Let's revisit our dplyr session:

The mean number of beds available by date:

```
beds_data %>%
  group_by(date) %>%
  summarise(
    mean_beds = mean(beds_av, na.rm = T))
```


Output = Object = New data frame!

Next step: Visualise

Piecing it together

```
beds_data %>%
  group_by(date) %>%
  summarise(
    mean_beds = mean(beds_av, na.rm = T))
```


Output = Object = New data frame!

```
ggplot(data =      )
```

Now, we could do this:

```
ggplot(data = beds_data %>%
           group_by(date) %>%
           summarise(
           mean_beds = mean(beds_av, na.rm = T))
         )+
geom_...
```

But, it's often better to:

Keep wrangling separate

And it's always better to:

Keep your code readable

as

as possible

Solution:

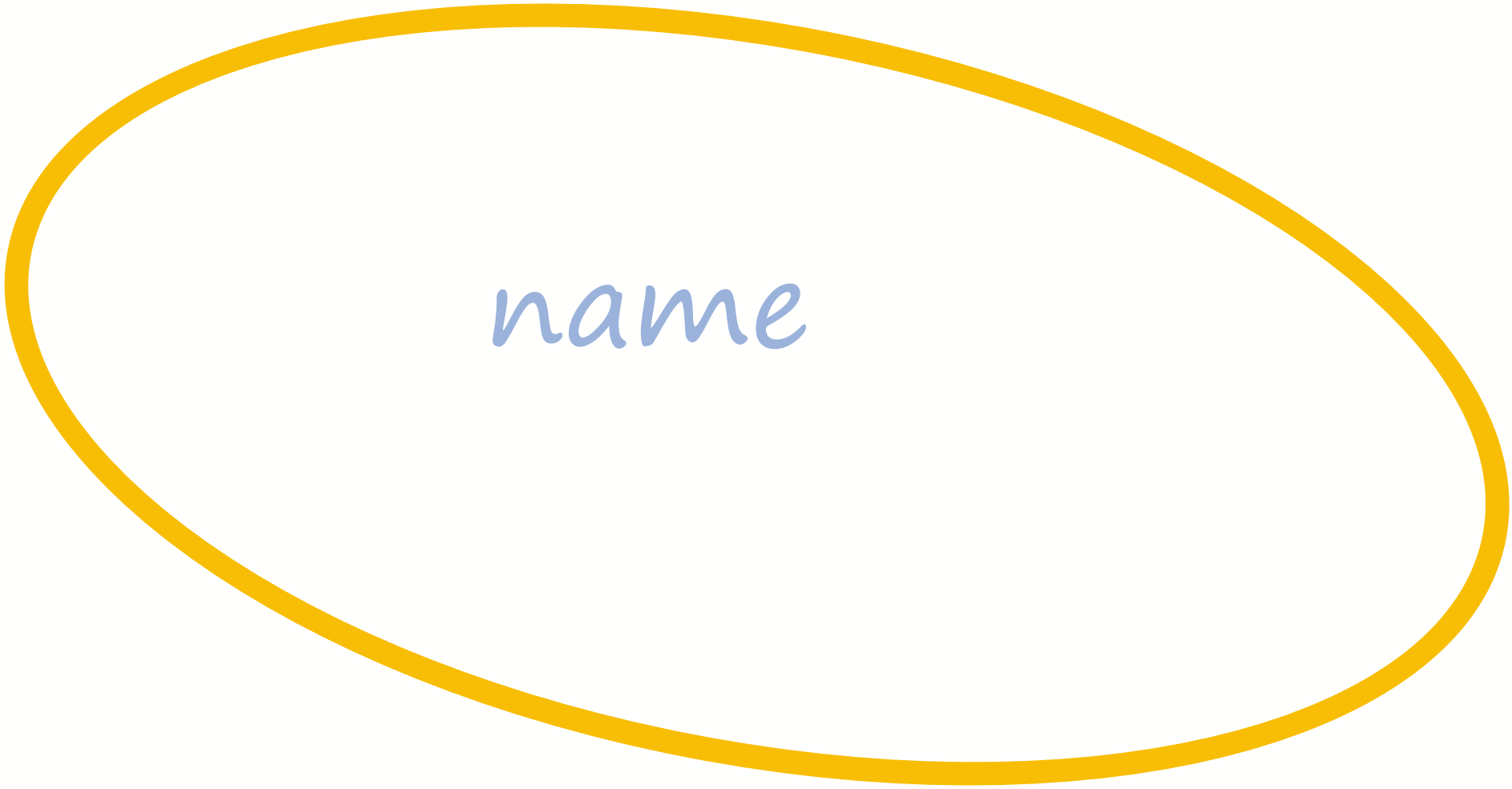
```
beds_data %>%
  group_by(date) %>%
  summarise(
   mean_beds = mean(beds_av, na.rm = T))
```

name

Replace this:

```
beds_data %>%
  group_by(date) %>%
  summarise(
   mean_beds = mean(beds_av, na.rm = T))
```

