Before we start...

Import the `titanic` dataset into R

https://web.stanford.edu/class/archive/cs/cs109/cs109.1166/problem12.html

Install the packages ONCE (to check, skip this step)

```
> install.packages(“tidyverse”)
```

After installing, you must load the packages every time you open R

```
> library(tidyverse)
```

Concept of data analysis and maintenance Illustration by WOOBRO DESIGN on Iconscout

Introduction to R for Research

Data Wrangling with Tidyverse

Penn State, University Libraries, Research Informatics and Publishing

Workshop Housekeeping

Questions?

Use the chat or raise hand feature.

Feedback Survey

After each session, please fill out the Qualtrics survey (link in chat and emailed out after session).

If you could live anywhere in the world for a year, where would it be?

Motivation

What is the purpose of Data Wrangling?

Agenda

- Why Tidyverse?
- Quick R Recap
- Loading Libraries
- The Fundamental Functions
- Exporting Datasets
- Additional Exercises

What is the Tidyverse?

Over 87 packages combined together

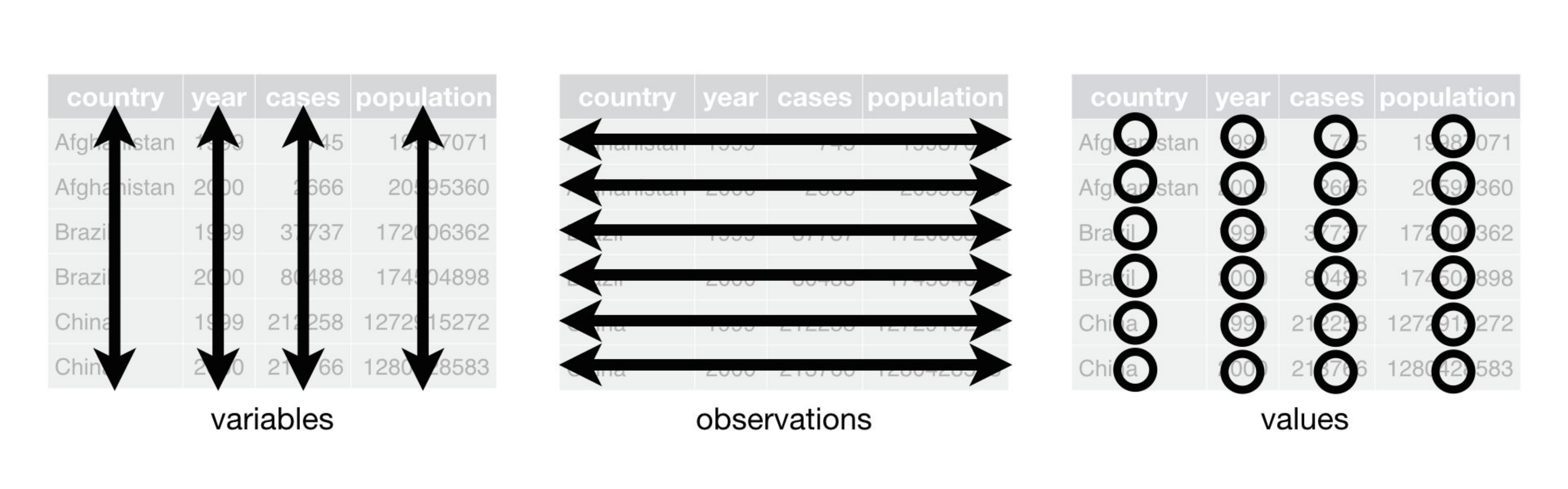
Packages align on data processing philosophy

Encompasses all aspects of a project, including data processing, visualization, modeling

https://cran.r-project.org/web/packages/tidyverse/vignettes/paper.html https://r4ds.had.co.nz/tidy-data.html

Tidyverse Philosophy

“Variables are in columns, observations are in rows, and values are in cells.”



R Recap

General Comments

If you see: > print(“Hello World”)

Run print(“Hello World”) in R Script. Do not include the `>`.

Use `#` to write comments - code after # is not run.

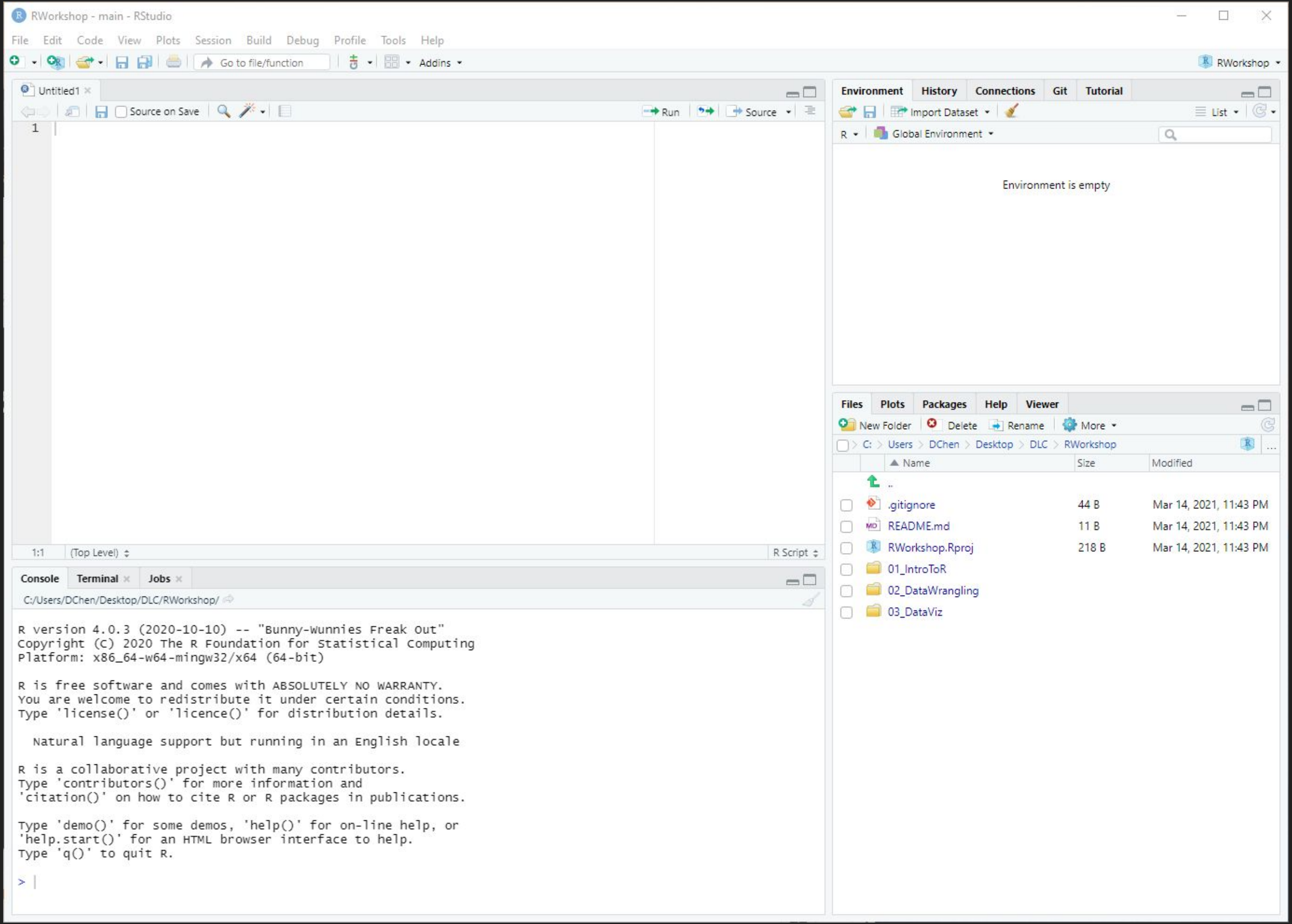
```
> # This is not run
```

RStudio

Type code into Script

Code output shows in Console

Variables appear in Environment



C: > Users > DChen > Desktop > DLC > RWorkshop

Name	Size	Modified
..		
.gitignore	44 B	Mar 14, 2021, 11:43 PM
README.md	11 B	Mar 14, 2021, 11:43 PM
RWorkshop.Rproj	218 B	Mar 14, 2021, 11:43 PM
01_IntroToR		
02_DataWrangling		
03_DataViz		

C:/Users/DChen/Desktop/DLC/RWorkshop/

```
R version 4.0.3 (2020-10-10) -- "Bunny-Wunnies Freak Out"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> |
```

Variable Assignment

variable <- value

Assignment
Operator

(Can also be `=`)

