden <- trans %>% filter(as.character(speaker) == "Joe Biden")
bernie_summary <- summarise(bernie,
            speaking_time = sum(speaking_time_seconds), word_count = sum(word_count),
            avg_WPM = word_count/(speaking_time/60),
            got_cutoff = sum(got_cutoff), cutoff_opponet = sum(cutoff_opponet))

biden_summary <- summarise(biden,
            speaking_time = sum(speaking_time_seconds), word_count = sum(word_count),
```

```
            avg_WPM = word_count/(speaking_time/60),
            got_cutoff = sum(got_cutoff), cutoff_opponet=sum(cutoff_opponet))

bernie_summary
```

```
## # A tibble: 10 x 6
##    debate_name         speaking_time word_count avg_WPM got_cutoff cutoff_opponet
##    <fct>                       <dbl>      <int>   <dbl>      <int>          <int>
##  1 "December Democra~           1236        499    24.2          1              2
##  2 "Democratic Debat~            943        456    29.0          3              3
##  3 "January Iowa Dem~           1049        849    48.6          1              0
##  4 "New Hampshire De~           1209        807    40.0          0              2
##  5 "November Democra~            705        225    19.1          0              1
##  6 "October Democrat~            795        348    26.3          2              6
##  7 "September Housto~            847        639    45.3          1              0
##  8 "South Carolina D~            911        734    48.3         12              7
##  9 "Transcript from ~            648        802    74.3          6              2
## 10 "Transcript of Ju~           1046        626    35.9          4              2
```

```
biden_summary
```

```
## # A tibble: 10 x 6
##    debate_name         speaking_time word_count avg_WPM got_cutoff cutoff_opponet
##    <fct>                       <dbl>      <int>   <dbl>      <int>          <int>
##  1 "December Democra~            774         46    3.57          0              0
##  2 "Democratic Debat~            790        201   15.3           1              3
##  3 "January Iowa Dem~            978        111    6.81          0              0
##  4 "New Hampshire De~           1188         59    2.98          1              2
##  5 "November Democra~            763         85    6.68          0              0
##  6 "October Democrat~            991        281   17.0           2              3
##  7 "September Housto~           1069        210   11.8           1              1
##  8 "South Carolina D~            765        153   12             7              6
##  9 "Transcript from ~            799        517   38.8           0              1
## 10 "Transcript of Ju~           1193        305   15.3           3              1
```

Quiz Scribbles

1)

- **The American Community Survey distributes downloadable data about United States communities. Download the 2006 microdata survey about housing for the state of Idaho**

```
url <- "https://d396qusza40orc.cloudfront.net/getdata%2Fdata%2Fss06hid.csv"
saveLoc <- paste0(getwd(), "/data/Q4IdahoHousing06.csv")
download.file(url, saveLoc, "curl")
idaho <- read.csv(saveLoc)
```

- **Apply strsplit() to split all the names of the data frame on the characters "wgtp".**

What is the value of the 123 element of the resulting list?

```
strsplit(names(idaho), "wgtp")[123]
```

```
## [[1]]
## [1] ""   "15"
```

2)

- **Load the Gross Domestic Product data for the 190 ranked countries**

```
url <- "https://d396qusza40orc.cloudfront.net/getdata%2Fdata%2FGDP.csv"
saveLoc <- paste0(getwd(), "/data/Q4GDP.csv")
download.file(url, saveLoc, "curl")
rawGDP <- read.csv(saveLoc)
```

- **Remove the commas from the GDP numbers in millions of dollars and average them. What is the average?**

```
gdp <- rawGDP[5:length(rawGDP[,1]),]
gdp <- gdp %>% rename(GDP_Mil_USD = X.3,
                LongCountryName = X.2, ShortCountryName = X, Rank = Gross.domestic.product.2012
            select(ShortCountryName, Rank, GDP_Mil_USD, LongCountryName) %>%
            mutate(GDP_Mil_USD = as.numeric(gsub(",", "", GDP_Mil_USD))) %>%
            filter(as.numeric(as.character(Rank)) > 0)
```

```
## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion
```

```
mean(gdp$GDP_Mil_USD)
```

```
## [1] 377652.4
```

3)

- **In the data set from Question 2 what is a regular expression that would allow you to count the number of countries whose name begins with "United"? Assume that the variable with the country names in it is named countryNames. How many countries begin with United?**

```
countryNames <- as.character(gdp$LongCountryName)
length(grep("^United", countryNames))
```

```
## Warning in grep("^United", countryNames): input string 99 is invalid in this
## locale

## Warning in grep("^United", countryNames): input string 186 is invalid in this
## locale

## [1] 3
```

4)

- **Load the Gross Domestic Product data for the 190 ranked countries** (Same as Question 2)

```r
saveLoc <- paste0(getwd(), "/data/Q4GDP.csv")
rawGDP <- read.csv(saveLoc)
```

- **Load the educational data**

