In [1]: library(tidyverse) - tidyverse 1.3.0 — — Attaching packages ✓ ggplot2 3.3.3 ✓ purrr 0.3.4 ✓ dplyr 1.0.4 ✓ tibble 3.0.6 ✓ tidyr 1.1.2 ✓ stringr 1.4.0 ✓ forcats 0.5.1 ✓ readr 1.4.0 - Conflicts tidyverse conflicts() — \* dplyr::filter() masks stats::filter() \* dplyr::lag() masks stats::lag() In [2]: (x < -c(a = 1, b = 2, c = 3))**a:** 1 **b:** 2 **c:** 3 Warm-up: 1. Create a named vector of the ages of some of your family members 2. Create various subsets of this vector using numeric vectors 3. Try conditional subsetting (i.e. all ages above or below a certain value) 4. Now subset using a character of explicit names 5. What is the type of this vector? In [6]: (family ages  $\leftarrow$  c(me = 29, sibling = 40, dad = 82, wife = 30, grandmother = 98)) me: 29 sibling: 40 dad: 82 wife: 30 grandmother: 98 In [7]: #subsetting using a numeric vector family ages[c(1,3)] me: 29 dad: 82 In [8]: #conditional subsetting family ages[family ages > 40] dad: 82 grandmother: 98 In [9]: family ages > 40 me: FALSE sibling: FALSE dad: TRUE wife: FALSE grandmother: TRUE In [11]: #subsetting using the names family ages[c("me", "dad", "sibling")] me: 29 dad: 82 sibling: 40 In [12]: #find the type of a vector typeof(family ages) 'double' In [16]: (family relationships = c(me = "Mike", dad = "Mike", mom = "Ana")) typeof(family relationships) me: 'Mike' dad: 'Mike' mom: 'Ana' 'character' In [18]: names(family\_ages) 'me' · 'sibling' · 'dad' · 'wife' · 'grandmother' There are a variety of functions that we can apply to vectors • sum(): Sum of elements • prod(): Product of elements mean(): Mean of elements sd(): Standard deviation of the elements var(): Variation of the elements median(): Median of elements min(): Minimum max(): Maximum range(): Range of the values summary(): Summary statistics for the vector • unique(): Returns a vector with the unique values • which(): From a logical statement, returns the indices In [19]:  $x \leftarrow c(1, 2, 3, 4, 5, 10, 20)$ summary(x) Min. 1st Qu. Median Mean 3rd Qu. Max. 1.000 2.500 4.000 6.429 7.500 20.000 In [22]: # returns a two element vector range(x) 1 · 20 In [23]: # which indices of x correspond to values greater than 6? which(x > 6) 6 · 7 In [24]: # the following are equivalent x[x > 6]x[which(x > 6)]10 · 20 10 · 20 In [26]: # remember to remove missing values for certain functions! mean(c(1, 2, NA, 4), na.rm = TRUE)2.333333333333333 In [27]: # we can directly alter the elements of the list # this changes the local variable x x[1] < -10 $10 \cdot 2 \cdot 3 \cdot 4 \cdot 5 \cdot 10 \cdot 20$ Arithmetic operations and recycling • Standard Linear Algebra operations work as expected Adding two vectors of the same length Scaling a vector Products and Divsion occur element-wise Vectors of different lengths can be recycled The smaller vector is repeated until it is the necessary size ■ For vectors of length one, this is similar to *broadcasting* in Python In [32]:  $x \leftarrow c(1, 2, 3, 4)$  $y \leftarrow c(0, 0, 1, 2)$  $2 \cdot 4 \cdot 6 \cdot 8$ In [36]:  $x \leftarrow c(1, 2, 3, 4, 5, 6, 7, 8, 