

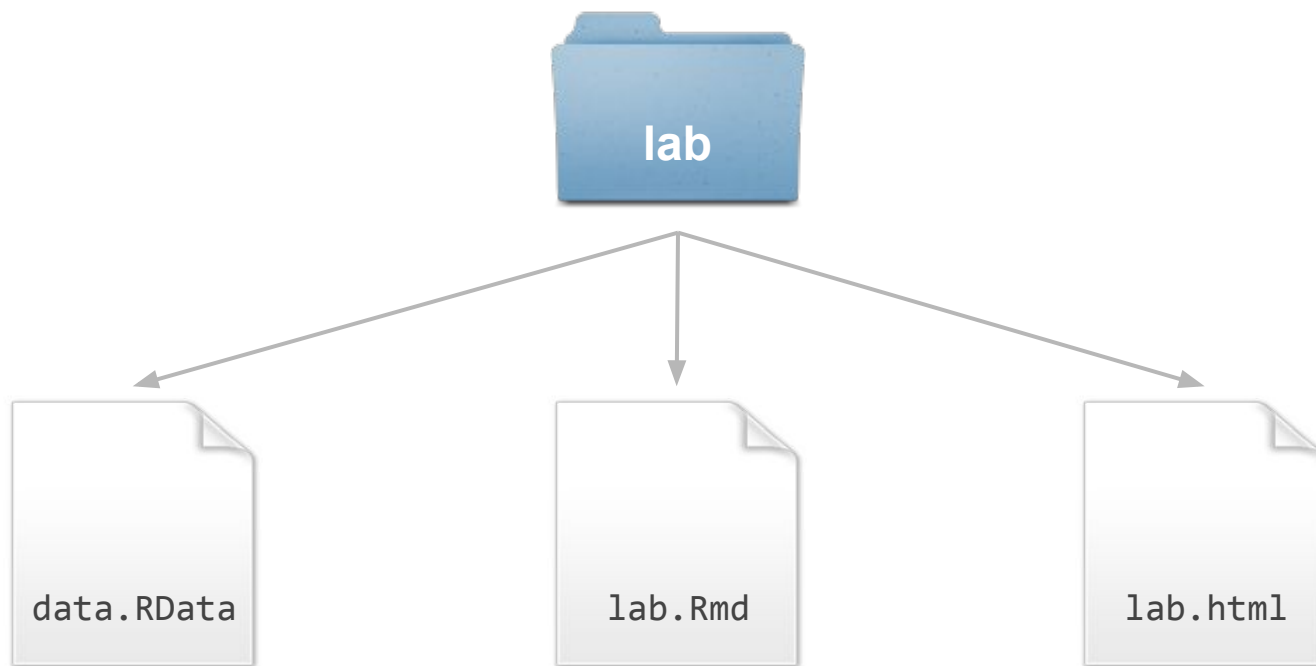
Filesystem

Stat 133 by Gaston Sanchez

Creative Commons Attribution Share-Alike 4.0 International CC BY-SA

How does a data
analysis project look like
from the files standpoint?

Example of files in a lab session

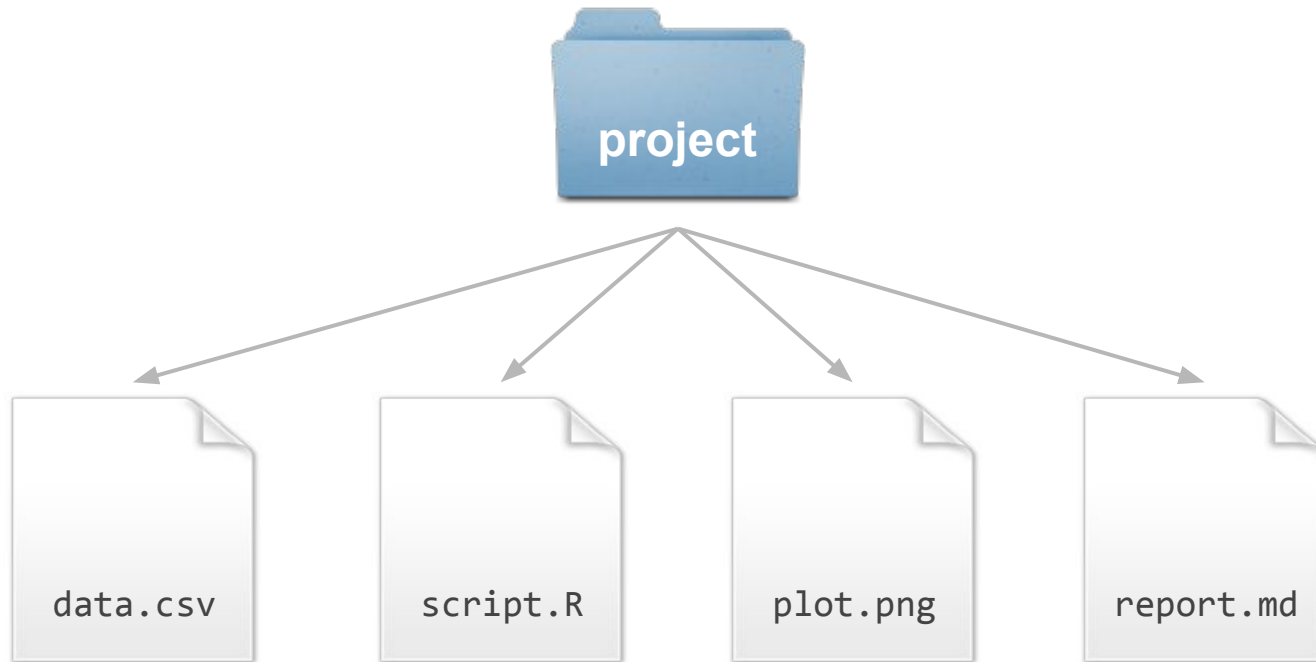


Relationships among files

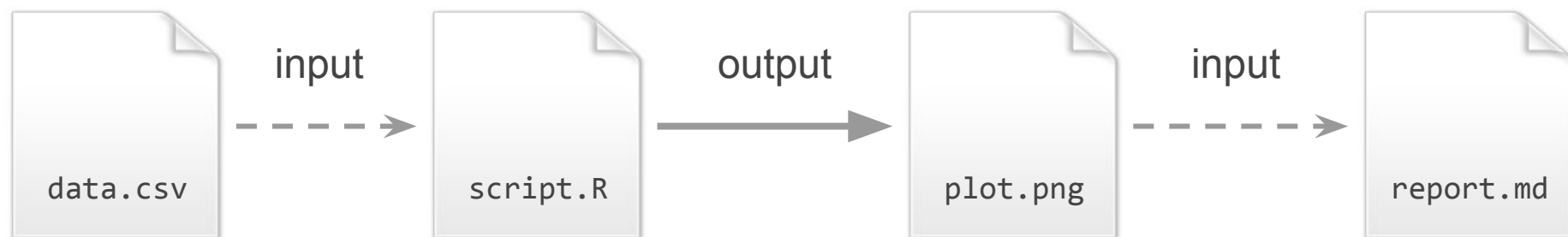


A Toy Project

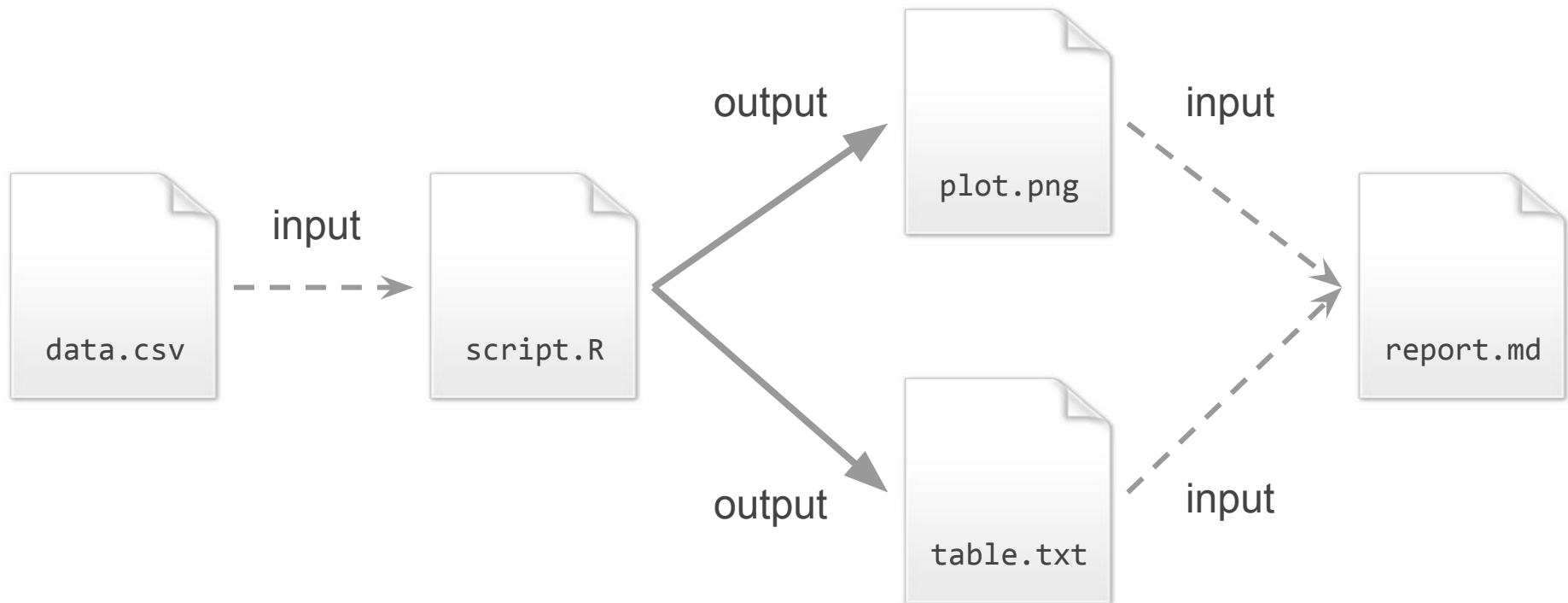
Basic Project example



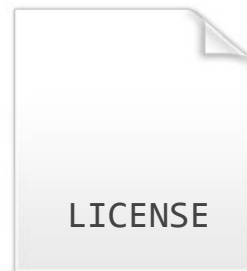
Relationships among files



Another example



A More Complete Project



Project File-Structure

project/

Project File-Structure

```
project/  
  data/  
    rawdata.csv  
    cleandata.csv
```