Common Data Types

Data Type	Examples
Numeric	-5, 1, 3.33, 100, pi
Integer	-5L, 1L, 100L
Character	‘words’, “3.33”, ‘TRUE’, “1L”
Boolean/Logical	TRUE, FALSE, T, F

(HBC Source - https://hbctraining.github.io/Training-modules/IntroR/)

Learn more about functions

- Google “<function name> in R”
- Use `help()` or `?` in R to see documentation

```
> help(sqrt)
```

```
> ?sqrt
```

Installing and Loading Packages

Install the packages ONCE (to check, skip this step)

```
> install.packages(“tidyverse”)
```

After installing, you must load the packages every time you open R

```
> library(tidyverse)
```

The Fundamental Functions

The Fundamental Functions

select	Select columns

select - Select which variables to keep

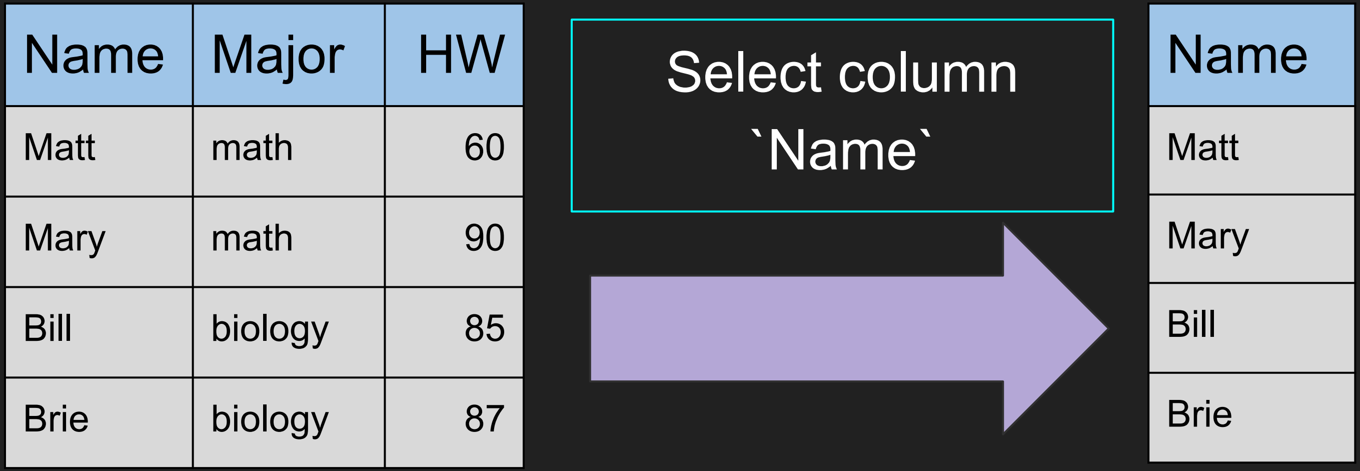
```
select(data, `Column1`, `Column2`,...)
```

Columns to keep or remove (-)

Use column name, number, or colon

select - Keep columns

select(data, Name)



Name	Major	HW
Matt	math	60
Mary	math	90
Bill	biology	85
Brie	biology	87

Select column `Name`

Name
Matt
Mary
Bill
Brie

select - Remove Columns

NOT

select(data, -Major)

Name	Major	HW
Matt	math	60
Mary	math	90
Bill	biology	85
Brie	biology	87

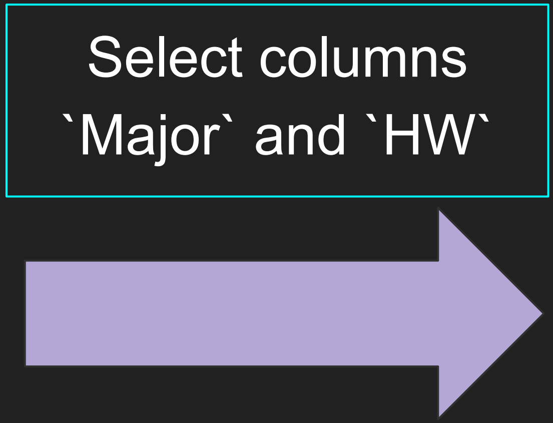
Select all columns **except** \`Major\`

Name	HW
Matt	60
Mary	90
Bill	85
Brie	87

select - Keep multiple columns

```
select(data, Major, HW)
```

Name	Major	HW
Matt	math	60
Mary	math	90
Bill	biology	85
Brie	biology	87



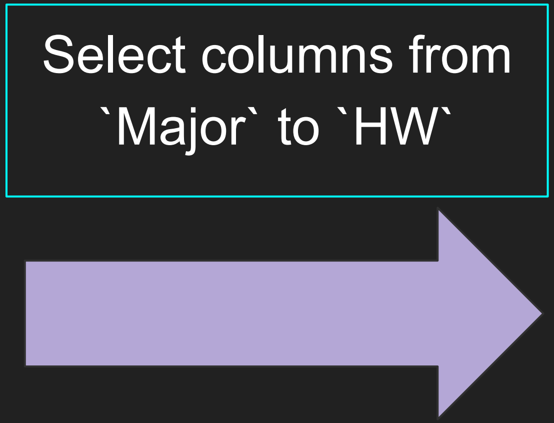
Select columns `Major` and `HW`

Major	HW
math	60
math	90
biology	85
biology	87

select - Keep Multiple Columns

```
select(data, Major:HW)
```

Name	Major	HW
Matt	math	60
Mary	math	90
Bill	biology	85
Brie	biology	87



Select columns from `Major` to `HW`

Major	HW
math	60
math	90
biology	85
biology	87

select - Titanic Exercises

1. Select the variables `Survived`, `Sex`, and `Parents.Children.Aboard`

```
> select(titanic, ______, _______, _______)
```

2. Remove the `Name` variable, and store the new results in `Titanic_nameless`

```
> Titanic_nameless <- select(titanic, ________)
```

3. Store the first four columns of `titanic` in a new dataframe. How many different ways can you do this using `select()`?

select - Titanic Exercise Solutions

1. Select the variables `Survived`, `Sex`, and `Parents.Children.Aboard`

```
> select(titanic, Survived, Sex, Parents.Children.Aboard
```

2. Remove the `Name` variable, and store the new results in `Titanic_nameless`

```
> Titanic_nameless <- select(titanic, -Name)
```

3. Store the first four columns of `titanic` in a new dataframe. How many different ways can you do this using `select()`?

select - Titanic Exercise Solutions

3. Store the first four columns of `titanic` in a new dataframe. How many different ways can you do this using `select()`?

```
> t2 <- select(titanic, 1:4)

> t2 <- select(titanic, Survived:Sex)

> t2 <- select(titanic, Survived, Pclass, Name, Sex)

> t2 <- select(titanic, -Age, -Siblings.Aboard, -Parents.Children.Aboard, -Fare)

> t2 <- select(titanic, -c(Age:Fare))
```

The Fundamental Functions

select	Select columns

The Fundamental Functions

select	Select columns
filter	Filter rows

filter - filter the rows

```
filter(data, <Boolean condition>,...)
```

Condition on the variables

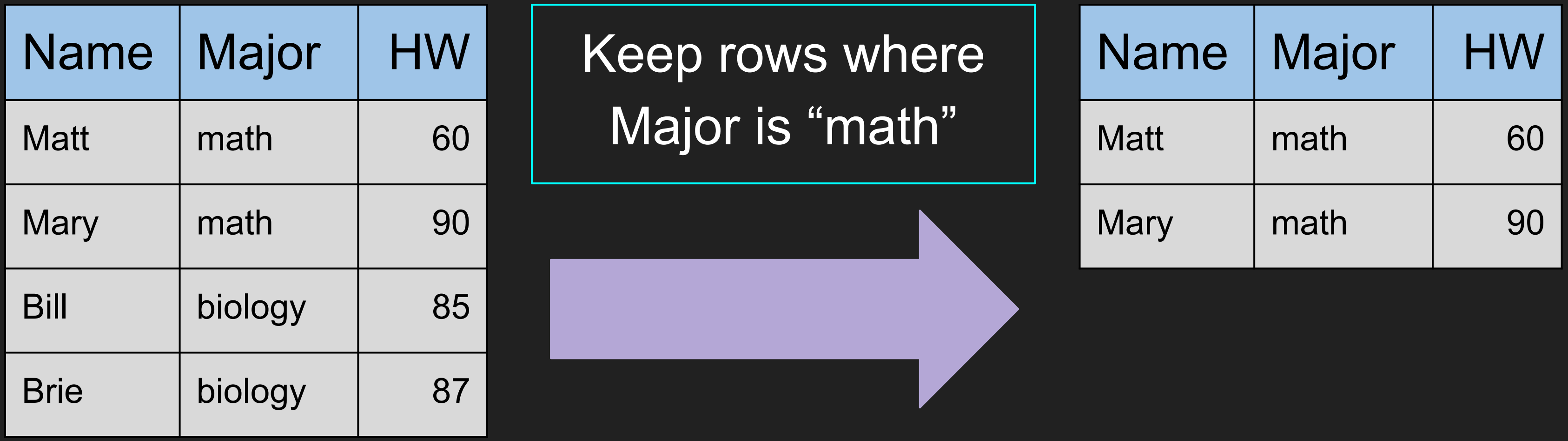
filter - filter the rows

$$\text{filter(data, }\underbrace{\text{<Boolean condition>,...)}}_{\text{Condition on the variables}}$$