```r
url <- "https://d396qusza40orc.cloudfront.net/getdata%2Fdata%2FEDSTATS_Country.csv"
saveLoc <- paste0(getwd(), "/data/Q4Edu.csv")
download.file(url, saveLoc, "curl")
rawEdu <- read.csv(saveLoc)
```

- **Match the data based on the country shortcode. Of the countries for which the end of the fiscal year is available, how many end in June?**

```r
# Clean gdp
gdp <- rawGDP[5:length(rawGDP[,1]),]
gdp <- gdp %>% rename(GDP_Mil_USD = X.3,
                Long.Name = X.2, CountryCode = X, Rank = Gross.domestic.product.2012) %>%
            select(CountryCode, Rank, GDP_Mil_USD, Long.Name) %>%
            mutate(GDP_Mil_USD = as.numeric(gsub(",", "", GDP_Mil_USD))) %>%
            filter(!is.na(CountryCode))
```

```
## Warning: NAs introduced by coercion
```

```r
# Merge
combinedData <- merge(gdp, rawEdu, by = "CountryCode", all = FALSE)
combinedData <- rename(combinedData, Long.Name = Long.Name.y)



#Only need end of the fiscal year in june info
condencedData <- combinedData %>% select(Rank, CountryCode, Long.Name, Special.Notes, GDP_Mil_U
qualifyingData <- condencedData[grep("^Fiscal year end: June", condencedData$Special.Notes),]
qualifyingData
```

```
##     Rank CountryCode                         Long.Name
## 11    12         AUS       Commonwealth of Australia
## 18    59         BGD People's Republic of Bangladesh
## 31   117         BWA             Republic of Botswana
## 58    38         EGY           Arab Republic of Egypt
## 74   175         GMB           Republic of The Gambia
## 102   87         KEN                Republic of Kenya
## 109   56         KWT                  State of Kuwait
## 157   44         PAK    Islamic Republic of Pakistan
## 164   61         PRI                      Puerto Rico
## 179  157         SLE        Republic of Sierra Leone
## 189   21         SWE                Kingdom of Sweden
## 206  106         UGA               Republic of Uganda
## 224  134         ZWE             Republic of Zimbabwe
##                                                                Special.Notes
## 11  Fiscal year end: June 30; reporting period for national accounts data: FY.
## 18  Fiscal year end: June 30; reporting period for national accounts data: FY.
```

```
## 31  Fiscal year end: June 30; reporting period for national accounts data: FY.
## 58  Fiscal year end: June 30; reporting period for national accounts data: FY.
## 74  Fiscal year end: June 30; reporting period for national accounts data: CY.
## 102 Fiscal year end: June 30; reporting period for national accounts data: CY.
## 109 Fiscal year end: June 30; reporting period for national accounts data: CY.
## 157 Fiscal year end: June 30; reporting period for national accounts data: FY.
## 164 Fiscal year end: June 30; reporting period for national accounts data: FY.
## 179 Fiscal year end: June 30; reporting period for national accounts data: CY.
## 189 Fiscal year end: June 30; reporting period for national accounts data: CY.
## 206 Fiscal year end: June 30; reporting period for national accounts data: FY.
## 224 Fiscal year end: June 30; reporting period for national accounts data: CY.
##     GDP_Mil_USD
## 11      1532408
## 18       116355
## 31        14504
## 58       262832
## 74          917
## 102       40697
## 109      160913
## 157      225143
## 164      101496
## 179        3796
## 189      523806
## 206       19881
## 224        9802
```

5)

- **You can use the *quantmod package* to get historical stock prices for publicly traded companies on the NASDAQ and NYSE. Use the following code to download data on Amazon's stock price and get the times the data was sampled.**
  (Following code was given in the question)

```
library(quantmod)
```

```
## Loading required package: xts

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

##
## Attaching package: 'xts'

## The following objects are masked from 'package:dplyr':
##
##     first, last
```

```
## The following objects are masked from 'package:data.table':
##
##     first, last

## Loading required package: TTR

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

## Version 0.4-0 included new data defaults. See ?getSymbols.

##
## Attaching package: 'quantmod'

## The following object is masked from 'package:Hmisc':
##
##     Lag
```

```r
amzn = getSymbols("AMZN",auto.assign=FALSE)
```

```
## 'getSymbols' currently uses auto.assign=TRUE by default, but will
## use auto.assign=FALSE in 0.5-0. You will still be able to use
## 'loadSymbols' to automatically load data. getOption("getSymbols.env")
## and getOption("getSymbols.auto.assign") will still be checked for
## alternate defaults.
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.warning4.0"=FALSE). See ?getSymbols for details.
```

```r
sampleTimes = index(amzn)
```

- **How many values were collected in 2012?**

```r
library(lubridate)
qualify <- (sampleTimes >= ymd("2012-01-01") & sampleTimes < ymd("2013-01-01"))
qualifyingDates <- sampleTimes[qualify]
length(qualifyingDates)
```

```
## [1] 250
```

- **How many values were collected on Mondays in 2012?**

```r
sum(wday(qualifyingDates)==2)
```

```
## [1] 47
```