Project File-Structure

```
project/  
  data/  
    rawdata.csv  
    cleandata.csv  
  code/  
    functions/  
      clean-data.R  
      plots.R  
    scripts/  
      cleaning.R  
      analysis.R
```

Project File-Structure

```
project/  
  data/  
    rawdata.csv  
    cleandata.csv  
  code/  
    functions/  
      clean-data.R  
      plots.R  
    scripts/  
      cleaning.R  
      analysis.R  
  images/  
    plot1.png  
    plot2.png
```

Project File-Structure

```
project/  
  data/  
    rawdata.csv  
    cleandata.csv  
  code/  
    functions/  
      clean-data.R  
      plots.R  
    scripts/  
      cleaning.R  
      analysis.R  
  images/  
    plot1.png  
    plot2.png  
  report/  
    doc.tex  
    doc.pdf  
  README.md
```

*THIS IS THE TYPICAL VIEW WE
HAVE OF A PROJECT'S STRUCTURE*

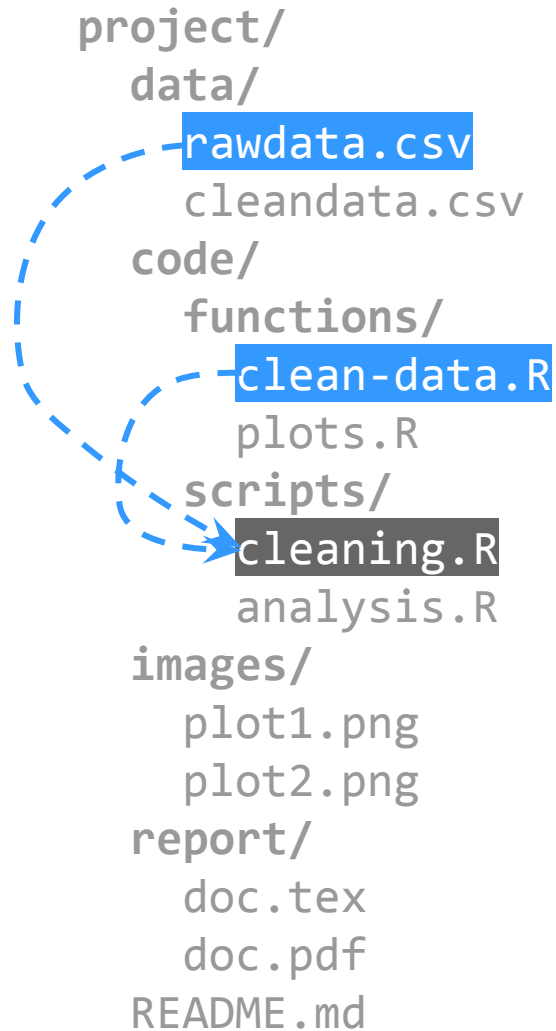
*BUT THERE'S ANOTHER WAY TO
LOOK AT A PROJECT'S FILES*

There's a network of
relationships among files

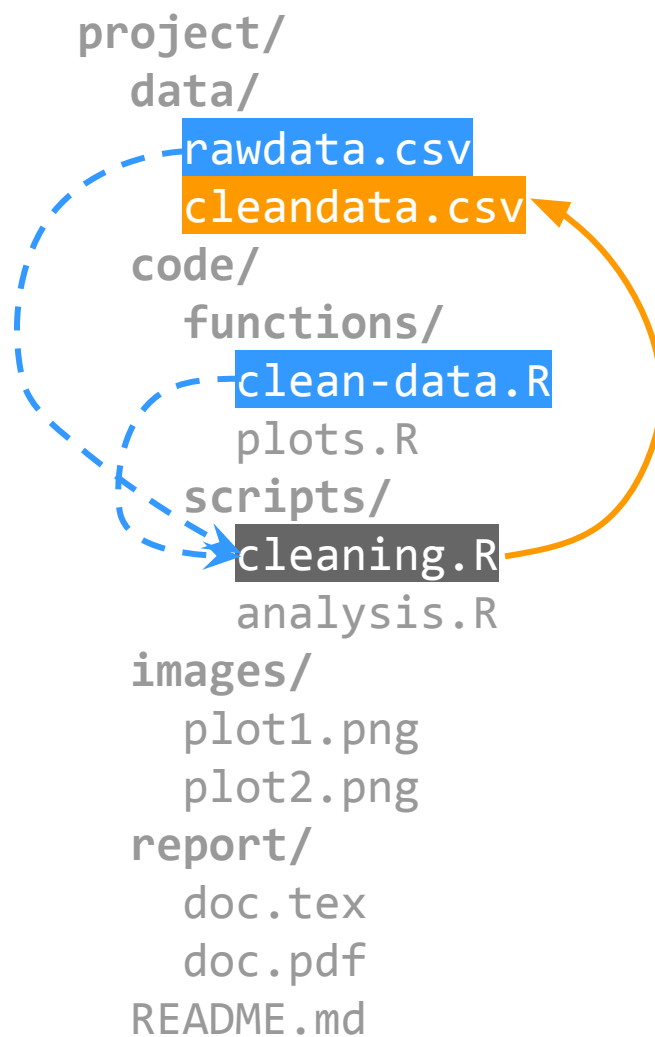
Project File-Structure

```
project/  
  data/  
    rawdata.csv  
    cleandata.csv  
  code/  
    functions/  
      clean-data.R  
      plots.R  
    scripts/  
      cleaning.R  
      analysis.R  
  images/  
    plot1.png  
    plot2.png  
  report/  
    doc.tex  
    doc.pdf  
  README.md
```

Project File-Structure



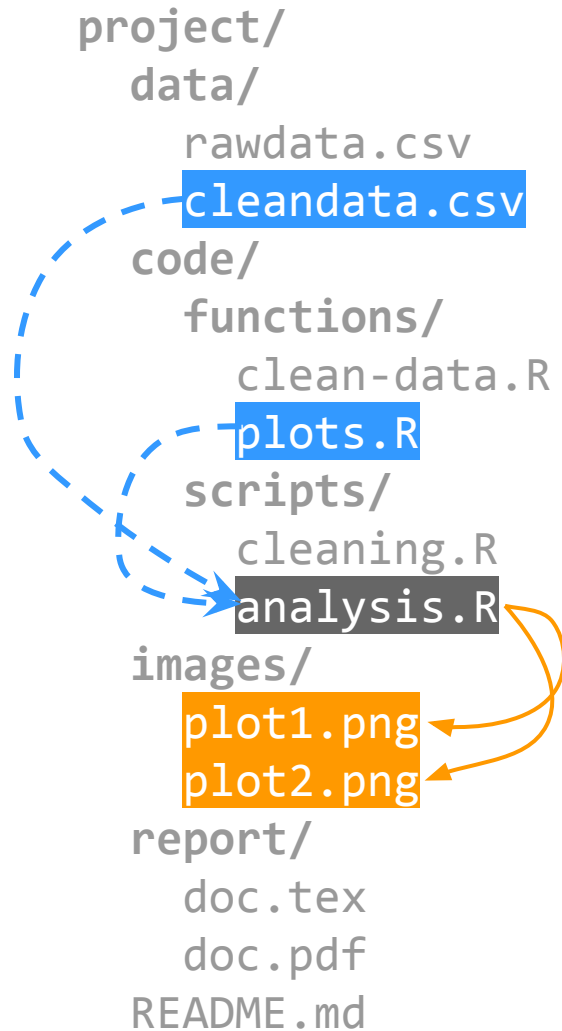
Project File-Structure



Project File-Structure

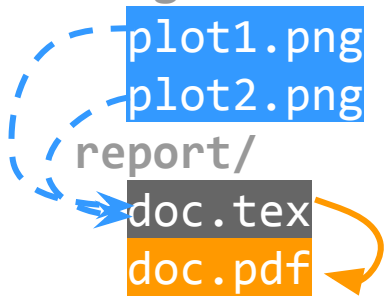


Project File-Structure



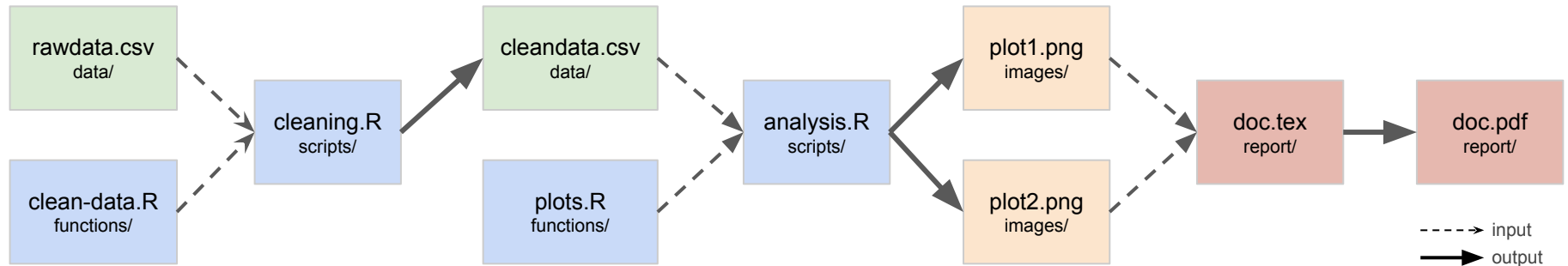
Project File-Structure

```
project/  
  data/  
    rawdata.csv  
    cleandata.csv  
  code/  
    functions/  
      clean-data.R  
      plots.R  
    scripts/  
      cleaning.R  
      analysis.R  
  images/  
    plot1.png  
    plot2.png  
  report/  
    doc.tex  
    doc.pdf  
  README.md
```