With this:



Naming

```
beds_data %>%
  group_by(date) %>%
  summarise(
   mean_beds = mean(beds_av, na.rm = T))
```

naming is hard

Naming

```
beds_data %>%
  group_by(date) %>%
  summarise(
   mean_beds = mean(beds_av, na.rm = T))
```

Murray

Naming

```
beds_data %>%
  group_by(date) %>%
  summarise(
   mean_beds = mean(beds_av, na.rm = T))
```

Murray

Good (object) names are:

1. Descriptive
2. Short(ish)
3. Consistent with other names

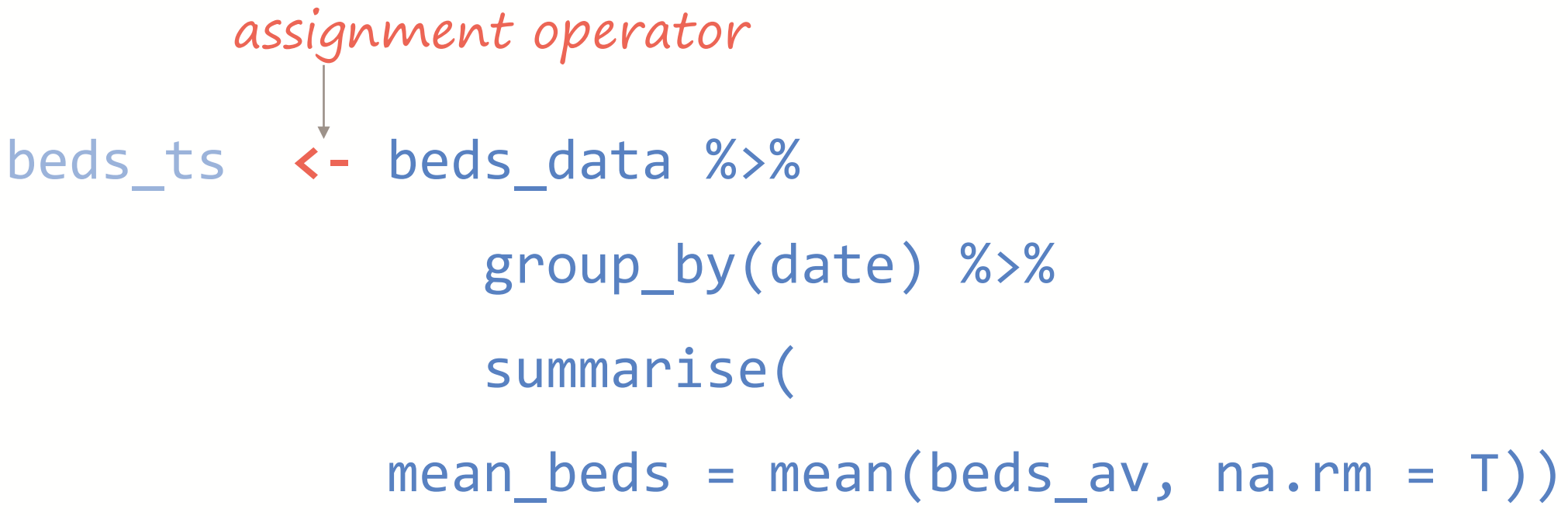
Naming?

```
beds_data %>%
  group_by(date) %>%
  summarise(
   mean_beds = mean(beds_av, na.rm = T))
```

beds_ts

Assignment

Name objects with the assignment operator:



Assignment operator

Shortcut: Alt -

Assignment

Assigns this name...

```
beds_ts  <- beds_data %>%
              group_by(date) %>%
              summarise(
            mean_beds = mean(beds_av, na.rm = T))
```

Assignment

Assigns this name...

To the object produced by this code

```
beds_ts  <- beds_data %>%
              group_by(date) %>%
              summarise(
            mean_beds = mean(beds_av, na.rm = T))
```

