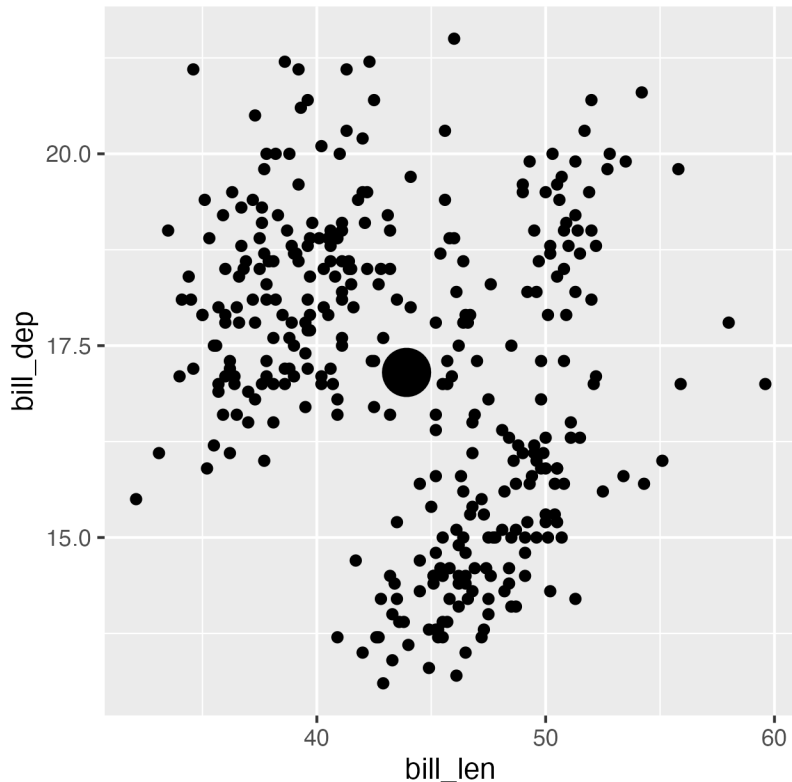




Where are we headed? Writing and using new function, `geom\_means()`

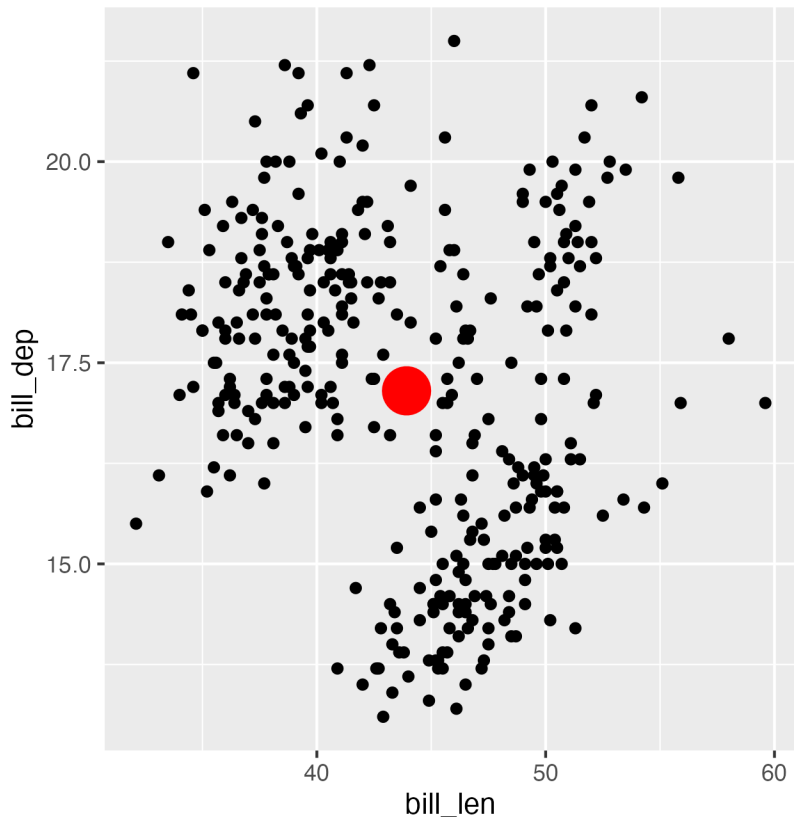
```
1 ggplot(penguins) +  
2   aes(bill_len, bill_dep) +  
3   geom_point() +  
4   geom_means(size = 8)
```





Step 0. Get the job done with 'base' ggplot2.

```
1 penguins_means <- penguins |>
2   summarise(
3     mean_bill_len = mean(bill_len
4     mean_bill_dep = mean(bill_dep
5   )
6
7 penguins |>
8   ggplot() +
9   aes(x = bill_len,
10      y = bill_dep) +
11   geom_point() +
12   geom_point(
13     data = penguins_means,
14     mapping = aes(
15       x = mean_bill_len,
16       y = mean_bill_dep
17     ),
18     color = "red",
19     size = 8)
```



Step 1. Define Compute. Test.