Drawing a flow diagram
showing the inputs and
outputs

Project Workflow



Files can be inputs
Files can be outputs
Or both

Project File-Structure

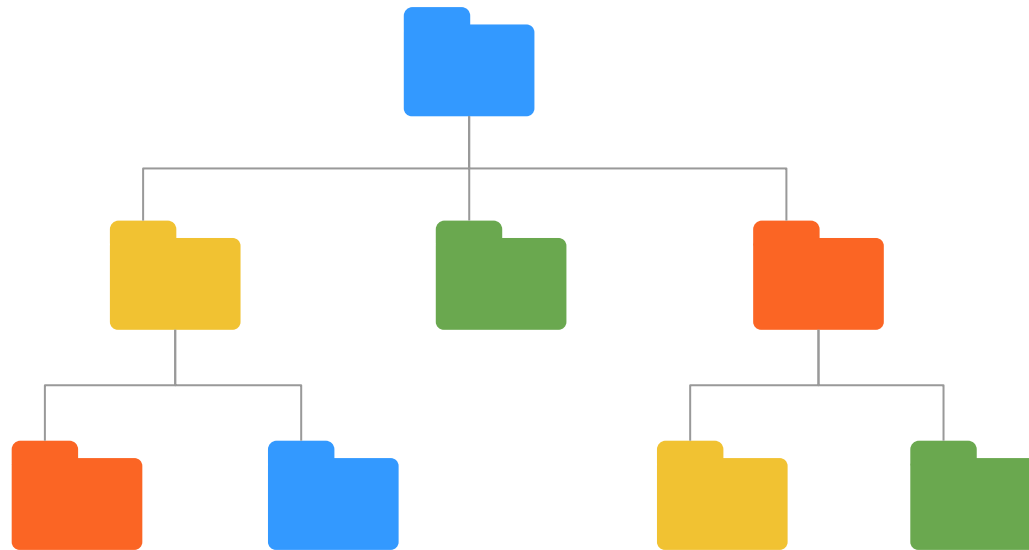
```
project/  
  data/  
    rawdata.csv  
    cleandata.csv  
  code/  
    functions/  
      clean-data.R  
      plots.R  
    scripts/  
      cleaning.R  
      analysis.R  
  images/  
    plot1.png  
    plot2.png  
  report/  
    doc.tex  
    doc.pdf  
  README.md
```

You'll be working with files
(some of them will be inputs,
some outputs, some both)



Lesson 1: Learn how to refer
to the files in a project

File System



Main Unix Concepts

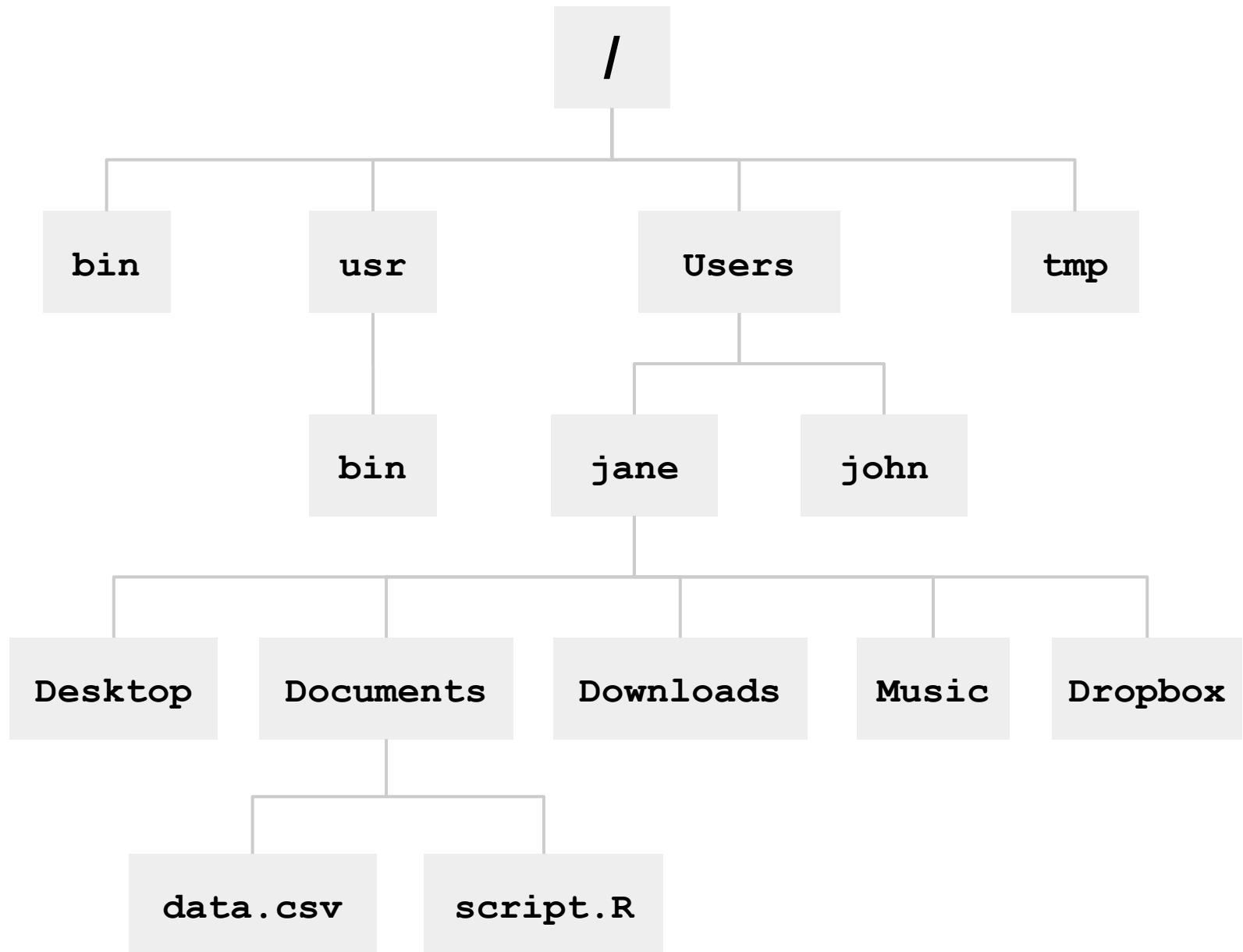
Everything is a **File** (including directories)

Files organized in a tree structure

The filesystem is a hierarchy (of folders & files)

Folder = Directory

No concept of “disk”

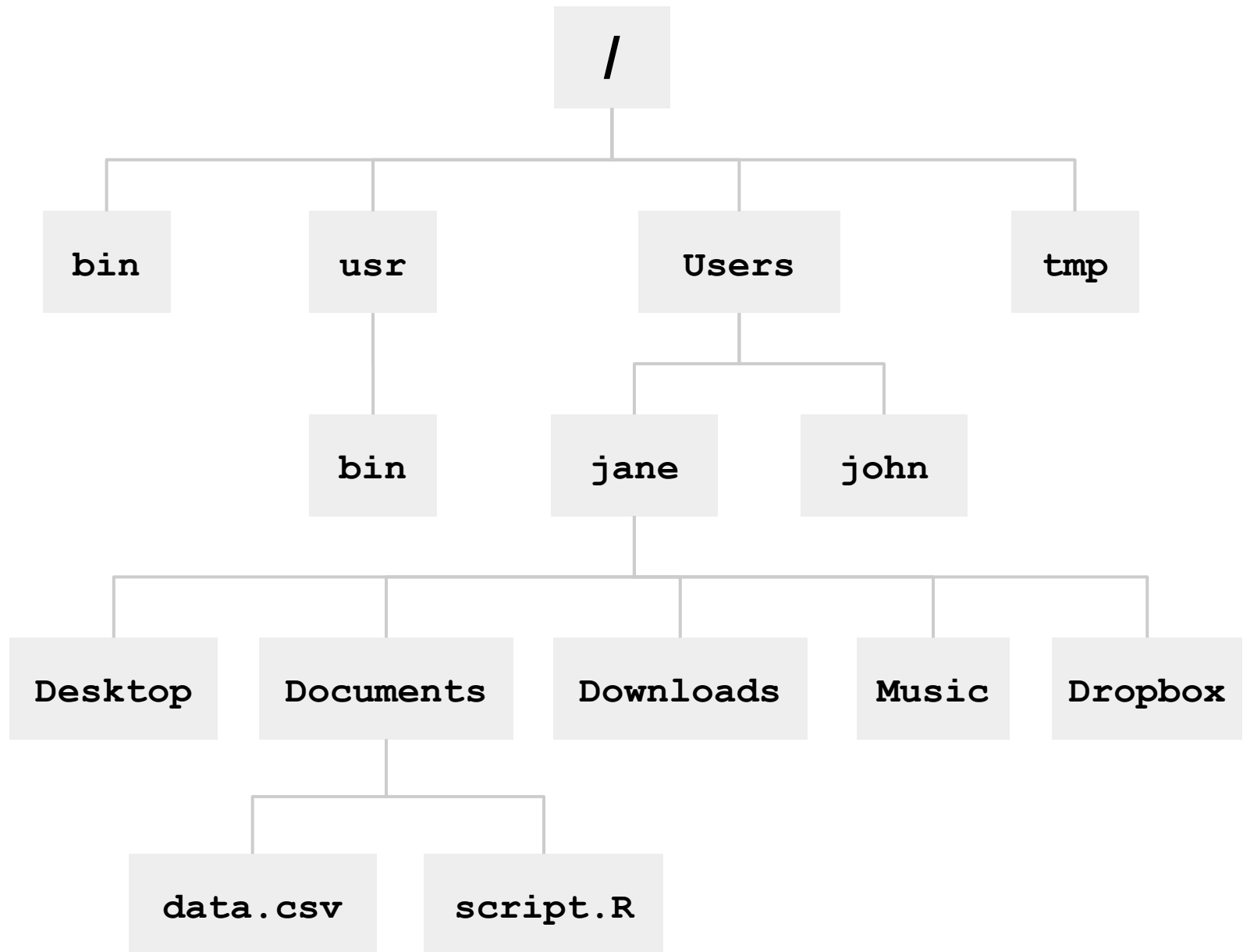


Typical filesystem in UNIX-like OS

Directory/Folder	Contents
<code>/</code>	Root
<code>/bin</code>	Binaries and programs (UNIX stuff)
<code>/sbin</code>	System binaries
<code>/dev</code>	Devices: hard drives, keyboard, mouse
<code>/etc</code>	System configurations
<code>/home</code>	User home directories (except on Mac)
<code>/lib</code>	Libraries of code
<code>/tmp</code>	Temporary files
<code>/var</code>	Various (files the system uses)
<code>/usr</code> <code>/usr/bin</code> <code>/usr/etc</code> <code>/usr/lib</code> <code>/usr/local</code>	User programs, tools and libraries

Mac-only files and directories

Directory/Folder	Contents
<code>/Applications</code>	Mac programs
<code>/Library</code>	Mac libraries of code
<code>/Network</code>	Networked devices
<code>/System</code>	Mac OS X
<code>/Users</code>	User home directories
<code>/Volumes</code>	Mounted volumes (hard drive, dvd, etc)
<code>.DS_Store</code>	Holds folder view options, icon positions
<code>~/ .MacOSX</code>	Directory for Mac OS X to store options
<code>~/ .Trash</code>	User trash can