<	Less than	>=	Greater than or equal to
<=	Less than or equal to	==	Equal to
>	Greater than	!=	Not equal to

filter - Keep specific value

filter(data, Major == “math”)



Name	Major	HW
Matt	math	60
Mary	math	90
Bill	biology	85
Brie	biology	87

Keep rows where Major is “math”

Name	Major	HW
Matt	math	60
Mary	math	90

filter - Keep range of values

```
filter(data, HW <= 85)
```

Name	Major	HW
Matt	math	60
Mary	math	90
Bill	biology	85
Brie	biology	87

Keep rows where `HW` is less than or equal to 85

Name	Major	HW
Matt	math	60
Bill	biology	85

filter - Multiple Conditions

```
filter(data, HW <= 85, Major == “math”)
```

Name	Major	HW
Matt	math	60
Mary	math	90
Bill	biology	85
Brie	biology	87

Keep rows where `HW` is less than or equal to 85, **AND** `Major` is math

Name	Major	HW
Matt	math	60

filter - Multiple Conditions

AND

```
filter(data, HW <= 85 & Major == “math”)
```

Name	Major	HW
Matt	math	60
Mary	math	90
Bill	biology	85
Brie	biology	87

Keep rows where `HW` is less than or equal to 85, **AND** `Major` is math

Name	Major	HW
Matt	math	60

filter - Multiple Conditions

OR

```
filter(data, HW <= 85 | Major == “biology”)
```

Name	Major	HW
Matt	math	60
Mary	math	90
Bill	biology	85
Brie	biology	87

Keep rows where `HW` is less than or equal to 85, **OR** `Major` is biology

Name	Major	HW
Matt	math	60
Bill	biology	85
Brie	biology	87

filter - Titanic Exercises

1. Create a new dataset with only those who survived (`Survived` value of 1)

```
> survived <- filter(titanic, ______ == _____)
```

2. Only show rows with 1st class (`Pclass`) females (`Sex`)

```
> filter(titanic, _____ == 1 _ Sex == _____)
```

3. How many rows are there if you keep only males between ages 20-30?

filter - Titanic Exercise Solutions

1. Create a new dataset with only those who survived (`Survived` value of 1)

```
> survived <- filter(titanic, Survived == 1)
```

```
> survived <- filter(titanic, Survived == “1”)
```

filter - Titanic Exercise Solutions

1. Create a new dataset with only those who survived (`Survived` value of 1)

```
> survived <- filter(titanic, Survived == 1)
```

```
> survived <- filter(titanic, Survived == “1”)
```

2. Only show rows with 1st class (`Pclass`) females (`Sex`)

```
> filter(titanic, Pclass == 1 & Sex == “female”)
```

```
> filter(titanic, Pclass == 1, Sex == “female”)
```

filter - Titanic Exercise Solutions

3. How many rows are there if you keep only males between ages 20-30?

```
> tm <- filter(titanic, Sex == “male”, Age >= 20, Age <= 30)

> nrow(tm) # 223

> t2 <- filter(titanic, Sex == “male”)

> t3 <- filter(t2, Age >= 20)

> t4 <- filter(t3, Age <= 30)
```

The Fundamental Functions

select	Select columns
filter	Filter rows

Pipes

%>%

Pipes %>%

Pipes (%>%) insert the data into the next line

```
data %>%
    filter(A > 1)
```

==

```
filter(data, A > 1)
```

Pipes %>%

Without pipes, multi-line processing is hard to read

```
df1 <- select(df, A, B, C)

df2 <- filter(df1, A == 1, B > 2)

df3 <- filter(df2, C == 1 | C == 5)
```

Creates unnecessary intermediate datasets

Pipes %>%

Without pipes, multi-line processing is hard to read

```
df1 <- select(df, A, B, C)

df2 <- filter(df1, A == 1, B > 2)

df3 <- filter(df2, C == 1 | C == 5)


df1 <- filter(filter(select(df, A, B, C), A ==
1, B > 2), C == 1 | C == 5)
```

Hard to read

Pipes %>%

With pipes, one line is one logical step

```
df1 <- df %>%
```

Start with dataset `df`

Pipes %>%

With pipes, one line is one logical step

df1 <- df %>%	Start with dataset `df`
select(A, B, C) %>%	Select variables A, B, C

Pipes %>%

With pipes, one line is one logical step

df1 <- df %>%	Start with dataset `df`
select(A, B, C) %>%	Select variables A, B, C
filter(A == 1, B > 2) %>%	Keep rows with A == 1 and B > 2

Pipes %>%

With pipes, one line is one logical step

```
df1 <- df %>%
    select(A, B, C) %>%
    filter(A == 1, B > 2) %>%
    filter(C == 1 | C == 5)
```

Start with dataset `df`

Select variables A, B, C

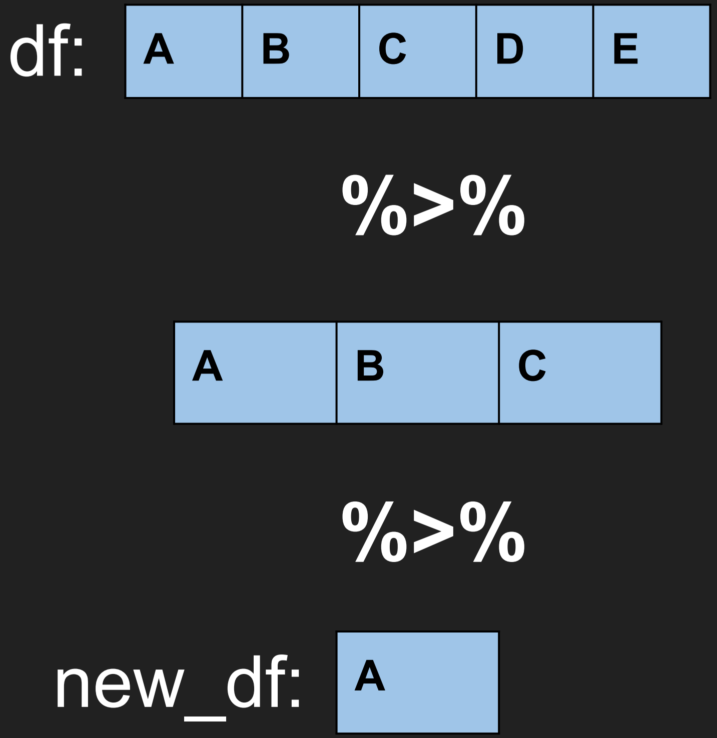
Keep rows with A == 1 and B > 2

Keep rows with C == 1 or C == 5

(Store results in df1)

Pipes %>%

```
new_df <- df %>%
    select(A, B, C) %>%
    select(A)
```



df:

A	B	C	D	E

%>%

A	B	C

%>%

new_df:

A

Pipe %>% - Titanic Exercise

1. Using pipes, store the first four columns and keep only 1st class (`Pclass`)

```
> firstclass <- _________ %>%

>   select(1_4) _____

>   filter(Pclass __ ______)
```

Pipe %>% - Titanic Exercise Solution

1. Using pipes, store the first four columns and keep only 1st class (`Pclass`)

```
> firstclass <- titanic %>%
>   select(1:4) %>%
>   filter(Pclass == 1)
```

The Fundamental Functions

select	Select columns
filter	Filter rows

The Fundamental Functions

select	Select columns
filter	Filter rows
mutate	Create new variables

mutate - Create new variables/columns

```
mutate(., new_variable = value, ...)
```

Computation of new variable

Can be based on another variable, a function, a single value, etc.