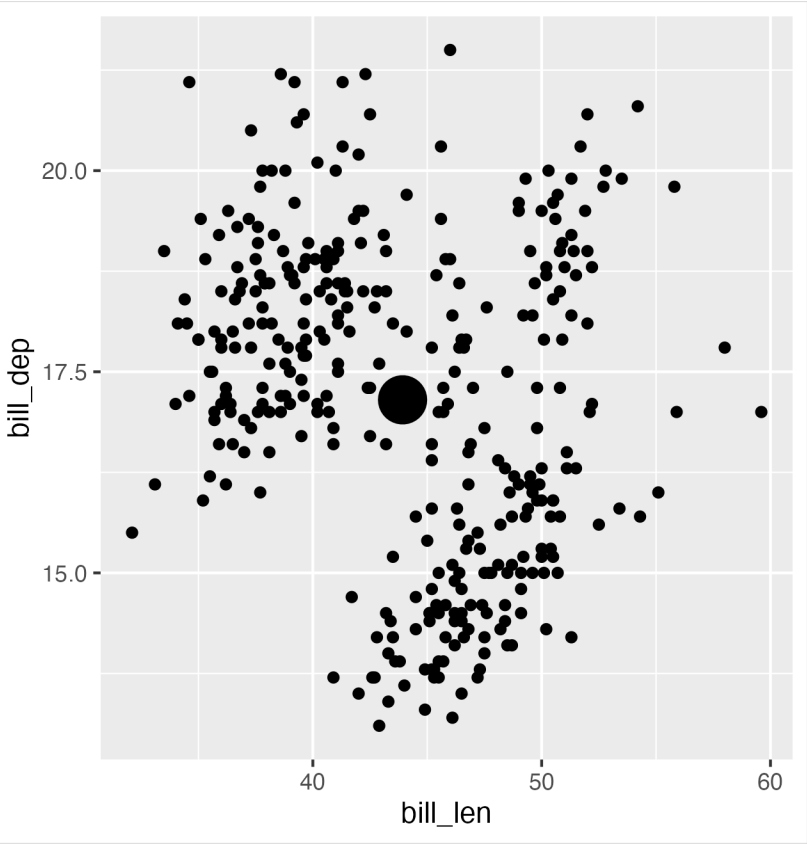
```
1 compute_group_means <-  
2   function(data, scales){  
3  
4     data |>  
5       summarise(  
6         x = mean(x, na.rm = T),  
7         y = mean(y, na.rm = T)  
8       )  
9  
10    }  
11  
12 penguins |>  
13   select(x = bill_len,  
14          y = bill_dep) |>  
15   compute_group_means()
```

x	y
43.92193	17.15117



Step 2. Define StatMeans. Test.

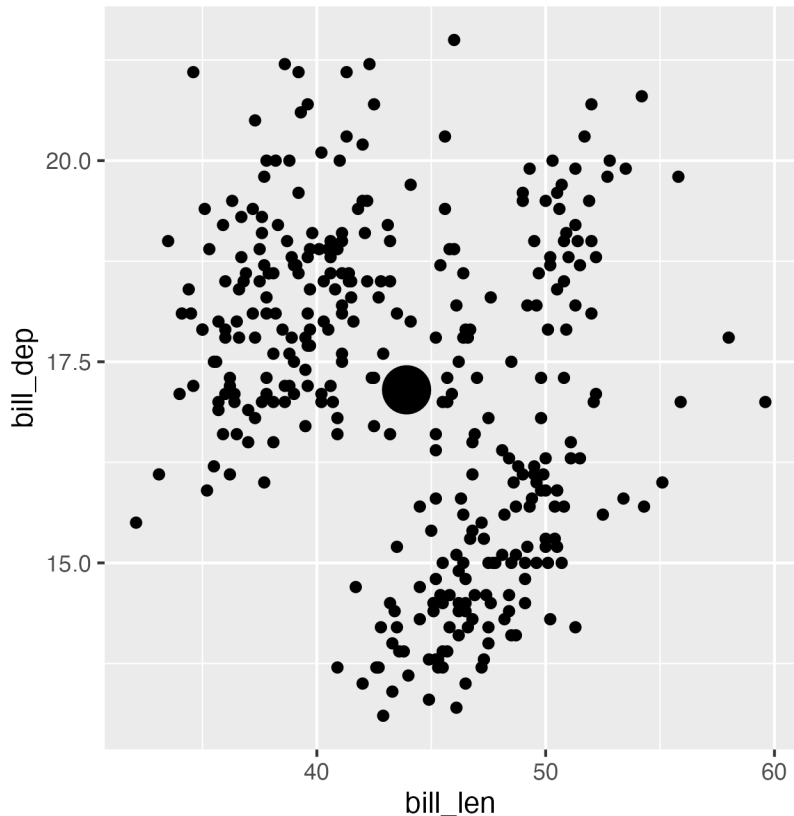
```
1 StatMeans <-  
2   ggproto(  
3     ` _class ` = "StatMeans",  
4     ` _inherit ` = Stat,  
5     compute_group =  
6       compute_group_means,  
7     required_aes = c("x", "y")  
8   )  
9  
10  
11 ggplot(penguins) +  
12   aes(x = bill_len,  
13     y = bill_dep) +  
14   geom_point() +  
15   geom_point(  
16     stat = StatMeans,  
17     size = 8  
18   )
```





Step 4. Define user-facing function. Enjoy!

```
1 geom_means <-  
2   make_constructor( # >v4.0 ggplot2  
3     x = GeomPoint,  
4     stat = "means"  
5   )  
6  
7  
8 ggplot(penguins) +  
9   aes(bill_len, bill_dep) +  
10  geom_point() +  
11  geom_means(size = 8)
```





Step 4.b Test out group-wise compute (for free!)

```
1 ggplot(penguins) +  
2   aes(bill_len, bill_dep) +  
3   geom_point() +  
4   geom_means(size = 8) +  
5   aes(color = species)
```