Main Unix Concepts

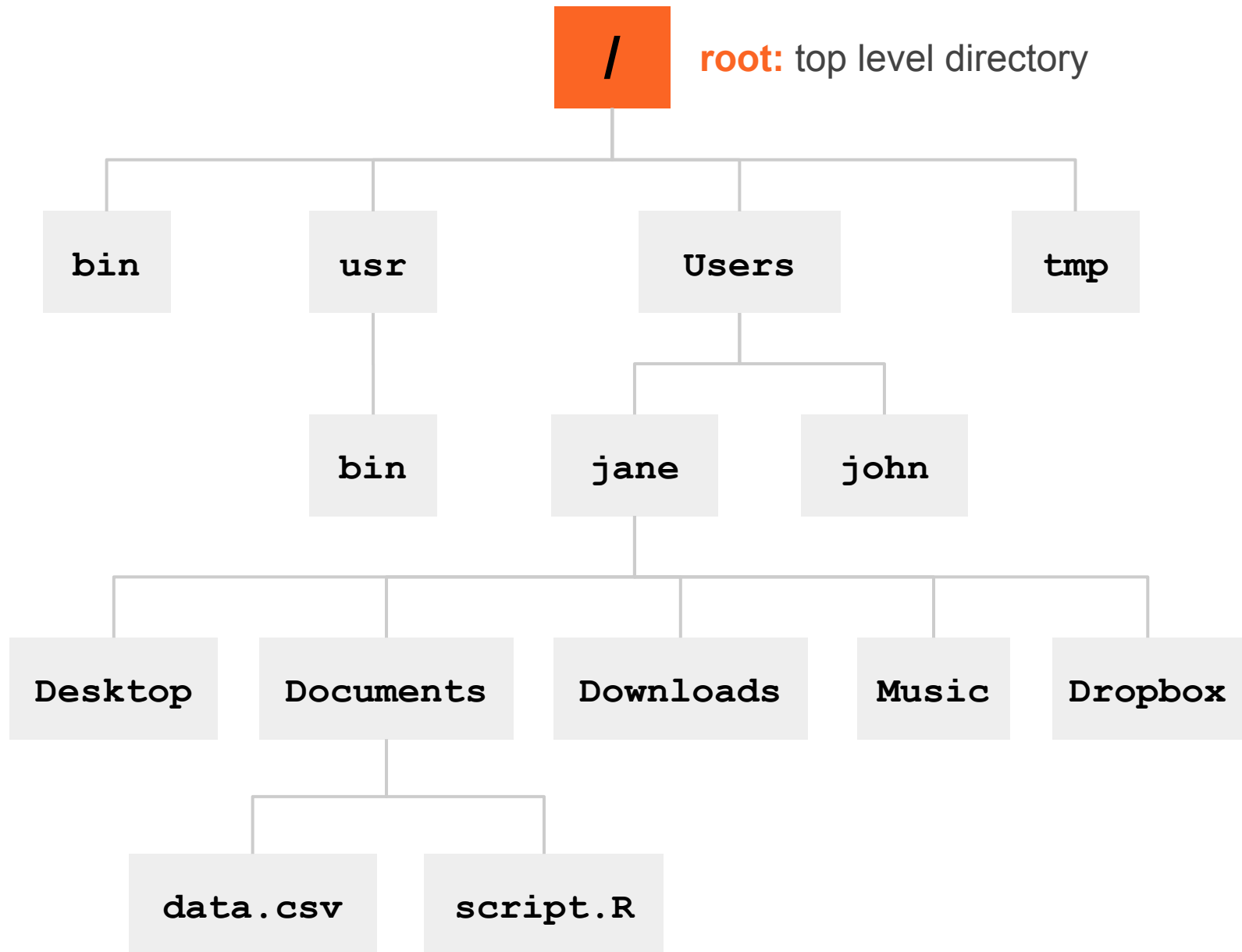
At any given time we are inside a directory

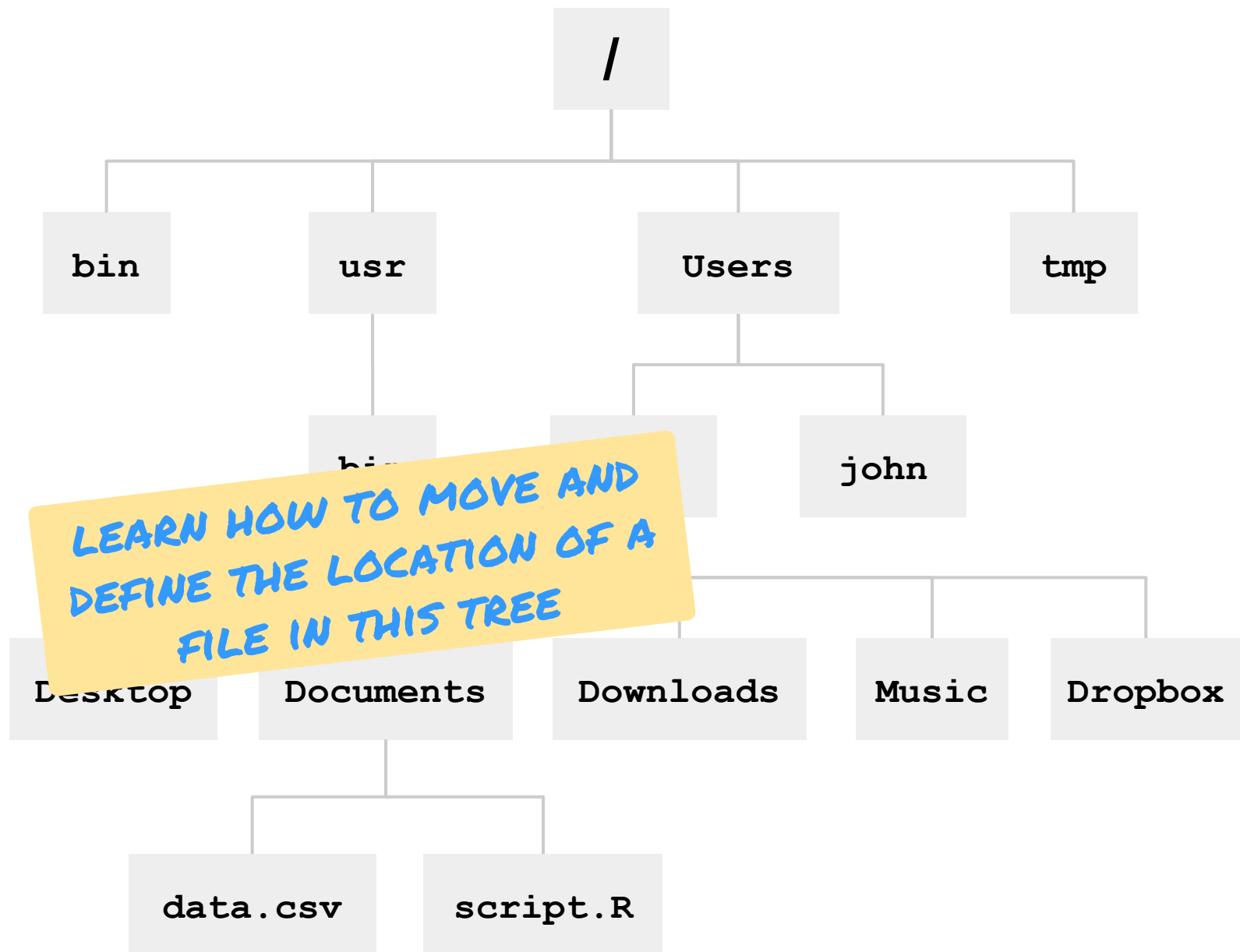
The current directory is the **working directory**

From that directory we can move up or down

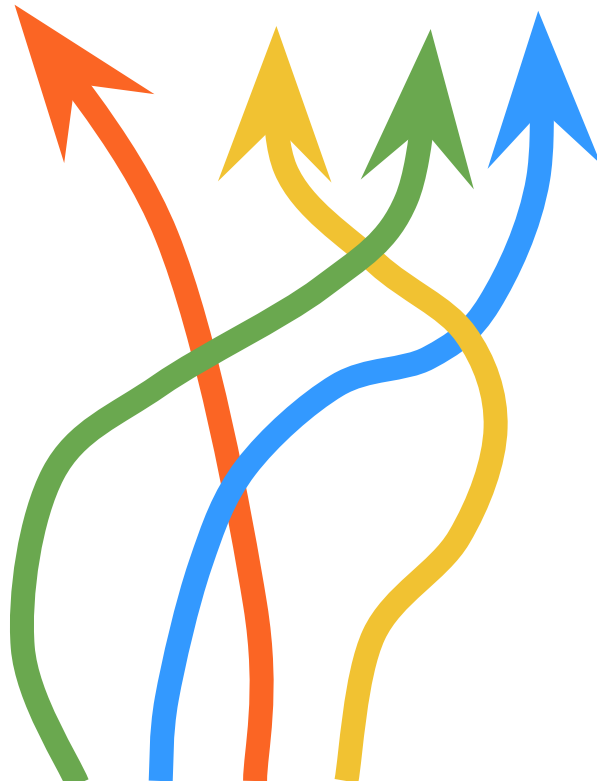
When a new **R session** is started, a working directory will be associated to the session