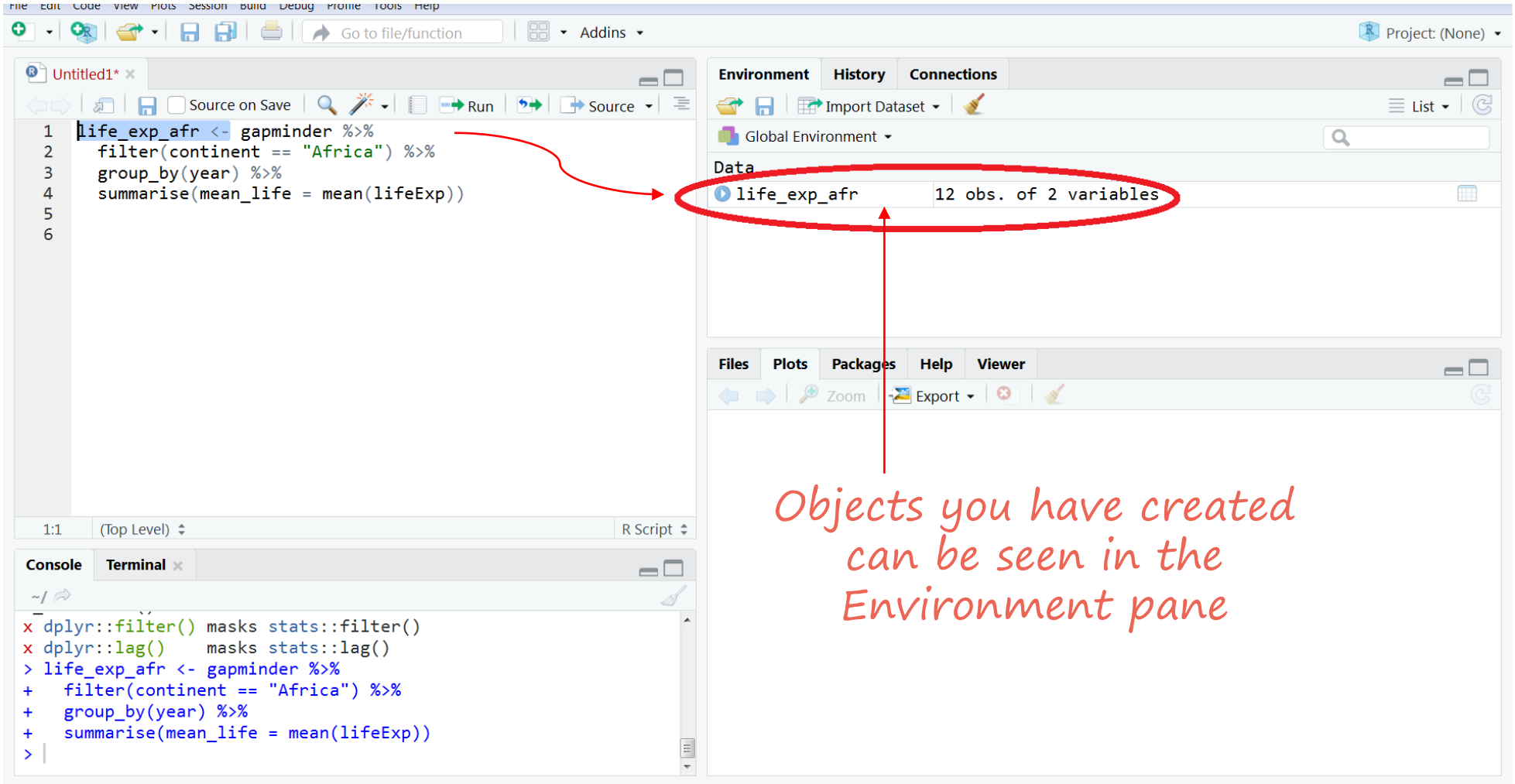
Assignment

```
beds_ts  <- beds_data %>%
              group_by(date) %>%
              summarise(
            mean_beds = mean(beds_av, na.rm = T))
```

```
beds_ts
```

running **beds_ts** recalls data frame above

Objects


Untitled1*
Source on Save
Run
Source
life_exp_afr <- gapminder %>%
filter(continent == "Africa") %>%
group_by(year) %>%
summarise(mean_life = mean(lifeExp))
Environment
History
Connections
Import Dataset
List
Global Environment
Data
life_exp_afr
12 obs. of 2 variables
Files
Plots
Packages
Help
Viewer
Zoom
Export
Console
Terminal
x dplyr::filter() masks stats::filter()
x dplyr::lag() masks stats::lag()
> life_exp_afr <- gapminder %>%
+ filter(continent == "Africa") %>%
+ group_by(year) %>%
+ summarise(mean_life = mean(lifeExp))
Objects you have created can be seen in the Environment pane

Returning to the plot:

```
ggplot(data = beds_data %>%
         group_by(date) %>%
         summarise(
         mean_beds = mean(beds_av, na.rm = T)))+
    geom_...(aes(x = …, y = …))
```

Becomes:

```
ggplot(data = beds_ts)+
  geom_...(aes(x = …, y = …))
```

Or:

```
ggplot(beds_ts)+
  geom_...(aes(…, …))
```

Your turn (1)

If you haven't already, follow the steps we've just covered to create the object **`beds_ts`**.

Your turn (2)

Plot the trend in mean available beds. You may require `ggplot(…)` plus `+` the following layers:

```
geom_line(aes(… , …))+
geom_point(…(… , …))+
ylim(… , …)
```

To get help with this (or any) function type `?ylim`

Your turn (3)

Give your plot object a name.

To summarise

Give an object (data frame, plot, value) a name when you may need that object later in your analysis.

This work is licensed as

End