mutate - Create var with constant value

```
data %>% mutate(Sect = 1)
```

Name	HW	Ex
Matt	60	75
Mary	90	100
Bill	85	60
Brie	87	85

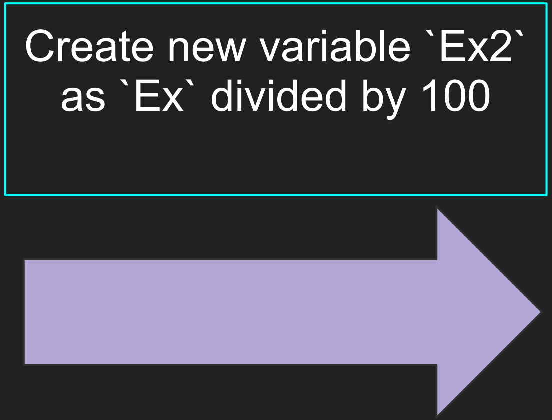
Create new variable \`Sect\` where all values are 1

Name	HW	Ex	Sect
Matt	60	75	1
Mary	90	100	1
Bill	85	60	1
Brie	87	85	1

mutate - Create var based on another var

```
data %>% mutate(Ex2 = Ex/100)
```

Name	HW	Ex
Matt	60	75
Mary	90	100
Bill	85	60
Brie	87	85



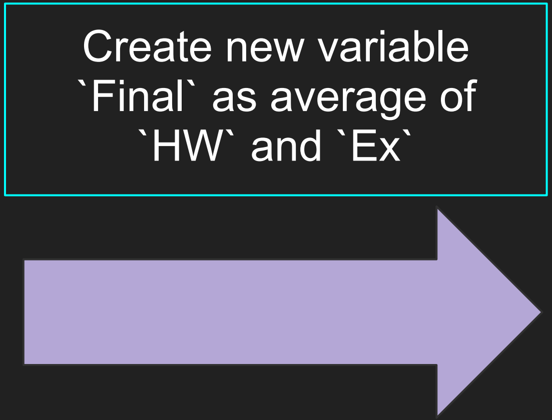
Create new variable `Ex2` as `Ex` divided by 100

Name	HW	Ex	Ex2
Matt	60	75	0.75
Mary	90	100	1.00
Bill	85	60	0.60
Brie	87	85	0.85

mutate - Create var based on other variables

```
data %>% mutate(Final = (HW + Ex)/2)
```

Name	HW	Ex
Matt	60	75
Mary	90	100
Bill	85	60
Brie	87	85



Create new variable `Final` as average of `HW` and `Ex`

Name	HW	Ex	Final
Matt	60	75	67.5
Mary	90	100	95
Bill	85	60	72.5
Brie	87	85	86

mutate - Titanic Exercises

1. `Fare` is currently in pounds. Convert it to dollars (1 pound = 1.37 dollars)

```
> titanic %>% mutate(FareDollars = _____*______)
```

2. How many family members total (combine `Siblings.Spouses.Aboard` and `Parents.Children.Aboard`) were on board for each passenger?

```
> titanic %>% mutate(titanic, Family = _____ + ______)
```

3. Using pipes: remove the name, keep only 1st class survivors, subtract 20 from `Age`. Store these results in a dataset named `AgeAdjusted`.

mutate - Titanic Exercise Solutions

1. `Fare` is currently in pounds. Convert it to dollars (1 pound = 1.37 dollars)

```
> titanic %>% mutate(FareDollars = Fare*1.37)
```

2. How many family members total (combine `Siblings.Spouses.Aboard` and `Parents.Children.Aboard`) were on board for each passenger?

```
> titanic %>% mutate(titanic, Family =
Siblings.Spouses.Aboard + Parents.Children.Aboard)
```

mutate - Titanic Exercise Solutions

3. Using pipes: remove the name, keep only 1st class survivors, subtract 20 from `Age`. Store these results in a dataset named `AgeAdjusted`.

```
> AgeAdjusted <- titanic %>%
>   select(-Name) %>%
>   filter(Pclass == 1, Survived == 1) %>%
>   mutate(Age = Age - 20)
```

The Fundamental Functions

select	Select columns
filter	Filter rows
mutate	Create new variables

The Fundamental Functions

select	Select columns
filter	Filter rows
mutate	Create new variables
summarize	Compute summary variable

summarize - Create new aggregated variables

```
summarize(., new_variable = function(_), ...)
```

Computation of new variable

Function that returns one value
(e.g. mean(), median(), max())

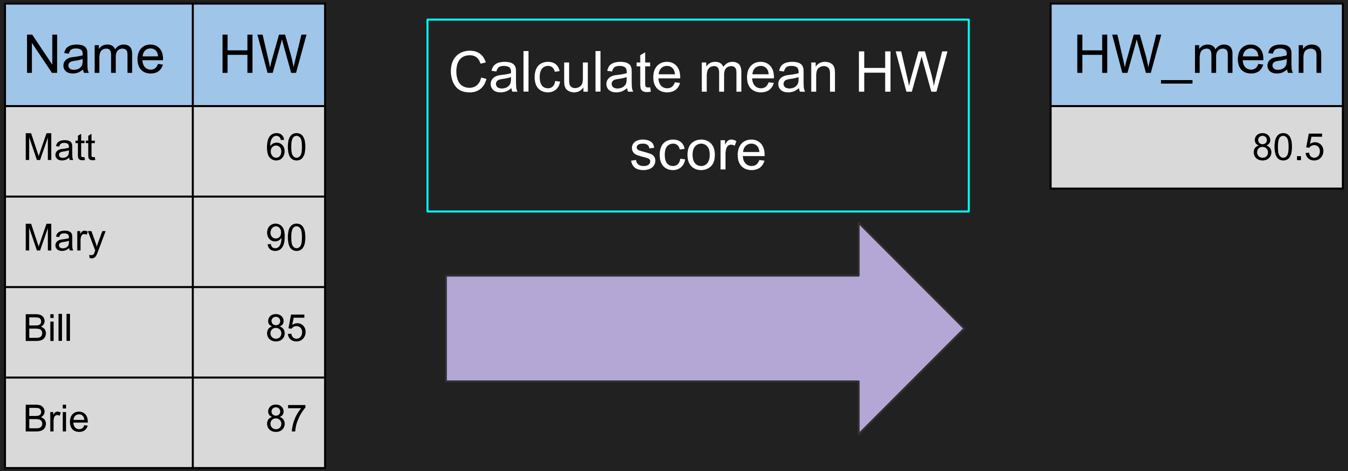
summarize - Create new aggregated variables

summarize() computes a single value or column

mutate() adds a variable to existing dataset

summarize - Compute mean of dataset

```
data %>% summarize(HW_mean = mean(HW))
```



Name	HW
Matt	60
Mary	90
Bill	85
Brie	87

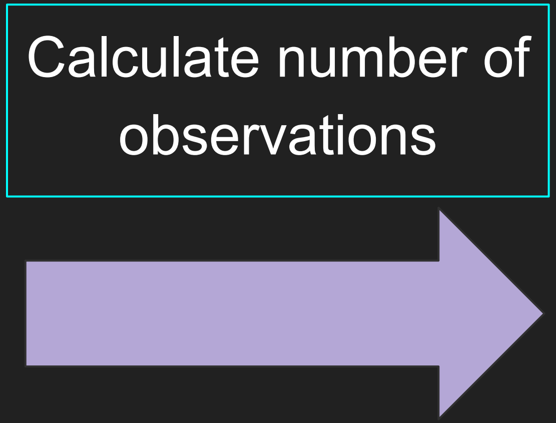
Calculate mean HW score

HW_mean
80.5

summarize - Count total number of observations

```
data %>% summarize(Count = n())
```

Name	HW
Matt	60
Mary	90
Bill	85
Brie	87



Calculate number of observations

Count
4

The Fundamental Functions

select	Select columns
filter	Filter rows
mutate	Create new variables
summarize	Compute summary variable

The Fundamental Functions

select	Select columns
filter	Filter rows
mutate	Create new variables
summarize	Compute summary variable
group_by	Split data by groups

group_by - split data by groups

```
group_by(., `id_var`, ...)
```

Split dataset by `id_var` (or multiple variables)

Dataset split based on unique values of `id_var`

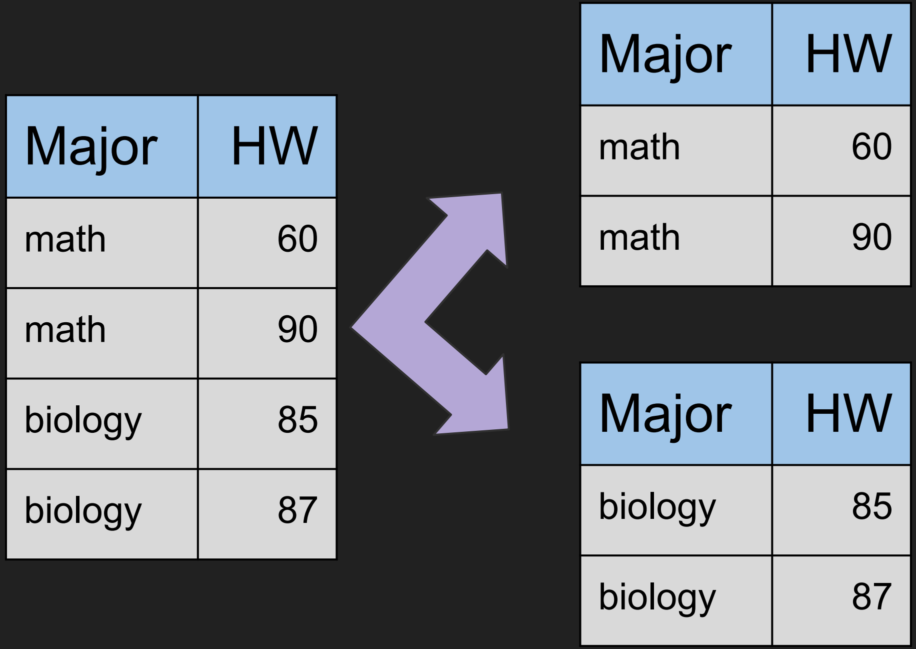
group_by - split data by groups

```
data %>% group_by(., Major)
```

Major	HW
math	60
math	90
biology	85
biology	87

group_by - split data by groups

```
data %>% group_by(., Major)
```



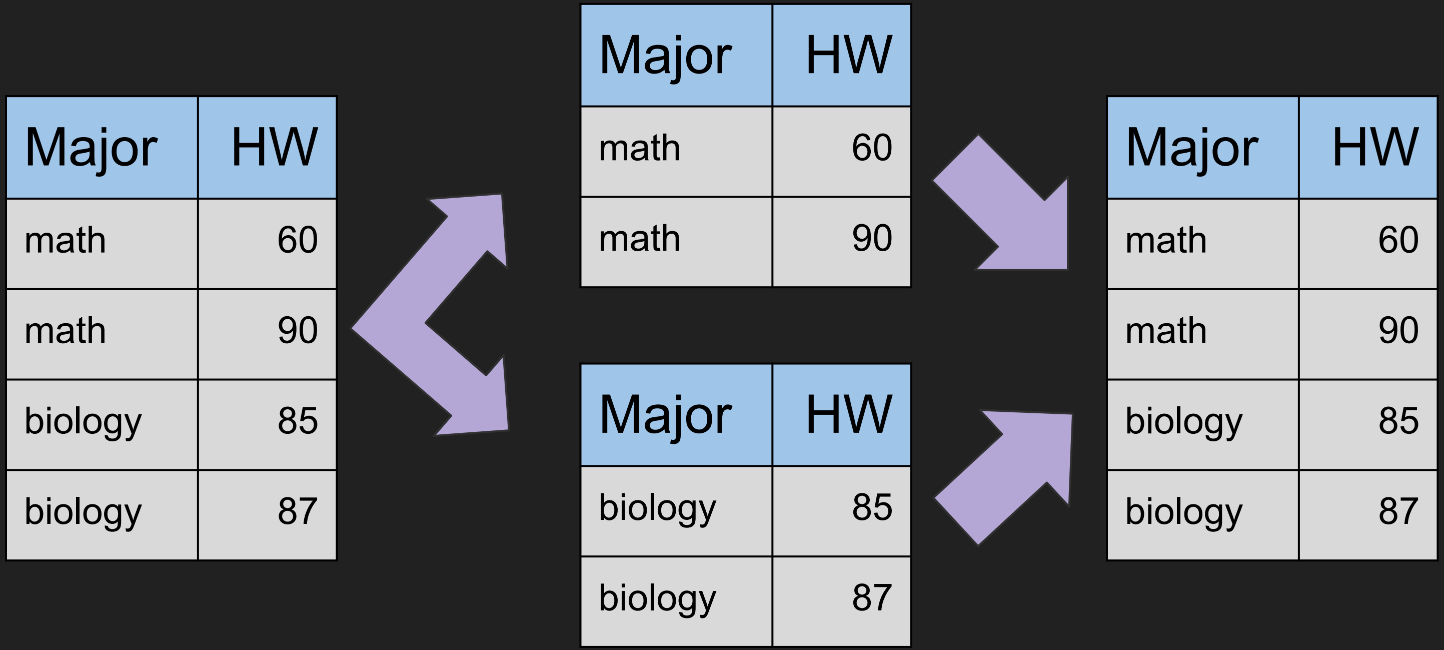
Major	HW
math	60
math	90
biology	85
biology	87

Major	HW
math	60
math	90

Major	HW
biology	85
biology	87

group_by - split data by groups

```
data %>% group_by(., Major)
```



Major	HW
math	60
math	90
biology	85
biology	87

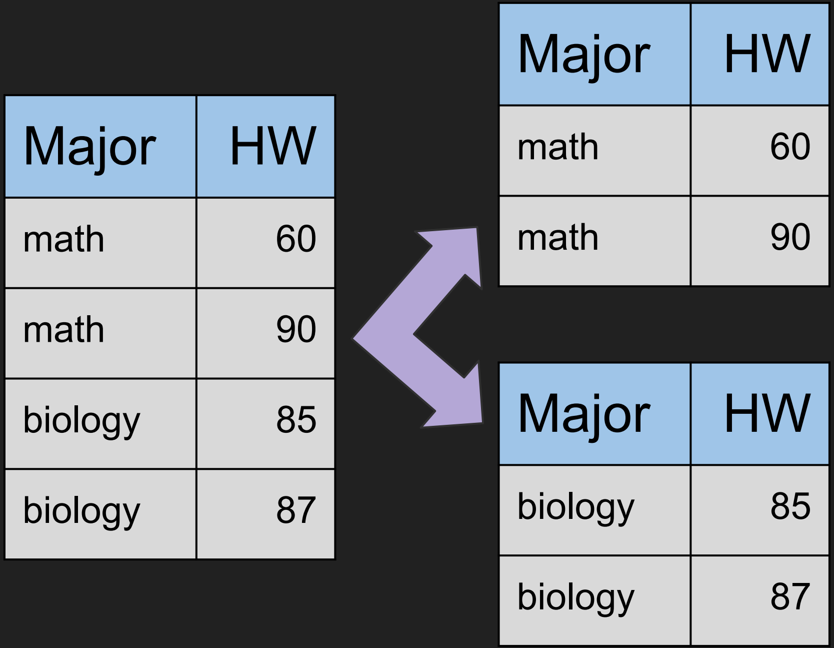
Major	HW
math	60
math	90

Major	HW
biology	85
biology	87

Major	HW
math	60
math	90
biology	85
biology	87

group_by() - split data by groups

group_by(., Major)



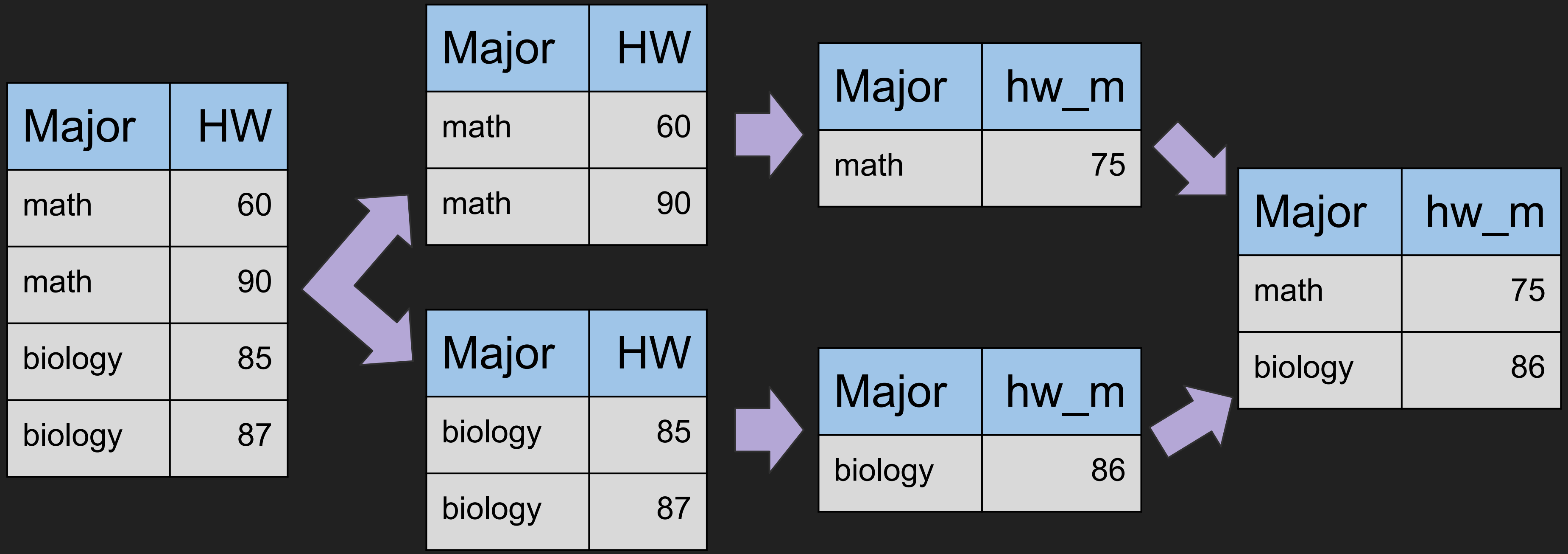
Major	HW
math	60
math	90
biology	85
biology	87

Major	HW
math	60
math	90

Major	HW
biology	85
biology	87

group_by() - split data by groups

```
group_by(., Major) %>% summarize(hw_m = mean(HW))
```



Major	HW
math	60
math	90
biology	85
biology	87

Major	HW
math	60
math	90

Major	HW
biology	85
biology	87

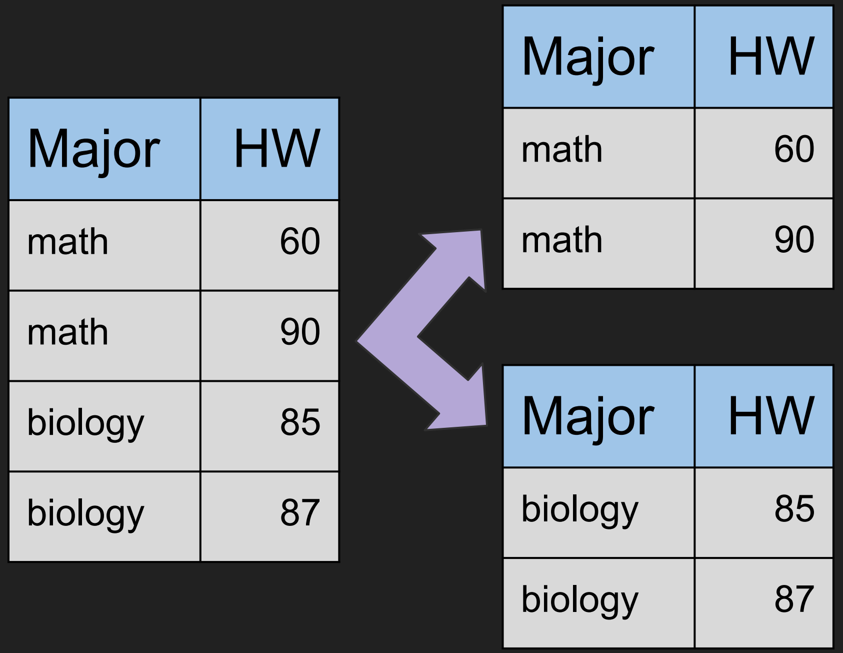
Major	hw_m
math	75

Major	hw_m
biology	86

Major	hw_m
math	75
biology	86

group_by() - split data by groups

group_by(., Major)



Major	HW
math	60
math	90
biology	85
biology	87

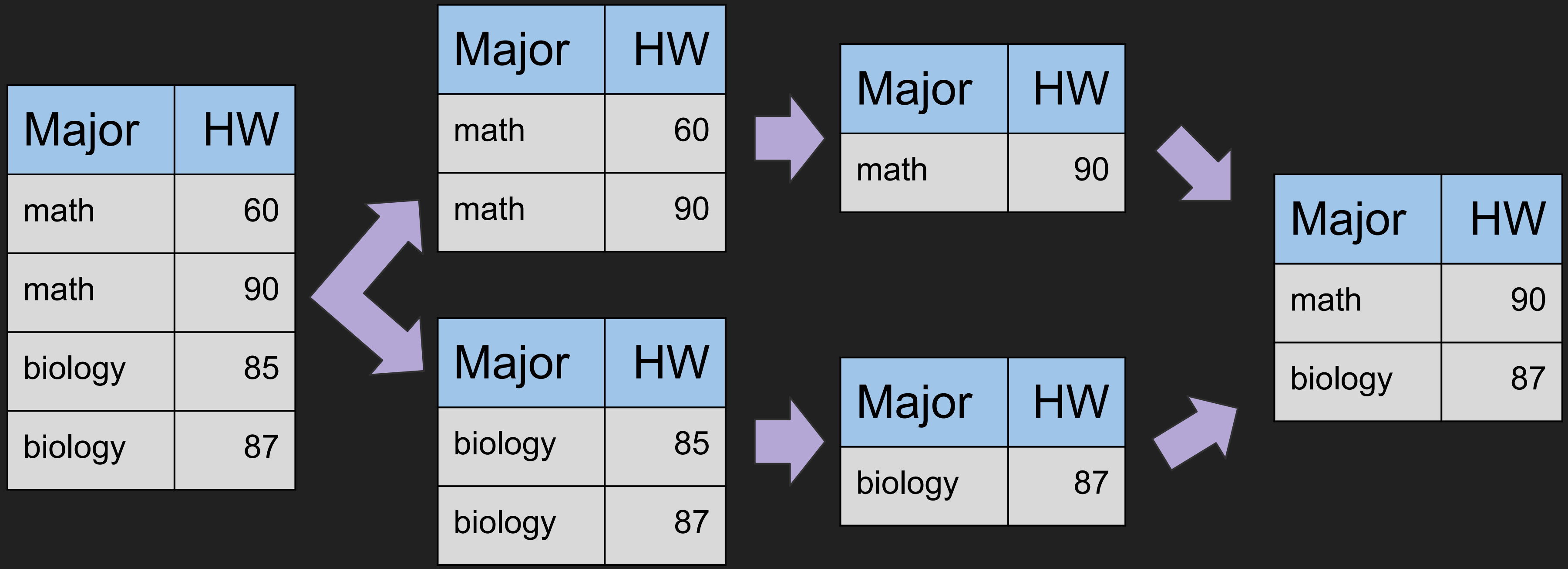
Major	HW
math	60
math	90

Major	HW
biology	85
biology	87

group_by() - split data by groups

Take top HW value

```
group_by(., Major) %>% top_n(1, HW)
```



Major	HW
math	60
math	90
biology	85
biology	87

Major	HW
math	60
math	90

Major	HW
biology	85
biology	87

Major	HW
math	90

Major	HW
biology	87

Major	HW
math	90
biology	87

summarize() and group_by() - Titanic Exercises

1. What was the average `Fare` cost?

```
> titanic %>% summarize(FareMean = mean(______))
```

2. What was the average `Age` for each `Pclass`? Store the results in `ClassAge`.

```
> ClassAge <- titanic %>%
```

```
> group_by(______) %>% summarize(_____ = ____(____))
```

3. What percent of each `Pclass` survived? (Hint: sum() and n() are helpful).

summarize() and group_by() - Titanic Exercise Solutions

1. What was the average `Fare` cost?

```
> titanic %>% summarize(FareMean = mean(Fare))
```

summarize() and group_by() - Titanic Exercise Solutions

1. What was the average `Fare` cost?

```
> titanic %>% summarize(FareMean = mean(Fare))
```

2. What was the average `Age` for each `Pclass`? Store the results in `ClassAge`.

```
> ClassAge <- titanic %>%
> group_by(Pclass) %>%
> summarize(AgeM = mean(Age))
```

summarize() and group_by() - Titanic Exercises

3. What percent of each `Pclass` survived? (Hint: sum() and n() are helpful).

```
> titanic %>%
>   group_by(Pclass) %>%
>   summarize(pct = sum(Survived)/n())
```

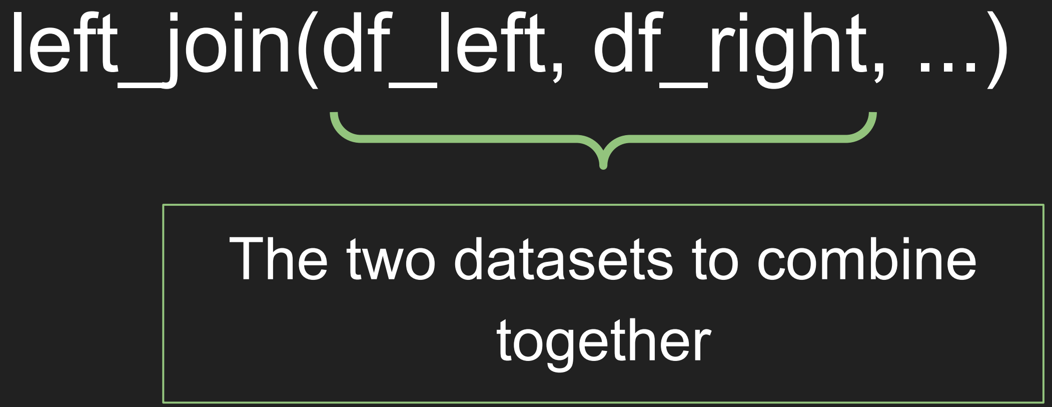
The Fundamental Functions

select	Select columns
filter	Filter rows
mutate	Create new variables
summarize	Compute summary variable
group_by	Split data by groups

The Fundamental Functions

select	Select columns
filter	Filter rows
mutate	Create new variables
summarize	Compute summary variable
group_by	Split data by groups
left_join	Combine datasets

left_join() - combine two datasets



```
left_join(df_left, df_right, ...)
```

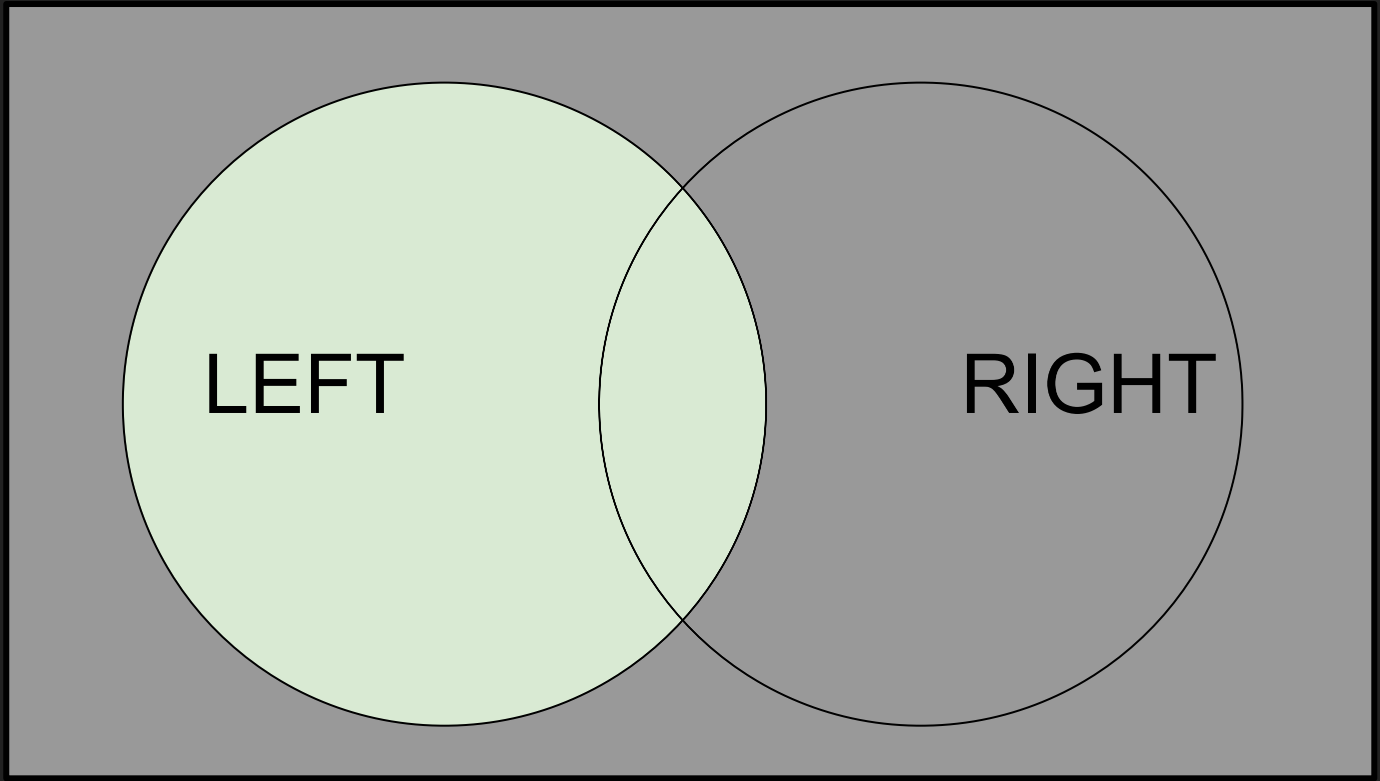
The two datasets to combine together

Join two tables, matching on a key column

left_join() - combine two datasets

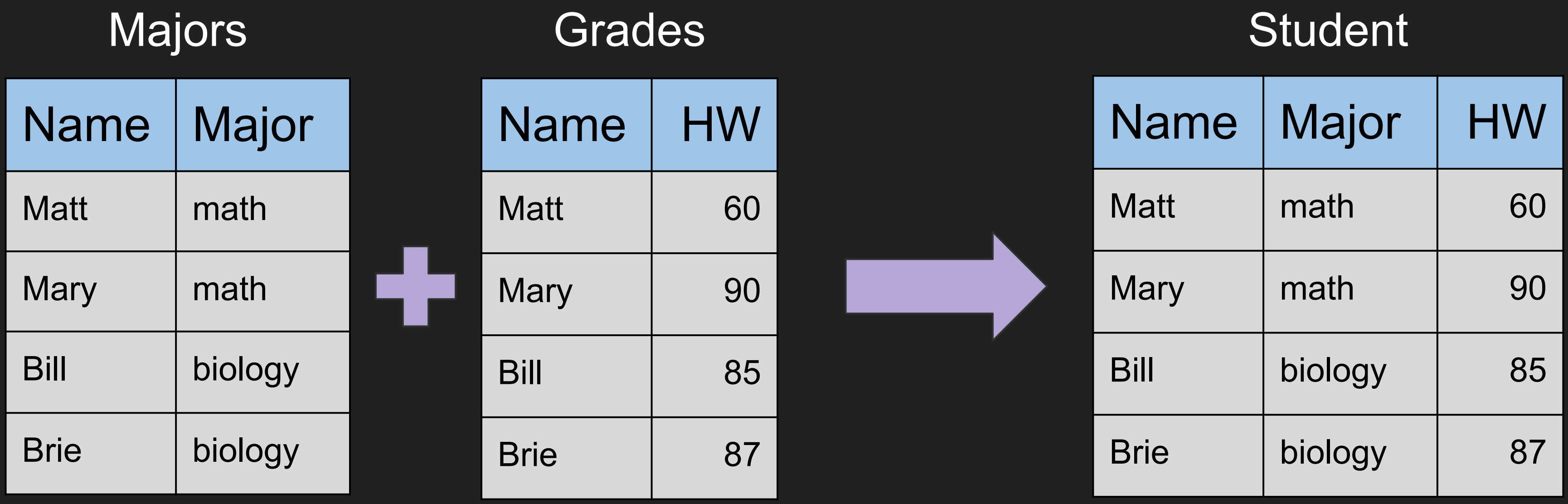
All values in the left dataset are kept

Only matching values in the right dataset are kept



left_join() - combine two datasets

Student <- left_join(Majors, Grades)



Majors

Name	Major
Matt	math
Mary	math
Bill	biology
Brie	biology

Grades

Name	HW
Matt	60
Mary	90
Bill	85
Brie	87

Student

Name	Major	HW
Matt	math	60
Mary	math	90
Bill	biology	85
Brie	biology	87

left_join() - combine two datasets

Student <- left_join(Majors, Grades)

Majors

Name	Major
Matt	math
Mary	math
Bill	biology
Brie	biology

+

Grades

Name	HW
Matt	60
Mary	90

→

Student

Name	Major	HW
Matt	math	60
Mary	math	90
Bill	biology	*NA*
Brie	biology	*NA*

left_join() - combine two datasets

Student <- left_join(Majors, Grades)

Majors

Name	Major
Matt	math
Mary	math

+

Grades

Name	HW
Matt	60
Mary	90
Bill	85
Brie	87

→

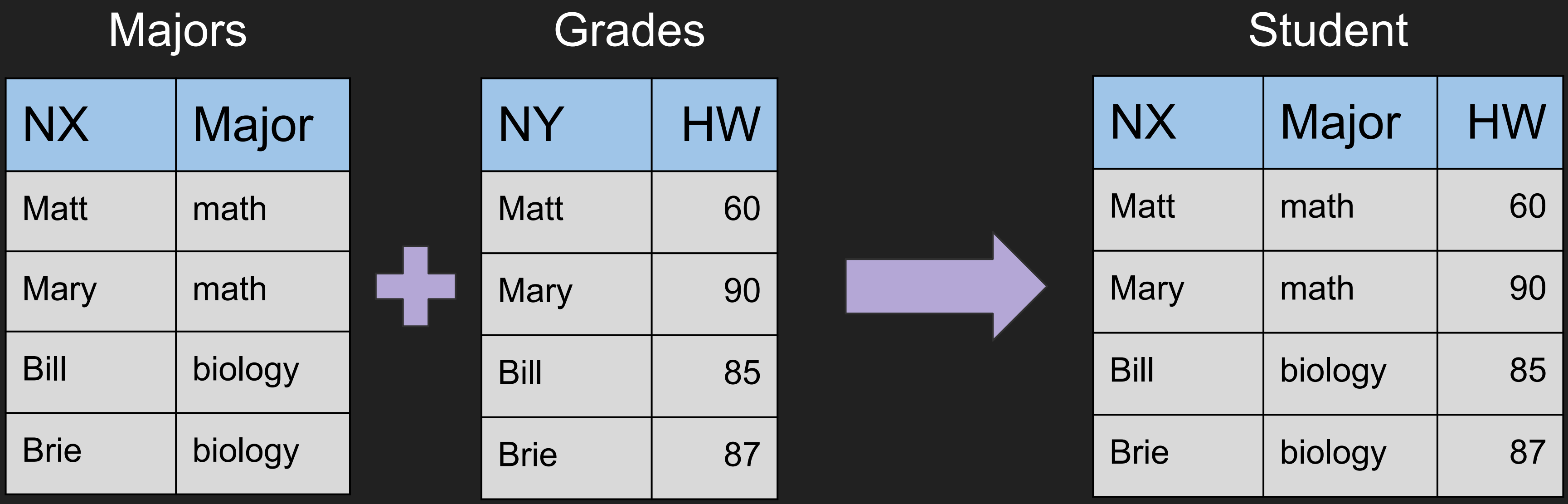
Student

Name	Major	HW
Matt	math	60
Mary	math	90

left_join() - combine two datasets

If matching columns have different names

Student <- left_join(Grades, Majors, by = c(“NX” = “NY”)



Majors

NX	Major
Matt	math
Mary	math
Bill	biology
Brie	biology

Grades

NY	HW
Matt	60
Mary	90
Bill	85
Brie	87

Student

NX	Major	HW
Matt	math	60
Mary	math	90
Bill	biology	85
Brie	biology	87

Other joins

Function	Description
left_join	Keep all left
right_join	Keep all right
inner_join	Match left and right
full_join	Keep all values
nest_join	Create list with match
anti_join	Keep non-matching

left_join() - Titanic Exercise

1. Split the titanic dataset accordingly:

```
> titanic1 <- titanic %>% select(1:3)
```

```
> titanic2 <- titanic %>% select(3:5)
```

2. Combine the datasets back together based on `Name`

```
> titanic3 <- left_join(_____, _____)
```

left_join() - Titanic Exercise

1. Split the titanic dataset accordingly:

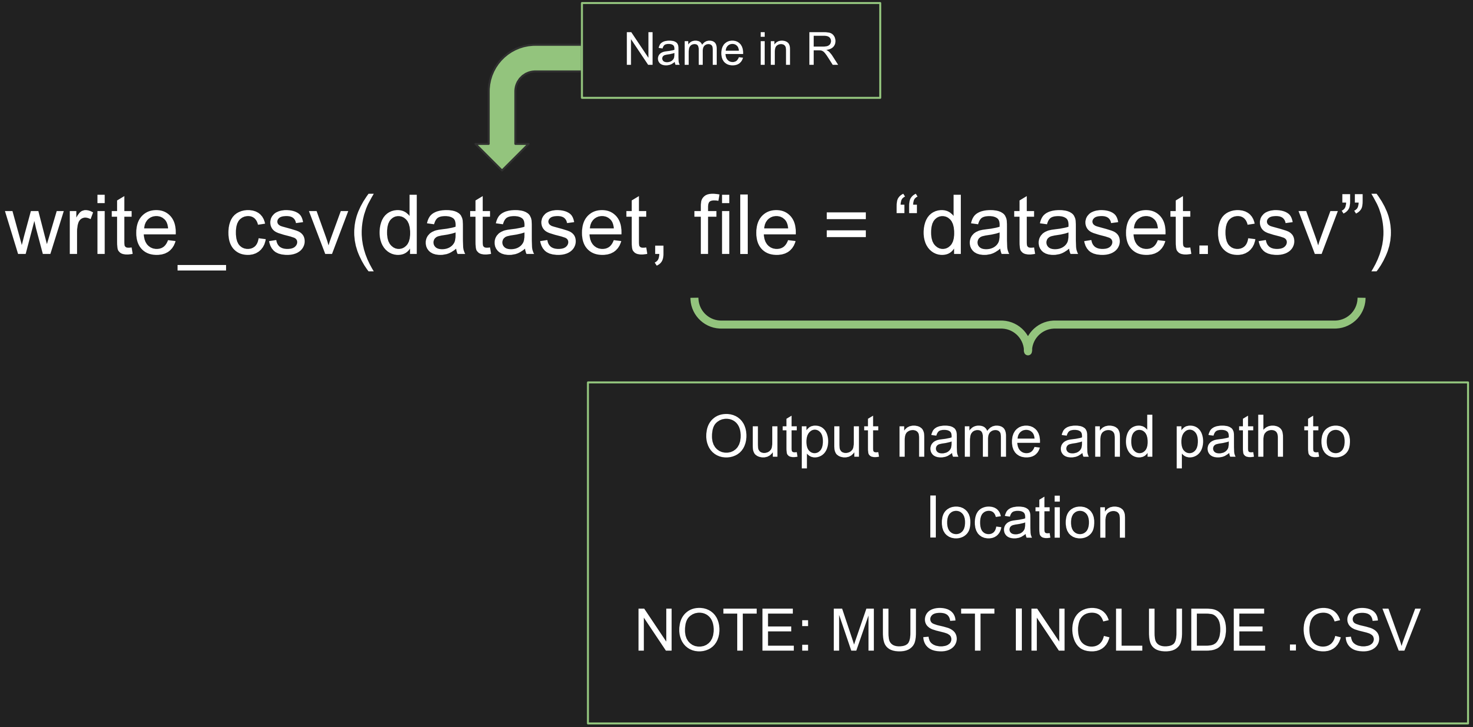
```
> titanic1 <- titanic %>% select(1:3)
```

```
> titanic2 <- titanic %>% select(3:5)
```

2. Combine the datasets back together based on `Name`

```
> titanic3 <- left_join(titanic1, titanic2)
```

Exporting Data



Name in R

```
write_csv(dataset, file = “dataset.csv”)
```

Output name and path to location

NOTE: MUST INCLUDE .CSV

Exporting Data - Titanic Exercises

1. Store the first four columns in `TitanicReduced`

```
> TitanicReduced <- titanic %>% select(____)
```

2. Export the previously created dataset

```
> write_csv(______, “RTitanic.csv”)
```

Exporting Data - Titanic Exercise Answers

1. Store the first four columns in `TitanicReduced`

```
> TitanicReduced <- titanic %>% select(1:4)
```

2. Export the previously created dataset

```
> write_csv(TitanicReduced, “RTitanic.csv”)
```

Additional Exercises - NYC Flights

Read in the NYC Flights Dataset into `nyflights`

```
> install.packages(“nycflights13”)

> library(nycflights13)

> nycflights <- flights # flights is the default dataset
```

1. What was the average `arr_delay` by `carrier`?

2. What was the longest `dep_delay` value?

3. How many times did planes depart early? (Negative `dep_delay` value)

4. Export the dataset with only JFK `origin`

Additional Exercises - NYC Flights Solutions

1. What was the average `arr_delay` by `carrier`?

```
> nycflights %>% group_by(carrier) %>% summarize(avg_delay = mean(arr_delay))
```

2. What was the longest `dep_delay` value?

```
> nycflights %>% summarize(long_delay = min(dep_delay))
```

3. How many times did planes depart early? (Negative `dep_delay` value)

```
> nycflights %>% filter(dep_delay < 0) %>% nrow()
```

4. Export the dataset with only JFK `origin`

```
> ny_jfk <- nycflights %>% filter(origin == “JFK”)  # Then use write_csv(ny_jfk,...)
```