When a **terminal** is started the working directory is the **home directory**

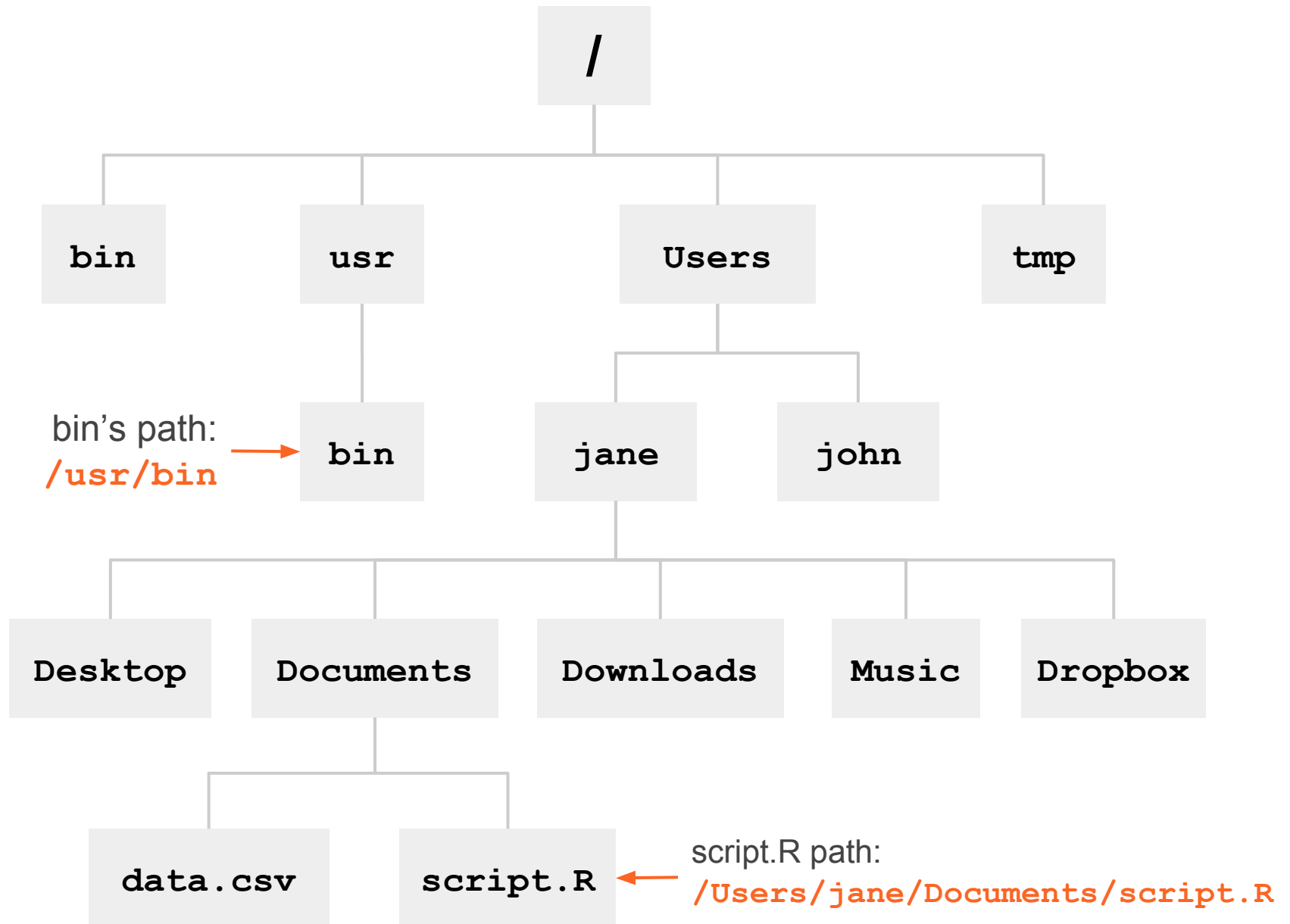


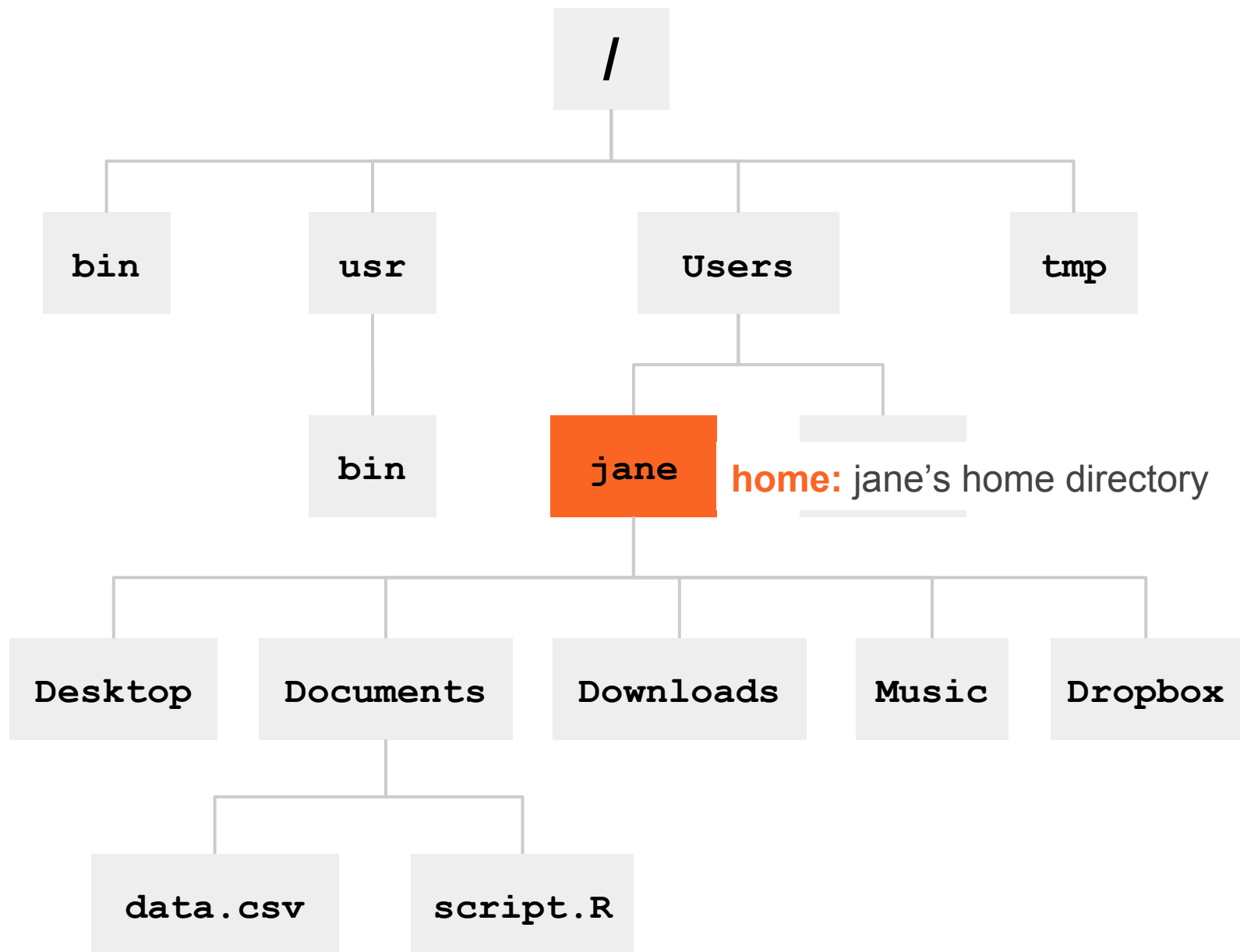


Paths



Each file and directory
has a unique name in the
filesystem called a **path**





Special Directories

/	root directory
~	home directory (i.e. <code>/home/user</code>)
.	current directory
..	parent directory

Absolute Paths

Absolute paths: an absolute pathname begins with the root directory and follows the tree branch by branch