9)$ y < -c(1, 2) $z \leftarrow c(1, 2, 1, 2, 1, 2, 1, 2, 1)$ x\*y  $X \times Z$ Warning message in x \* y: "longer object length is not a multiple of shorter object length"  $1 \cdot 4 \cdot 3 \cdot 8 \cdot 5 \cdot 12 \cdot 7 \cdot 16 \cdot 9$  $1 \cdot 4 \cdot 3 \cdot 8 \cdot 5 \cdot 12 \cdot 7 \cdot 16 \cdot 9$ Attempt 1. Create the following four vectors  $x \leftarrow c(1, 2, 3, 4, 5, 6)$  $y \leftarrow c(6, 5, 4, 3, 2, 1)$ z < -c(10, 20)W < -c(0, .1)1. Add, divide, multiply, and scale the vectors x and y. Try to guess what the output will be **before** running the code. 2. Do the same for z and w. 3. Run x + 1. What happened? 4. Run x \* w and y + z. What happened? In [37]:  $x \leftarrow c(1, 2, 3, 4, 5, 6)$  $y \leftarrow c(6, 5, 4, 3, 2, 1)$ z < -c(10, 20)w < -c(0, .1)In [42]: # x + 1 is equivalent to x + c(1, 1, 1, 1, 1, 1) $2 \cdot 3 \cdot 4 \cdot 5 \cdot 6 \cdot 7$ In [44]: x \* w y \* z  $0 \cdot 0.2 \cdot 0 \cdot 0.4 \cdot 0 \cdot 0.6$  $60 \cdot 100 \cdot 40 \cdot 60 \cdot 20 \cdot 20$ In [46]: mpg A tibble: 234 × 11 manufacturer model displ year fl class cyl trans drv cty hwy <chr> <dbl> <int> <int> <chr> <chr> <int> <int> <chr> 1.8 1999 18 audi а4 auto(I5) 29 compact audi 1.8 1999 4 manual(m5) 21 29 а4 compact audi 2008 20 2.0 4 manual(m6) f 31 а4 р compact 2008 30 audi a4 2.0 21 compact auto(av) 2.8 1999 16 26 audi auto(I5) а4 р compact 2.8 1999 audi a4 6 manual(m5) 18 26 compact audi 3.1 2008 auto(av) 18 27 а4 compact р a4 quattro 1999 audi 1.8 18 26 manual(m5) compact 1999 25 audi a4 quattro 1.8 auto(I5) 16 4 compact р 2.0 2008 4 manual(m6) 20 28 audi a4 quattro 4 compact audi 2008 19 27 a4 quattro 2.0 auto(s6) 4 р compact audi a4 quattro 2.8 1999 4 15 25 compact auto(I5) 1999 17 25 audi a4 quattro 2.8 6 manual(m5) 4 compact р 3.1 2008 17 audi a4 quattro auto(s6) 25 4 compact 2008 6 manual(m6) audi a4 quattro 4 15 25 р compact 1999 audi a6 quattro 2.8 auto(I5) 4 15 24 midsize audi a6 quattro 2008 17 25 midsize auto(s6) 4 р 4.2 2008 23 audi a6 quattro 8 auto(s6) 16 midsize 4 chevrolet c1500 suburban 2wd 5.3 2008 20 8 auto(I4) 14 suv chevrolet c1500 suburban 2wd 2008 5.3 8 15 auto(I4) 11 е suv chevrolet c1500 suburban 2wd 2008 20 5.3 8 auto(I4) 14 suv chevrolet c1500 suburban 2wd 5.7 1999 auto(I4) 13 17 suv 2008 17 chevrolet c1500 suburban 2wd 6.0 8 auto(I4) 12 suv 1999 chevrolet 5.7 16 26 corvette 8 manual(m6) 2seater 1999 23 5.7 15 chevrolet corvette auto(I4) р 2seater 6.2 2008 chevrolet corvette 8 manual(m6) 16 26 2seater р chevrolet 6.2 2008 auto(s6) 15 25 corvette 2seater р 2008 chevrolet corvette 7.0 8 manual(m6) 15 24 2seater р k1500 tahoe 4wd 2008 19 5.3 14 chevrolet auto(I4) 4 suv 5.3 2008 14 chevrolet k1500 tahoe 4wd auto(I4) 11 4 е suv : : 1999 15 19 toyota tacoma 4wd auto(I4) pickup toyota 6 manual(m6) 15 18 toyota tacoma 4wd 4.0 2008 pickup toyota 4.0 2008 toyota tacoma 4wd 6 auto(I5) 16 20 pickup toyota 1999 29 volkswagen gti 2.0 4 manual(m5) 21 compact 2.0 1999 26 compact volkswagen gti auto(I4) 19 2008 2.0 4 manual(m6) 21 29 volkswagen gti р compact 2008 29 2.0 auto(s6) f 22 volkswagen gti р compact 2.8 1999 6 manual(m5) f 17 24 volkswagen gti compact volkswagen jetta 1.9 1999 manual(m5) 33 44 d compact jetta 2.0 1999 4 manual(m5) 21 29 volkswagen compact 1999 2.0 auto(I4) f 19 26 volkswagen jetta compact 2008 f 2.0 auto(s6) 22 29 volkswagen jetta 4 compact р 2008 2.0 21 29 volkswagen jetta manual(m6) р compact auto(s6) volkswagen jetta 2.5 2008 21 29 compact jetta 2.5 2008 5 manual(m5) f 21 29 volkswagen compact f jetta 2.8 1999 6 16 23 volkswagen auto(I4) compact 2.8 1999 manual(m5) 17 24 volkswagen jetta compact 1999 35 1.9 4 manual(m5) 44 subcompact volkswagen new beetle new beetle 1999 volkswagen 1.9 auto(I4) f 29 41 d subcompact 1999 4 manual(m5) 21 29 volkswagen new beetle 2.0 r subcompact volkswagen 2.0 1999 19 26 r subcompact new beetle auto(I4) 2008 5 manual(m5) 20 28 volkswagen new beetle 2.5 r subcompact 2.5 2008 auto(s6) f 20 29 r subcompact volkswagen new beetle f volkswagen passat 1.8 1999 4 manual(m5) 21 29 р midsize 1.8 1999 18 29 midsize volkswagen passat auto(I5) р 2008 auto(s6) 19 28 midsize volkswagen passat 2.0 р 2008 midsize 2.0 4 manual(m6) 21 29 volkswagen passat р 1999 f midsize volkswagen passat 2.8 6 auto(I5) 16 26 р volkswagen passat 2.8 1999 manual(m5) 18 26 midsize р 3.6 2008 auto(s6) 17 26 midsize volkswagen passat р This is similar to the column operations we used before... • Use [[...]] to extract a column of the mpg dataset: x <- mpg[["model"]]</pre> What is x ? In [50]: x <- mpg[["model"]]</pre> X 'a4' · 'a4 quattro' · 'a6 quattro' · 'a6 quattro' · 'a6 quattro' · 'c1500 suburban 2wd' · 'c1500 suburban 'corvette' · 'corvette' · 'corvette' · 'k1500 tahoe 4wd' · 'k1500 tahoe 4wd' · 'k1500 tahoe 4wd' · 'k1500 tahoe 4wd' · 'malibu' · 'malibu' · 'malibu' · 'malibu' · 'caravan 2wd' 'caravan 2wd' · 'dakota pickup 4wd' · 'dakot 'dakota pickup 4wd' · 'dakota pickup 4wd' · 'durango 4wd' · 'd 'ram 1500 pickup 4wd' · 'expedition 2wd' · 'expedition 'explorer 4wd' · 'f150 pickup 4wd' · ' 'f150 pickup 4wd' · 'f150 pickup 4wd' · 'f150 pickup 4wd' · 'mustang' · 'mustang · 'mustang · 'mustang · 'mustang · 'mustang · 'mustang · 'civic' · 'civic 'tiburon' · 'tiburon' · 'tiburon' · 'tiburon' · 'grand cherokee 4wd' 'grand cherokee 4wd' · 'grand cherokee 4wd' · 'grand cherokee 4wd' · 'range rover' · 'range rover' · 'range rover' · 'range rover' · 'navigator 2wd' · 'navigator 2wd' · 'navigator 2wd' · 'mountaineer 4wd' · 'mountaineer 4wd' · 'mountaineer 4wd' · 'mountaineer 4wd' · 'altima' · ' 'maxima' · 'maxima' · 'pathfinder 4wd' · 'pathfinder 4wd' · 'pathfinder 4wd' · 'pathfinder 4wd' · 'grand prix' 'forester awd' · 'impreza 'impreza awd' · 'impreza awd' · 'impreza awd' · 'impreza awd' · '4runner 4wd' 'camry' · 'camry' · 'camry' · 'camry' · 'camry' · 'camry · 'camry solara' 'corolla' · 'corolla' · 'corolla' · 'corolla' · 'corolla' · 'land cruiser wagon 4wd' · 'land cruiser wagon 4wd' · 'toyota tacoma 4wd 'toyota tacoma 4wd' · 'toyota tacoma 4wd' · 'toyota tacoma 4wd' · 'toyota tacoma 4wd' · 'gti' · 'gti' · 'gti' · 'gti' · 'gti' · 'gti' · 'jetta' · 'new beetle' · 'passat' · Lists Tibbles are secretly fancy lists of vectors • Lists are ordered collections of elements like vectors, but the elements can be anything! In [51]: (myList <- list(1L, 2, "hello"))</pre> 1. 1 2. 2 3. 'hello' In [52]: c(1L, 2, "hello") '1' · '2' · 'hello' In [53]: # lists can contain other lists as well (anotherList <- list(myList, 2.2, "a string "))</pre> 1. A. 1 B. 2 C. 'hello' 2. 2.2 3. 'a string ' In [54]: # a good way to look at the STRucture of a list str(anotherList) List of 3 \$:List of 3 ..\$ : int 1 ..\$ : num 2 ..\$ : chr "hello" \$ : num 2.2 \$ : chr "a string " In [55]: # like with vectors, [] returns another list myList[c(2,3)]anotherList[c(1,2)] 1. 2 2. 'hello' 1. A. 1 B. 2 C. 'hello' 2. 2.2 In [56]: # The [[ ]] operator "pops" out an element of a list # A list of one element, consisting of a list anotherList[1] # a list of three elements anotherList[[1]] 1. A. 1 B. 2 C. 'hello' 1. 1 2. 2 3. 'hello' In [57]: str(anotherList[1]) str(anotherList[[1]]) List of 1 \$:List of 3 ..\$ : int 1 ..\$ : num 2 ..\$ : chr "hello" List of 3 \$ : int 1 \$ : num 2 \$ : chr "hello" In [58]: # lists also have lengths length(anotherList) length(myList) 3 3 In [59]: # Just like vectors, we can create lists with named elements myList <- list(a = 1, b = list("hello", "a</pre> string"), c = 2.2)# The elements can be accessed just like with vectors using [] myList[c('a', 'c')] # or we can pop out an element using [[ ]] myList[['b']] # The \$ operator is shorthand for [[ ]] myList\$b \$a \$c 2.2 1. 'hello' 2. 'a string' 1. 'hello' 2. 'a string' We can think of tibbles as lists of named vectors In [60]: typeof(mpg) 'list' A tibble: 234 × 11 manufacturer model displ year cyl drv cty hwy fl class trans <int> <chr> <chr> <chr> <dbl> <int> <int> <chr> <chr> <int> <chr> 1999 1.8 18 29 audi a4 4 auto(I5) р compact 1999 29 audi a4 1.8 4 manual(m5) compact audi 2.0 2008 4 manual(m6) f 20 31 а4 compact р 2008 audi 2.0 21 30 a4 auto(av) compact 1999 26 audi 2.8 auto(I5) 16 а4 compact р 2.8 1999 audi 6 manual(m5) 18 26 a4 compact audi 2008 18 27 a4 3.1 auto(av) f compact р audi a4 quattro 1.8 1999 4 18 26 4 manual(m5) compact 1999 25 audi a4 quattro 1.8 auto(I5) 4 16 р compact 2.0 2008 audi a4 quattro 4 manual(m6) 20 28 4 compact 2008 19 27 audi a4 quattro 2.0 auto(s6) 4 р compact 1999 25 audi a4 quattro 2.8 auto(I5) 4 15 compact audi a4 quattro 2.8 1999 6 manual(m5) 17 25 4 compact р 3.1 2008 25 audi a4 quattro auto(s6) 17 4 compact audi 2008 6 manual(m6) 15 25 a4 quattro 3.1 4 compact р audi 1999 a6 quattro 2.8 4 15 24 midsize auto(I5) 2008 17 25 audi a6 quattro 3.1 6 auto(s6) 4 р midsize audi a6 quattro 4.2 2008 8 auto(s6) 16 23 midsize 4 chevrolet c1500 suburban 2wd 2008 20 5.3 8 auto(I4) 14 suv chevrolet c1500 suburban 2wd 2008 5.3 8 auto(I4) 15 11 е suv chevrolet c1500 suburban 2wd 2008 20 5.3 8 auto(I4) 14 suv chevrolet c1500 suburban 2wd 5.7 1999 17 8 auto(I4) 13 suv chevrolet c1500 suburban 2wd 6.0 2008 8 auto(I4) 12 17 r suv 1999 chevrolet 5.7 8 manual(m6) 16 26 corvette 2seater 1999 23 chevrolet 5.7 auto(I4) 15 corvette 2seater р 6.2 2008 chevrolet 8 manual(m6) 26 corvette 16 р 2seater 6.2 2008 chevrolet corvette auto(s6) 15 25 2seater р 2008 8 manual(m6) 15 24 chevrolet corvette 7.0 2seater k1500 tahoe 4wd 2008 8 19 5.3 auto(I4) 4 14 chevrolet suv chevrolet k1500 tahoe 4wd 5.3 2008 8 auto(I4) 14 suv : toyota tacoma 4wd 1999 15 19 toyota 3.4 auto(I4) 4 pickup 2008 6 manual(m6) 15 18 pickup toyota toyota tacoma 4wd 4.0 4 2008 4.0 6 auto(I5) 16 20 pickup toyota toyota tacoma 4wd 4 1999 4 manual(m5) f 21 29 volkswagen gti 2.0 compact 1999 volkswagen gti 2.0 auto(I4) 19 26 compact 2008 4 manual(m6) 21 29 volkswagen gti 2.0 р compact 2008 2.0 auto(s6) 22 29 volkswagen gti compact р 2.8 1999 manual(m5) f 17 24 volkswagen gti compact r 1.9 1999 33 44 volkswagen jetta manual(m5) d compact volkswagen jetta 2.0 1999 manual(m5) 21 29 compact jetta 2.0 1999 auto(I4) f 19 26 volkswagen compact f 2.0 2008 auto(s6) 22 29 volkswagen jetta 4 compact р 2.0 2008 manual(m6) 21 29 volkswagen jetta compact р 21 jetta 2.5 2008 auto(s6) 29 volkswagen compact 2008 volkswagen jetta 2.5 5 manual(m5) 21 29 compact 1999 auto(I4) f 16 23 volkswagen jetta 2.8 6 compact volkswagen 2.8 1999 manual(m5) 17 24 jetta compact 1999 35 volkswagen new beetle 1.9 4 manual(m5) 44 subcompact new beetle 1999 volkswagen 1.9 auto(I4) f 29 41 d subcompact volkswagen new beetle 2.0 1999 4 manual(m5) f 21 29 r subcompact new beetle 2.0 1999 19 26 r subcompact volkswagen auto(I4) 20 2008 manual(m5) 28 volkswagen new beetle 2.5 r subcompact 2008 new beetle 2.5 5 auto(s6) 20 29 r subcompact volkswagen 1999 4 manual(m5) 21 29 midsize volkswagen passat 1.8 р volkswagen passat 1.8 1999 auto(I5) 18 29 midsize р 2008 auto(s6) 19 midsize volkswagen 2.0 28 passat р 2008 midsize 2.0 4 manual(m6) 21 29 volkswagen passat р 1999 f midsize volkswagen passat 2.8 6 auto(I5) 16 26 р 1999 manual(m5) 26 midsize volkswagen passat 2.8 18 р volkswagen passat 3.6 2008 auto(s6) 17 26 р midsize In [ ]: # list(manufactor = c("audi", "audi", ....), model = c("a4", "a4", ...), ...) The following are equivalent: mpg\$year mpg[['year']] They each return the column vector named 'year'. It is an integer vector. We can see this from the table display or if we pass the vector through is\_integer(). In [ ]: mpg %>% transmute(total\_mpg = cty + hwy) In [ ]: mpg %>% group\_by(drv) %>% summarize(mean\_hwy = mean(hwy)) • The column operations we use for creating new variables are inherited from vector operations and vector functions • For example, adding two columns is equivalent to adding two vectors (because columns are vectors!) In [61]: # tibbles are easy to create from vectors # Recall: vectors of shorter lengths are recycled tibble(names = c("me", "myself", "I"), ages = 29,  $name\_lengths = c(2L, 5L, 1L),$ age\_times\_nl = ages \* name\_lengths) A tibble:  $3 \times 4$ names ages name\_lengths age\_times\_nl <chr> <dbl> <int> <dbl> 2 29 58 me 29 5 145 myself 29 1 29 In [62]: # a tribble is a different way of creating a tibble # designed for data entry tribble( -x, -y, -z, #--|--| "a", 2, 3.6, "b", 1, 8.5 A tibble:  $2 \times 3$ <chr> <dbl> <dbl> b 1 8.5