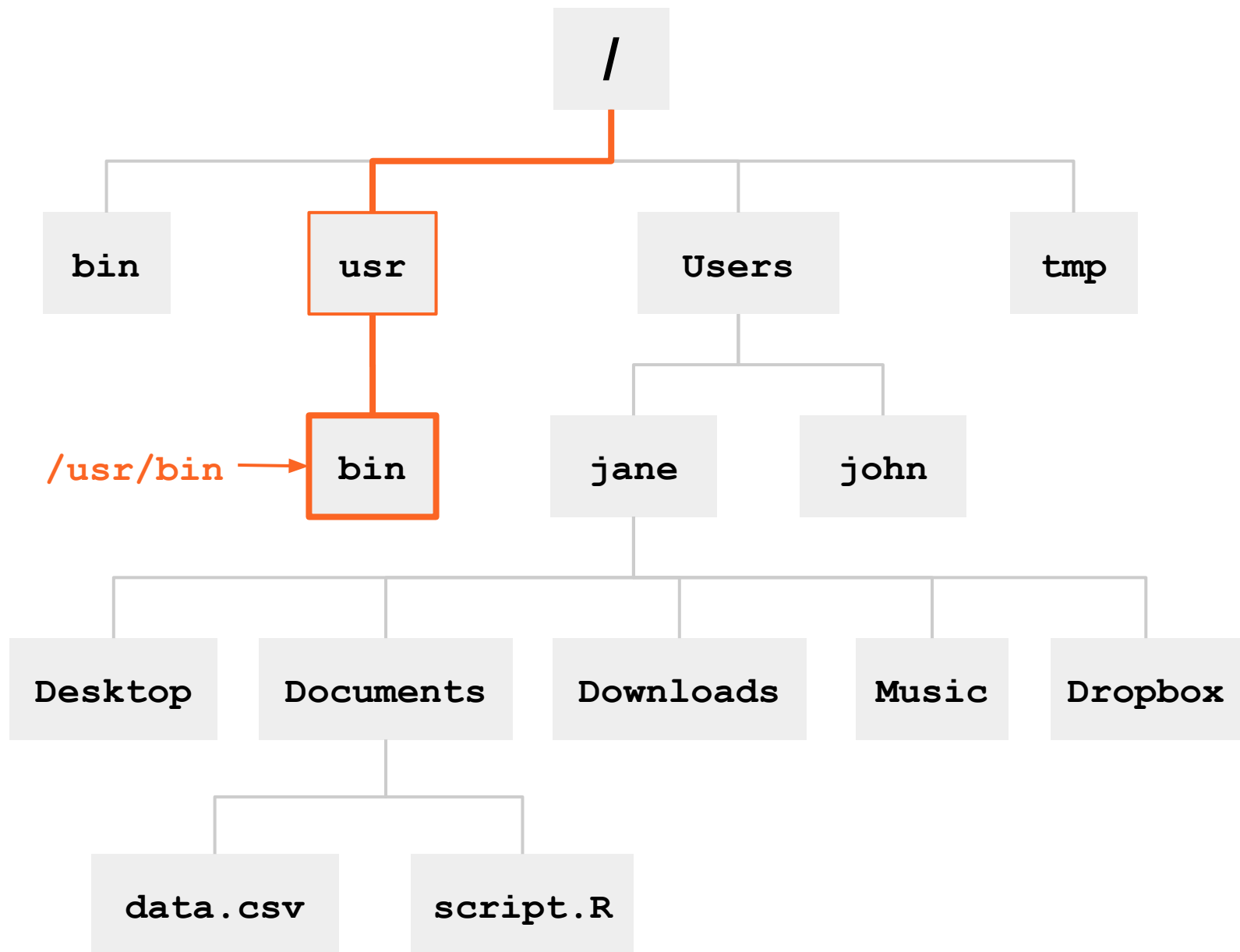
/A/B/C/D

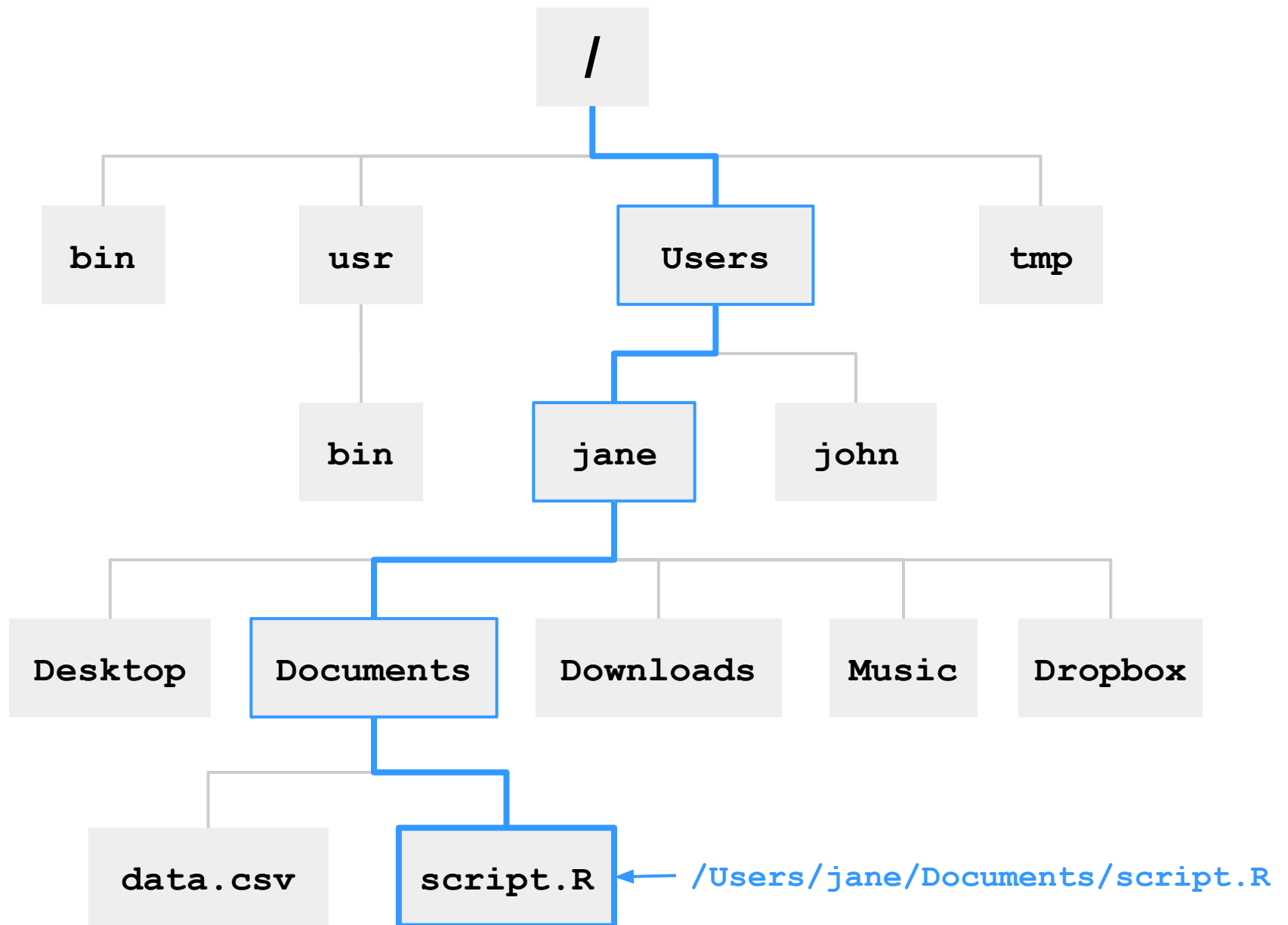


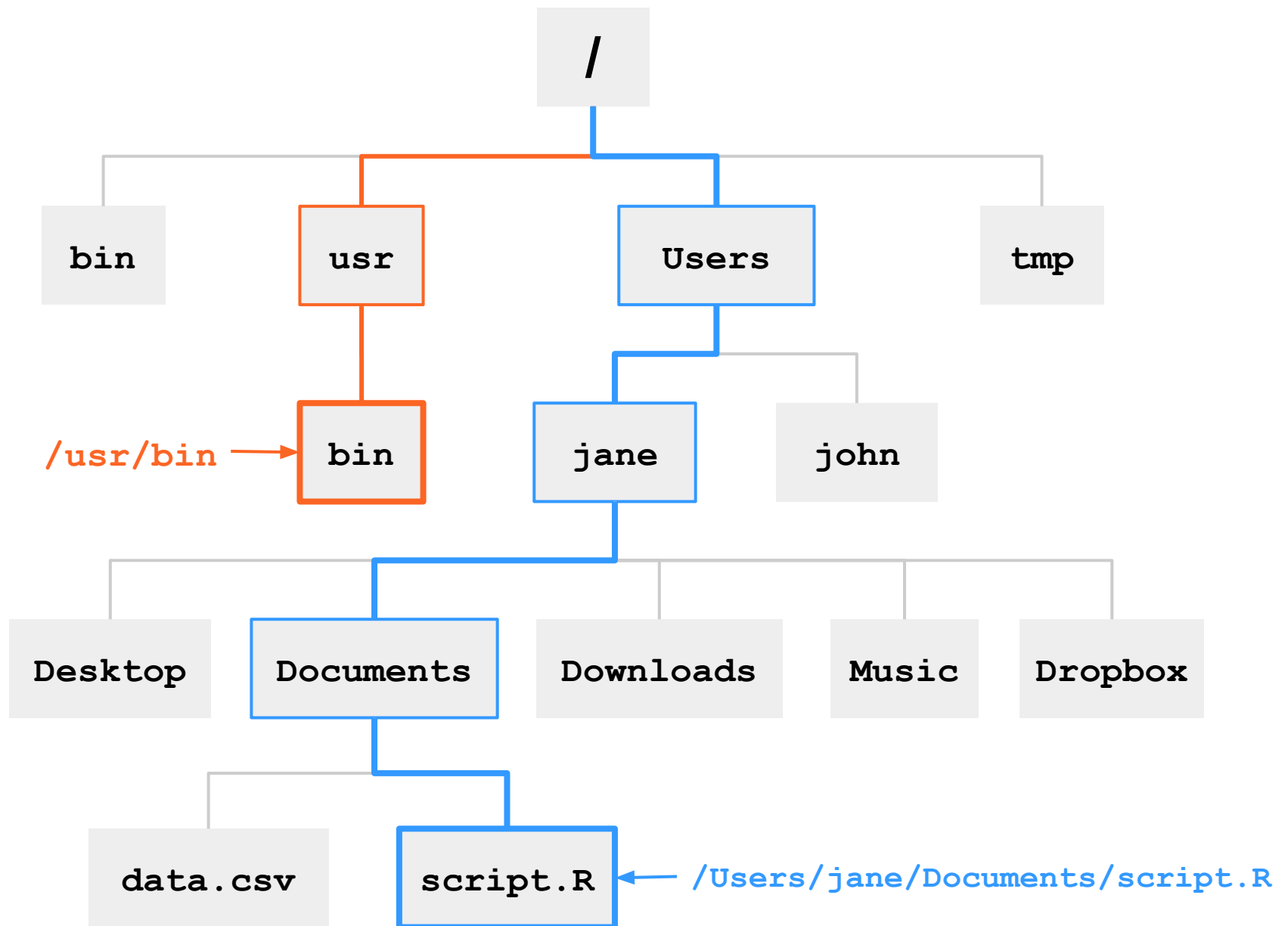
The first slash is the root directory



Subsequent slashes separate directories





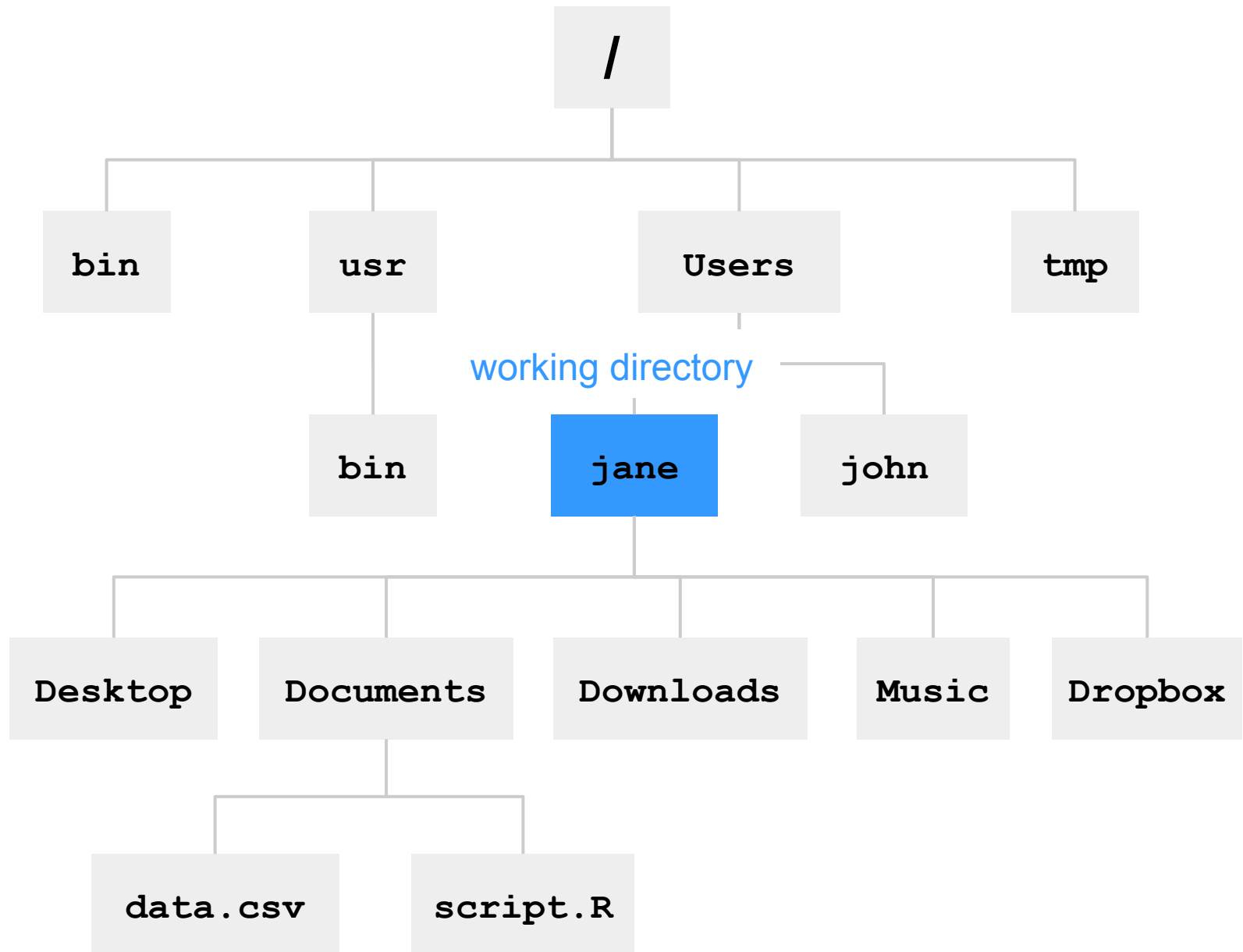


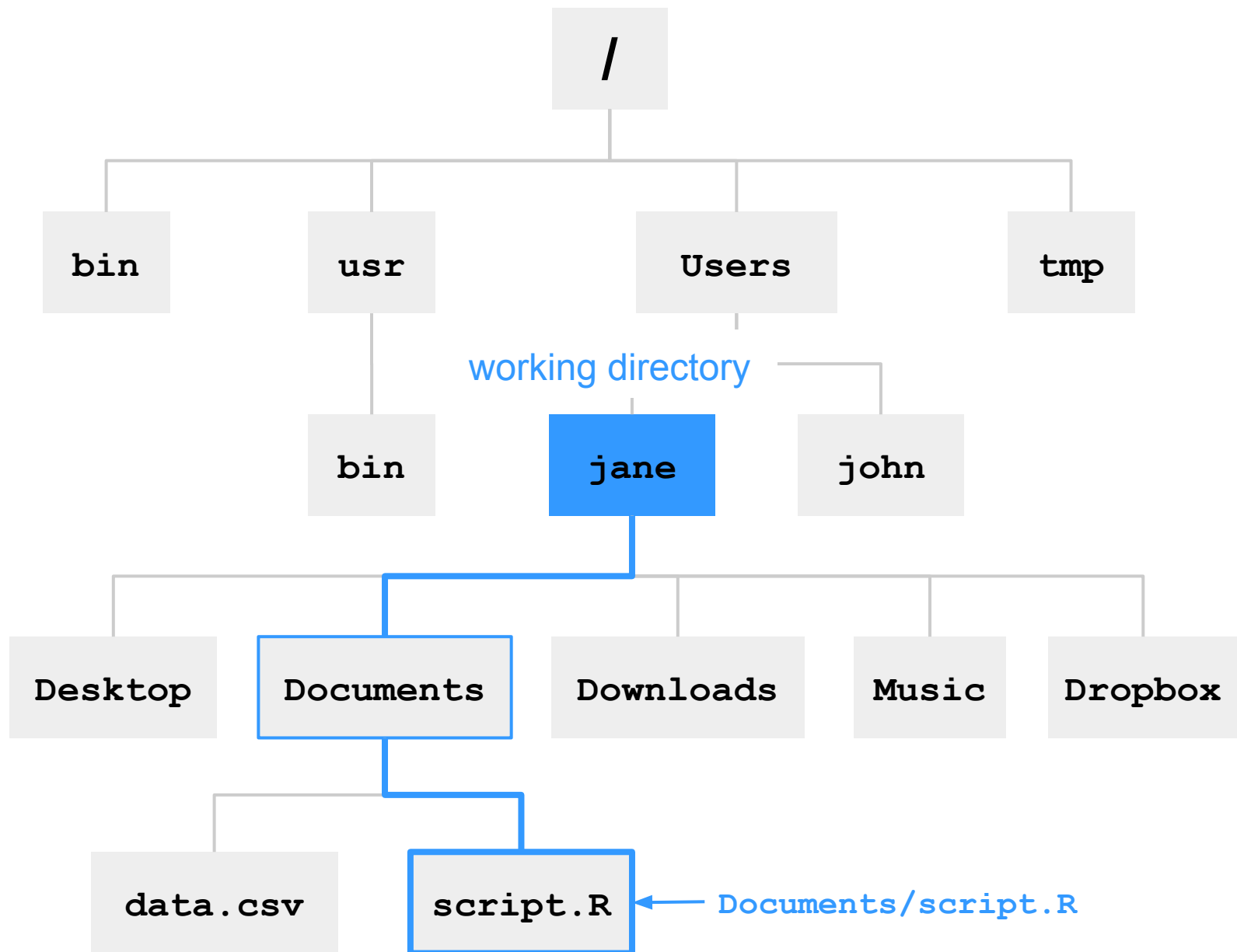
Relative Paths

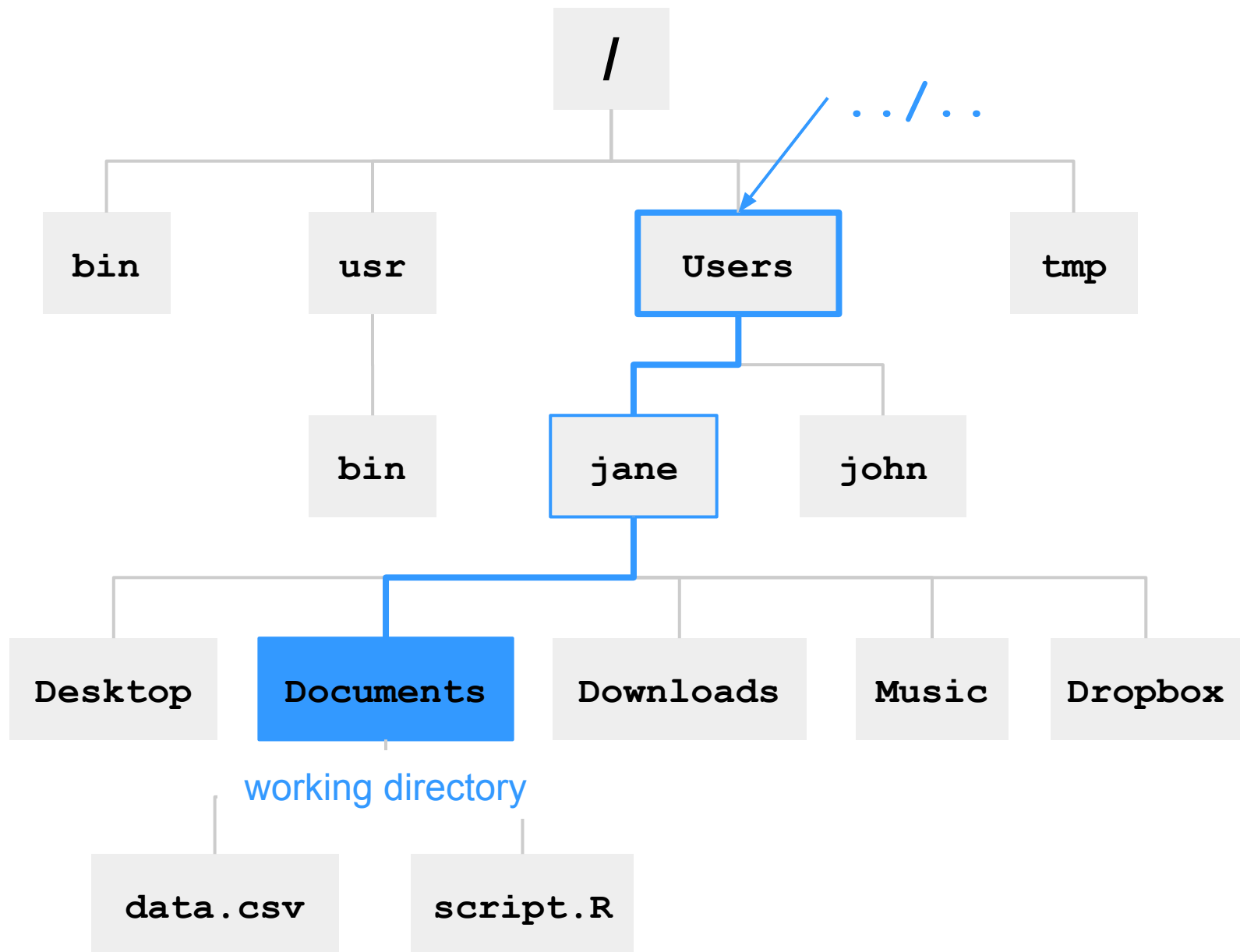
Relative paths: a relative pathname begins at some working directory, moving either up or down the tree

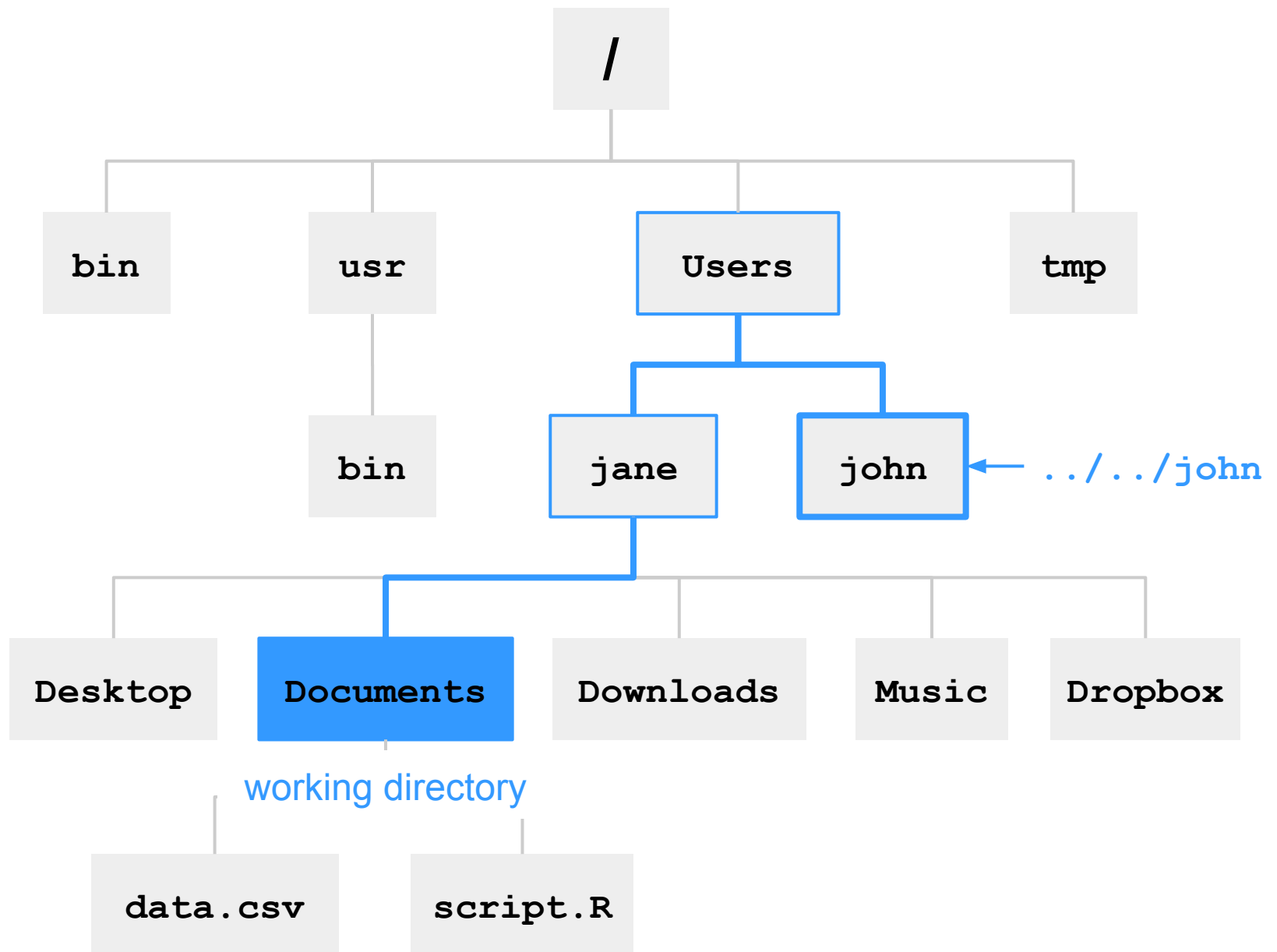
`../C`

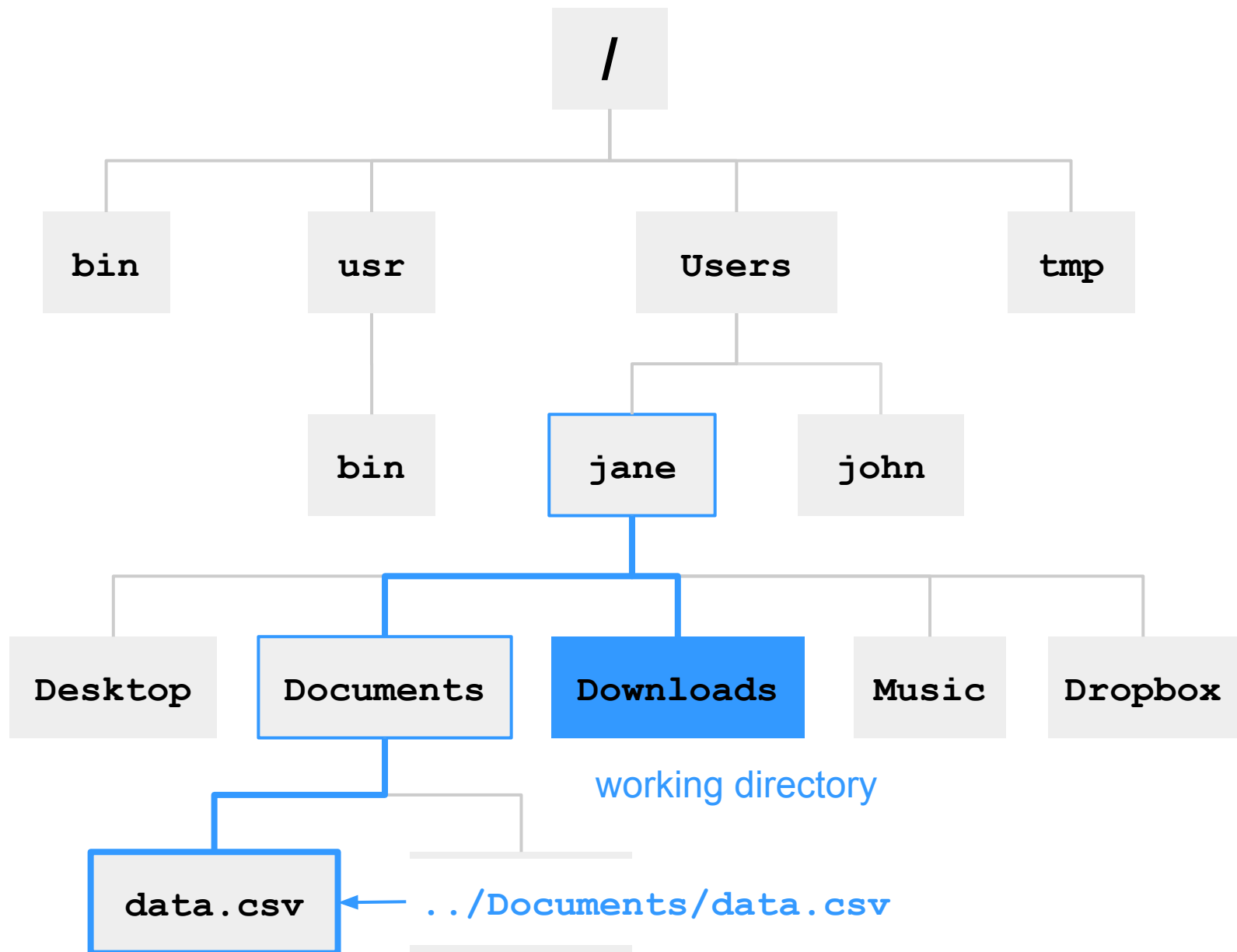
`./B`



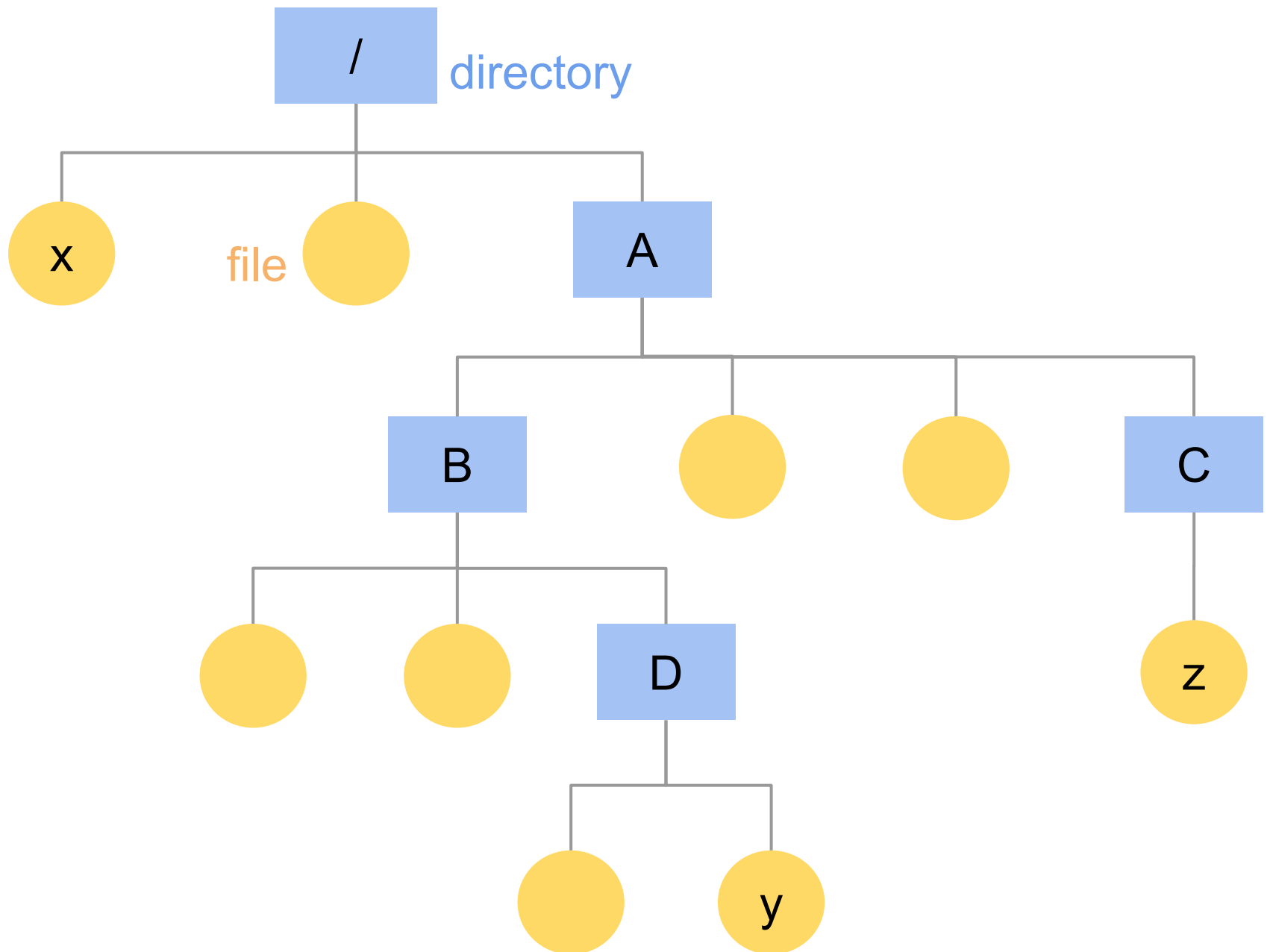








Your turn



Write the following Absolute Paths

From root to C:

From root to z:

From A to y:

From A to z:

Write the following Relative Paths

From D to C:

To x from within C:

To y in D from within C: