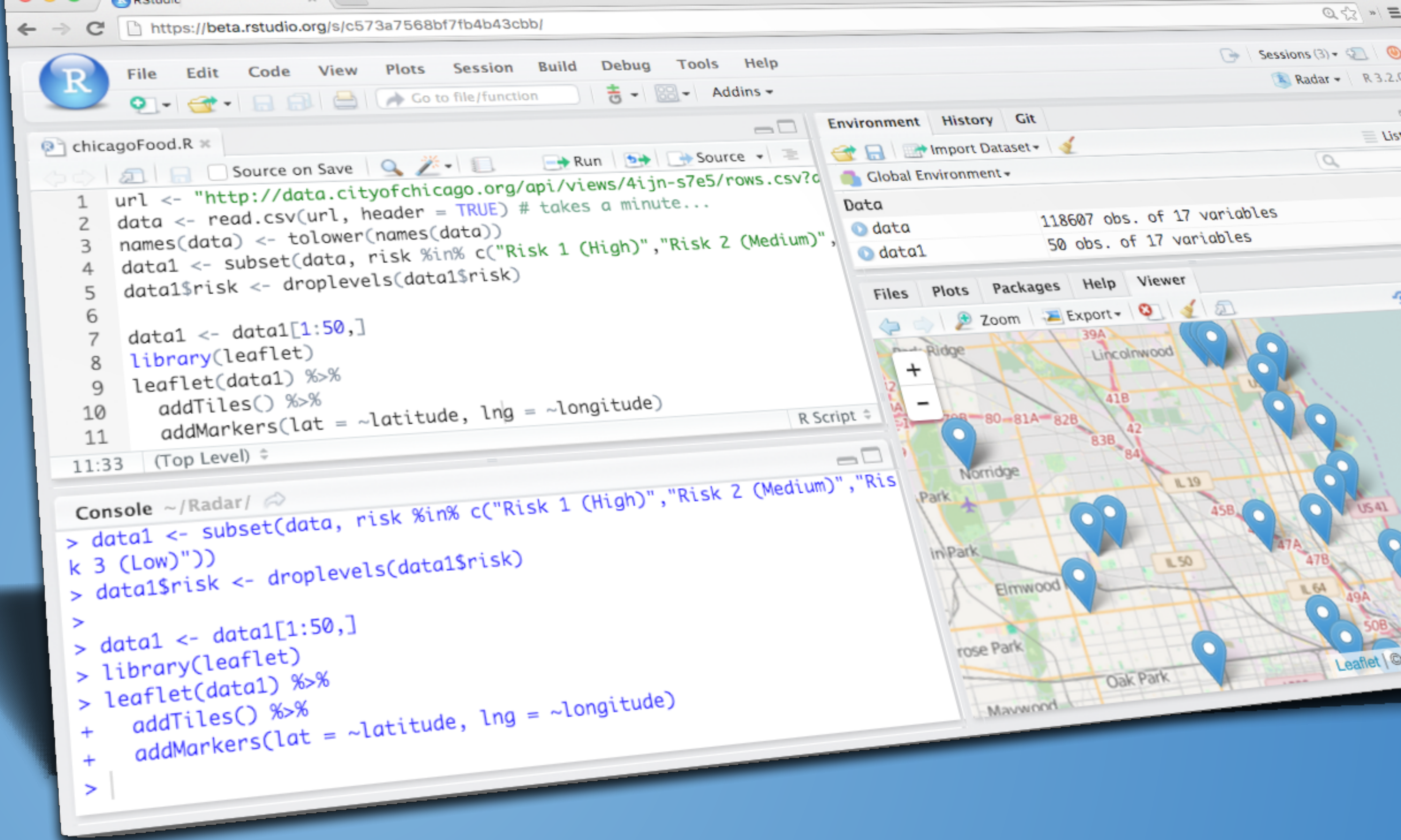
RStudio®



```
url <- "http://data.cityofchicago.org/api/views/4ijn-s7e5/rows.csv?a
data <- read.csv(url, header = TRUE) # takes a minute...
names(data) <- tolower(names(data))
data1 <- subset(data, risk %in% c("Risk 1 (High)","Risk 2 (Medium)",
data1$risk <- droplevels(data1$risk)

data1 <- data1[1:50,]
library(leaflet)
leaflet(data1) %>%
  addTiles() %>%
  addMarkers(lat = ~latitude, lng = ~longitude)
```

```
Data
data      118607 obs. of 17 variables
data1     50 obs. of 17 variables
```

Console ~/Radar/

```
> data1 <- subset(data, risk %in% c("Risk 1 (High)","Risk 2 (Medium)","Ris
k 3 (Low)"))
> data1$risk <- droplevels(data1$risk)
>
> data1 <- data1[1:50,]
> library(leaflet)
> leaflet(data1) %>%
+   addTiles() %>%
+   addMarkers(lat = ~latitude, lng = ~longitude)
>
```

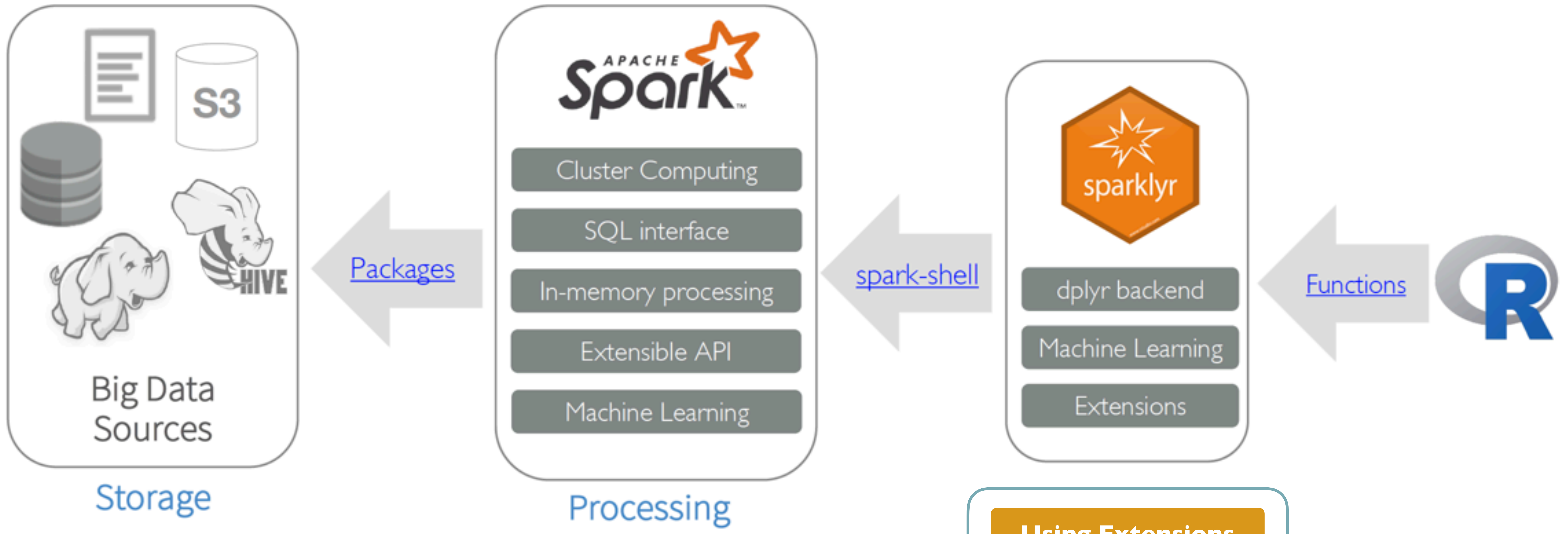
Advanced sparklyr features

Advanced sparklyr Features

- ✓ Overview
- ✓ Scaling R in Spark with spark_apply()
- ✓ Spark from your desktop using Livy
- ✓ Spark Applications with Shiny
- ✓ Questions

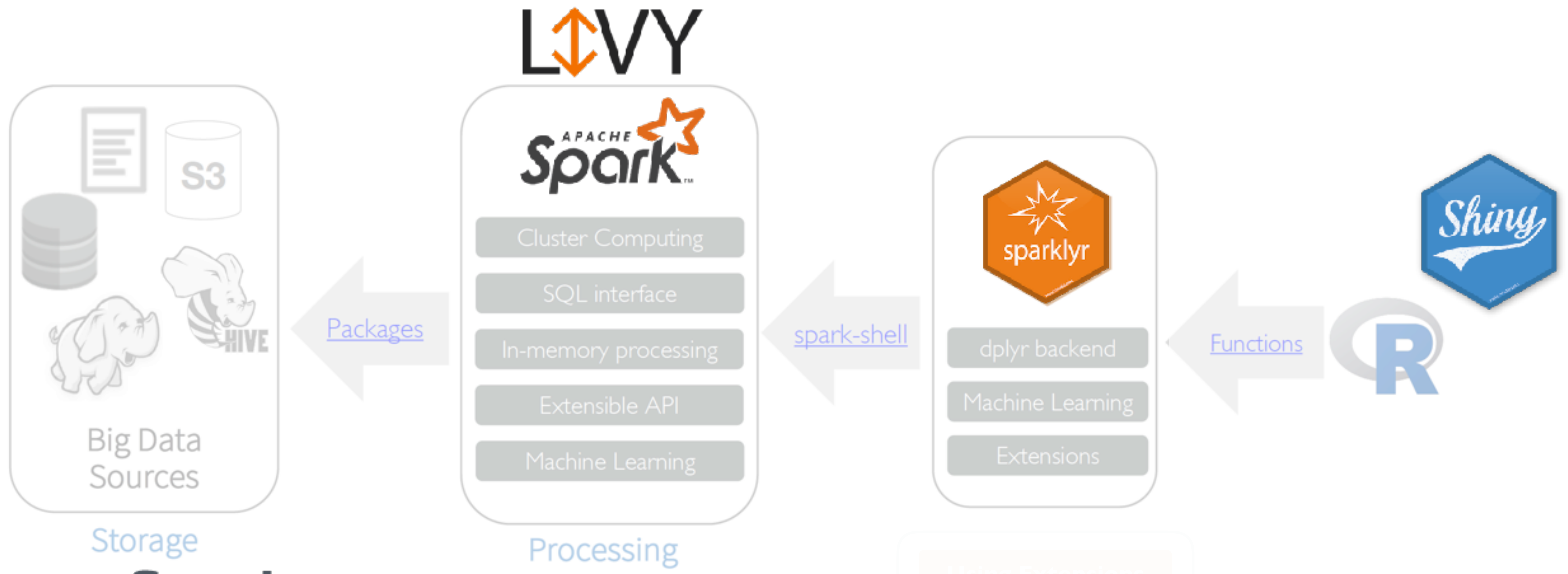
Overview

OVERVIEW



- Using Extensions
- Using R Code
- Using Scala Code
- Creating R Packages

OVERVIEW



Common Crawl

240 TiB

72K Files

Spark R
Spark R
Spark R
Spark R
Spark R
Spark R

spark_apply()

- ✓ Scaling R in Spark with spark_apply()
- ✓ Spark from your desktop using Livy
- ✓ Spark Applications with Shiny

Scaling R

Use Cases

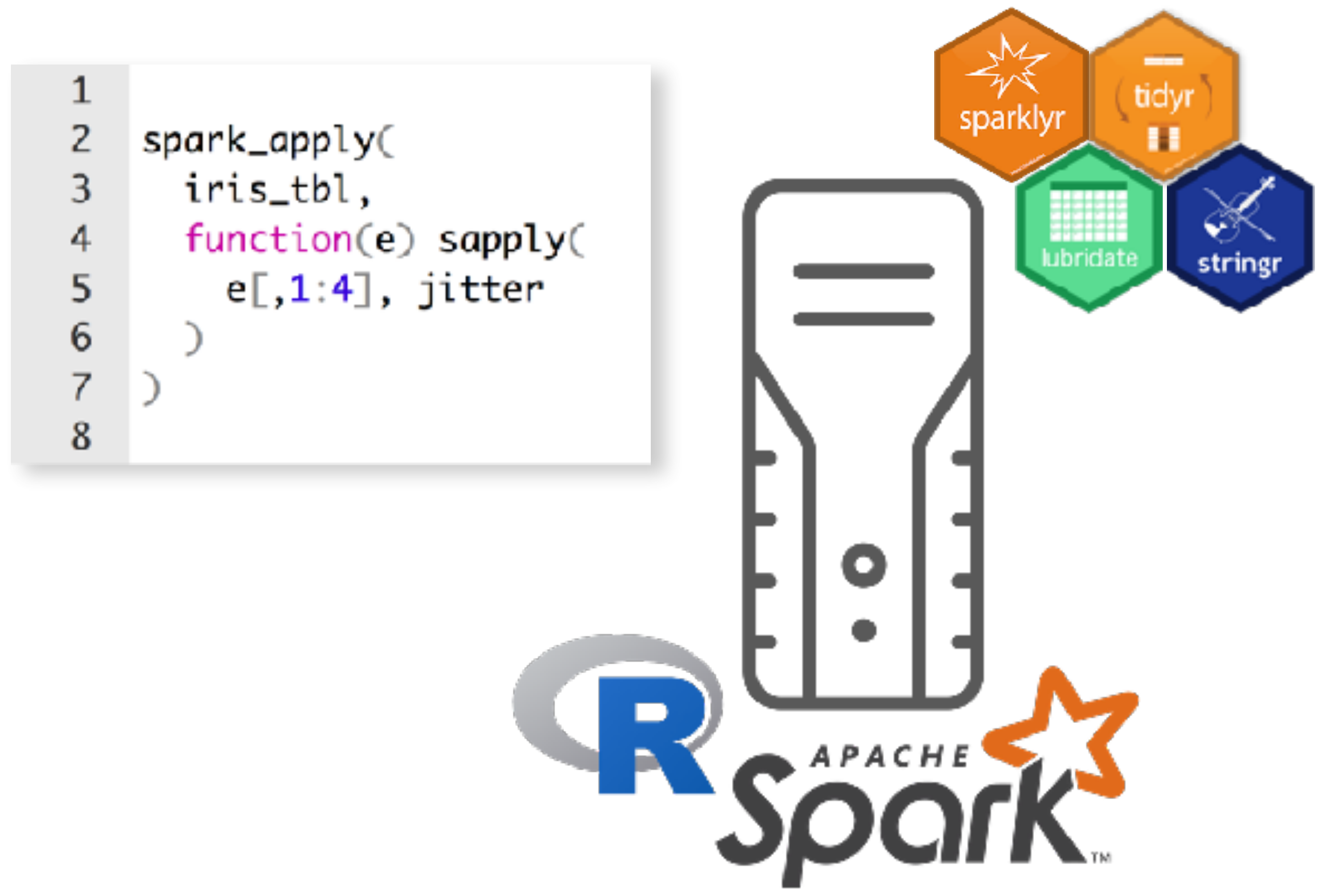
✓ Leverage R skills

```r

spark_apply(iris_tbl, function(e) sapply(e[,1:4], jitter))

```

✓ Complement R

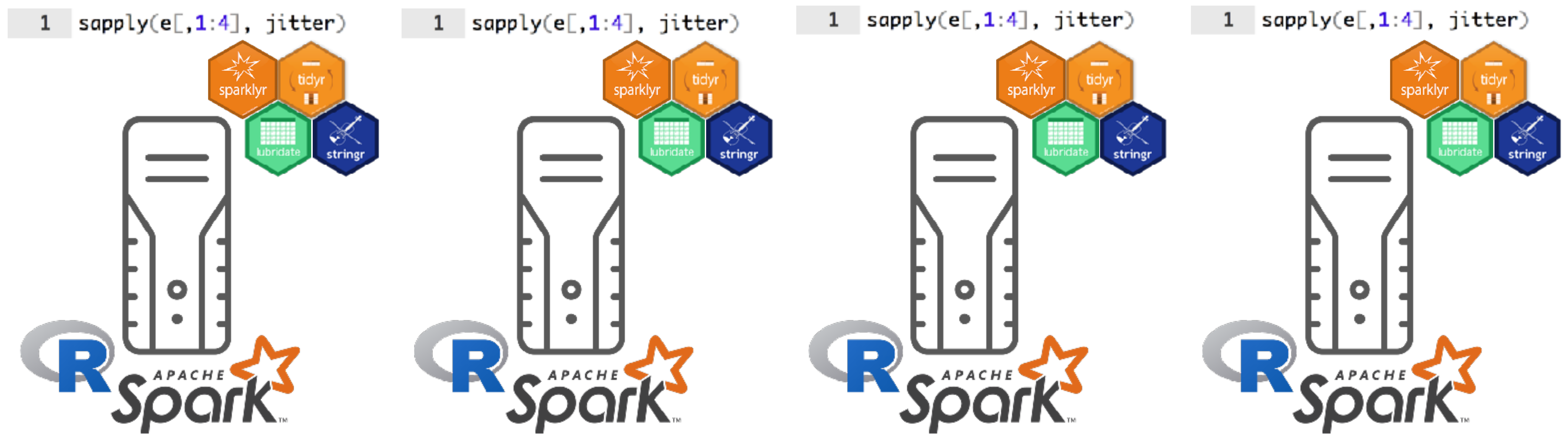
```r

spark_apply(
  iris_tbl,
  function(e) broom::tidy(lm(Petal_Width ~ Petal_Length, e)),
  colums = c("term", "estimate", "std.error", "statistic", "p.value"),
  group_by = "Species"
)

```

USING SPARK_APPLY()



```
spark_apply(
  iris_tbl,
  function(e) sapply(
    e[,1:4], jitter
  )
)
```

- ✓ Install R in every node, once per cluster.
- ✓ spark_apply() distributes packages, once per session.
- ✓ spark_apply() distributes your code, once per call.



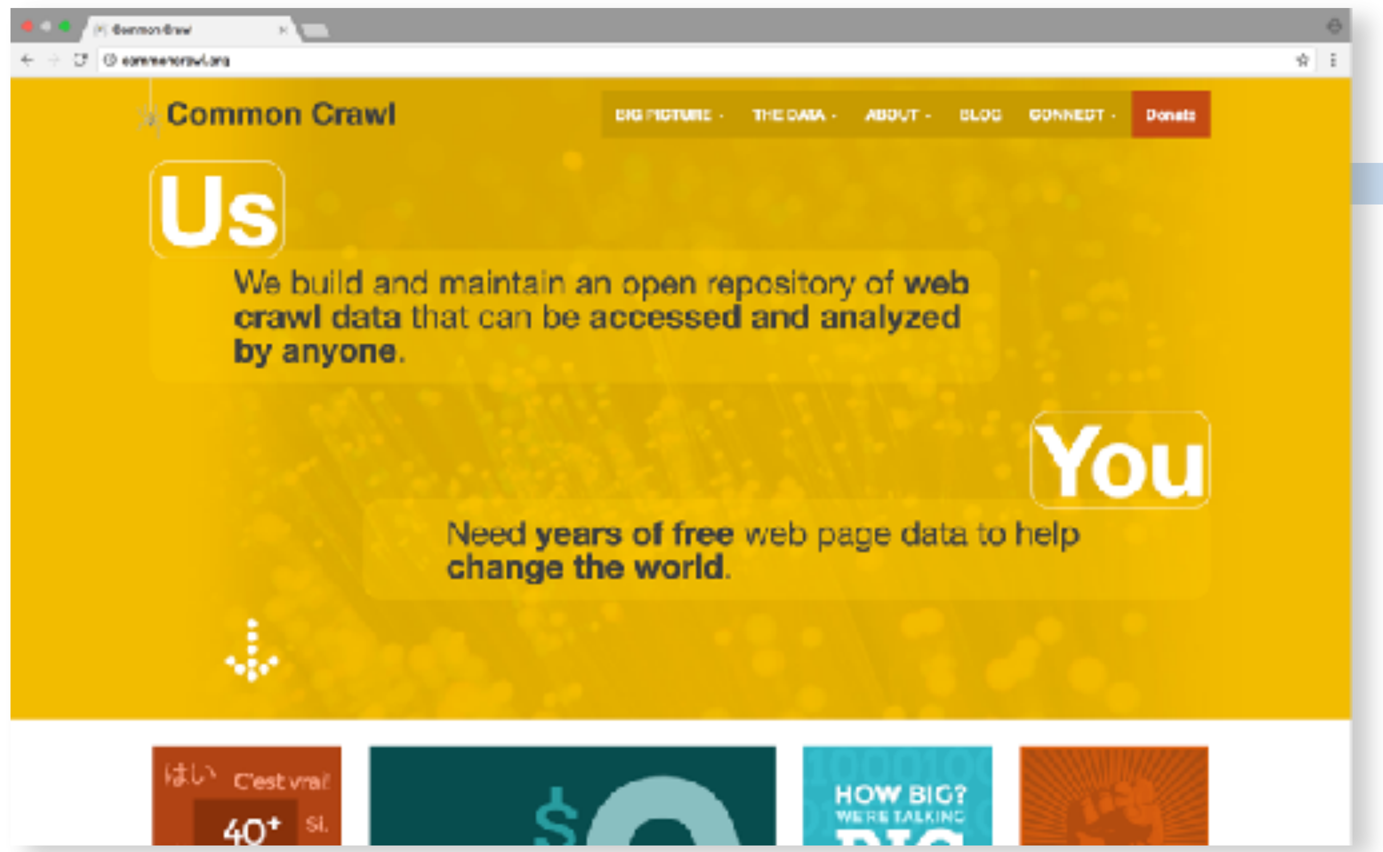
```
sapply(e[,1:4], jitter)
```

```
sapply(e[,1:4], jitter)
```

```
sapply(e[,1:4], jitter)
```

```
sapply(e[,1:4], jitter)
```

Parsing and Filtering the CommonCrawl



Us

We build and maintain an open repository of web crawl data that can be accessed and analyzed by anyone.

You

Need years of free web page data to help change the world.

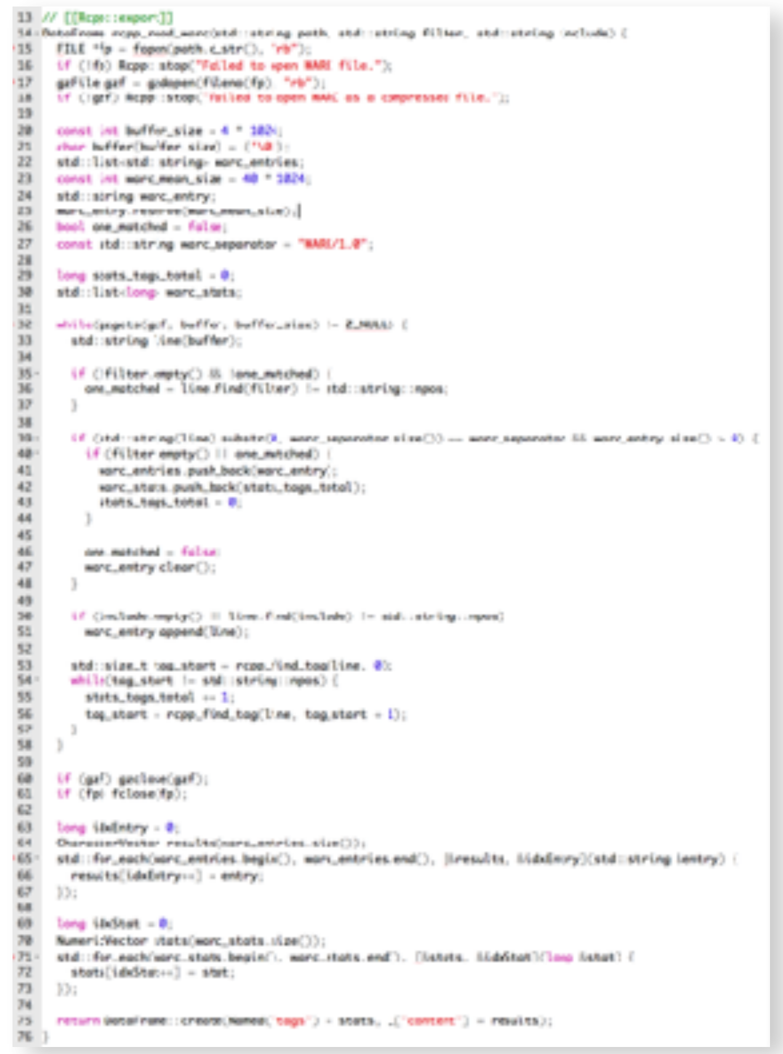
240 TiB

72K Files

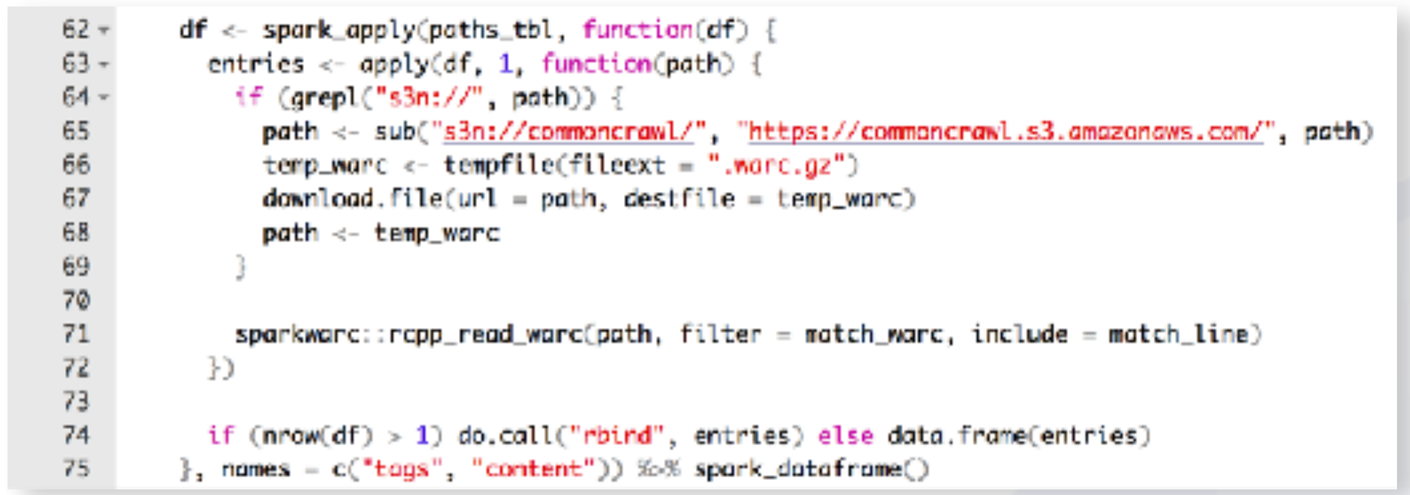
```r
library(sparkwarc)
spark_read_warc(sc, "warcs", cc_warc(1, 100))
```

sparkwarc uses

Rcpp:: + spark_apply()



```cpp
// [[Rcpp::export]]
DataFrame rcpp_read_warc(std::string path, std::string filter, std::string include) {
  FILE *fp = fopen(path.c_str(), "rb");
  if (!fp) Rcpp::stop("Failed to open WARC file.");
  gzFile gzf = gzdopen(fileno(fp), "rb");
  if (!gzf) Rcpp::stop("Failed to open WARC as a compressed file.");

  const int buffer_size = 4 * 1024;
  char buffer[buffer_size] = {'\0'};
  std::list<std::string> warc_entries;
  const int warc_mean_size = 40 * 1024;
  std::string warc_entry;
  warc_entry.reserve(warc_mean_size);
  bool one_matched = false;
  const std::string warc_separator = "WARC/1.0";

  long stats_tags_total = 0;
  std::list<long> warc_stats;

  while(gzgets(gzf, buffer, buffer_size) != Z_NULL) {
    std::string line(buffer);

    if (!filter.empty() && !one_matched) {
      one_matched = line.find(filter) != std::string::npos;
    }

    if (std::string(line).substr(0, warc_separator.size()) == warc_separator && warc_entry.size() > 0) {
      if (filter.empty() || one_matched) {
        warc_entries.push_back(warc_entry);
        warc_stats.push_back(stats_tags_total);
        stats_tags_total = 0;
      }

      one_matched = false;
      warc_entry.clear();
    }

    if (include.empty() || line.find(include) != std::string::npos)
      warc_entry.append(line);

    std::size_t tag_start = rcpp_find_tag(line, 0);
    while(tag_start != std::string::npos) {
      stats_tags_total += 1;
      tag_start = rcpp_find_tag(line, tag_start + 1);
    }
  }

  if (gzf) gzclose(gzf);
  if (fp) fclose(fp);

  long idxEntry = 0;
  CharacterVector results(warc_entries.size());
  std::for_each(warc_entries.begin(), warc_entries.end(), [&results, &idxEntry](std::string &entry) {
    results[idxEntry++] = entry;
  });

  long idxStat = 0;
  NumericVector stats(warc_stats.size());
  std::for_each(warc_stats.begin(), warc_stats.end(), [&stats, &idxStat](long &stat) {
    stats[idxStat++] = stat;
  });

  return DataFrame::create(Named("tags") = stats, _["content"] = results);
}
```



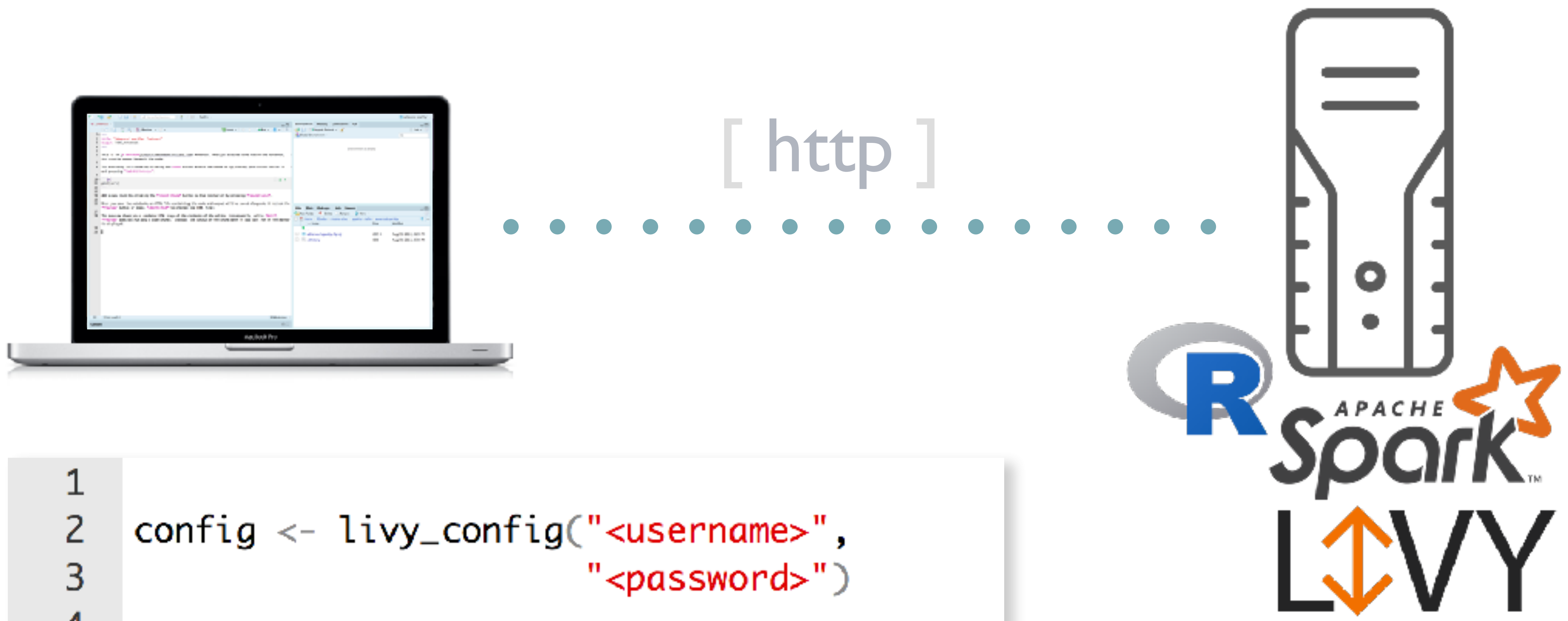
```r
  df <- spark_apply(paths_tbl, function(df) {
    entries <- apply(df, 1, function(path) {
      if (grepl("s3n://", path)) {
        path <- sub("s3n://commoncrawl/", "https://commoncrawl.s3.amazonaws.com/", path)
        temp_warc <- tempfile(fileext = ".warc.gz")
        download.file(url = path, destfile = temp_warc)
        path <- temp_warc
      }

      sparkwarc::rcpp_read_warc(path, filter = match_warc, include = match_line)
    })

    if (nrow(df) > 1) do.call("rbind", entries) else data.frame(entries)
  }, names = c("tags", "content")) %>% spark_dataframe()
```

github.com/javierluraschi/sparkwarc

Livy

Using Livy

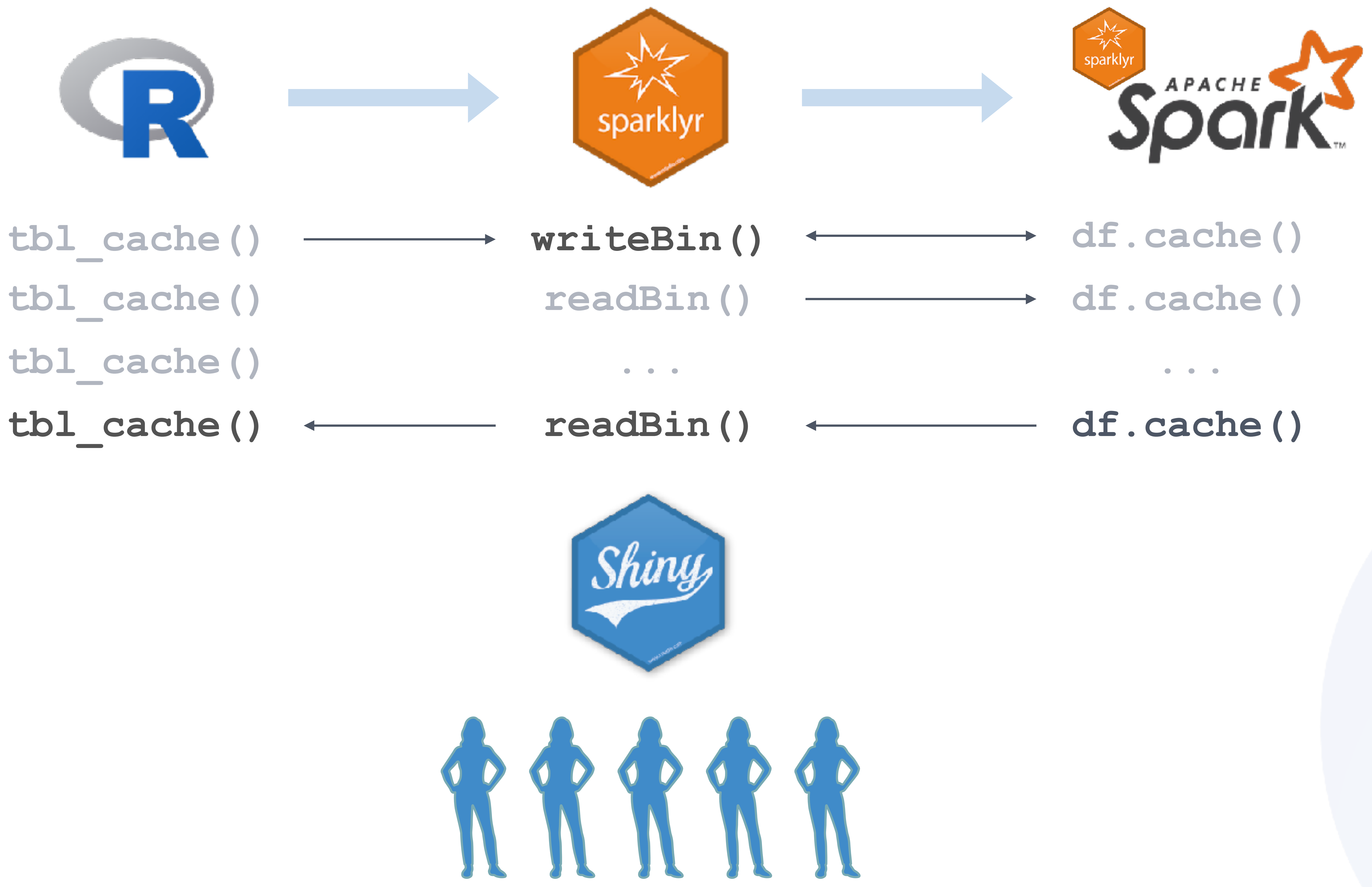
✓ LIVY installed in one node



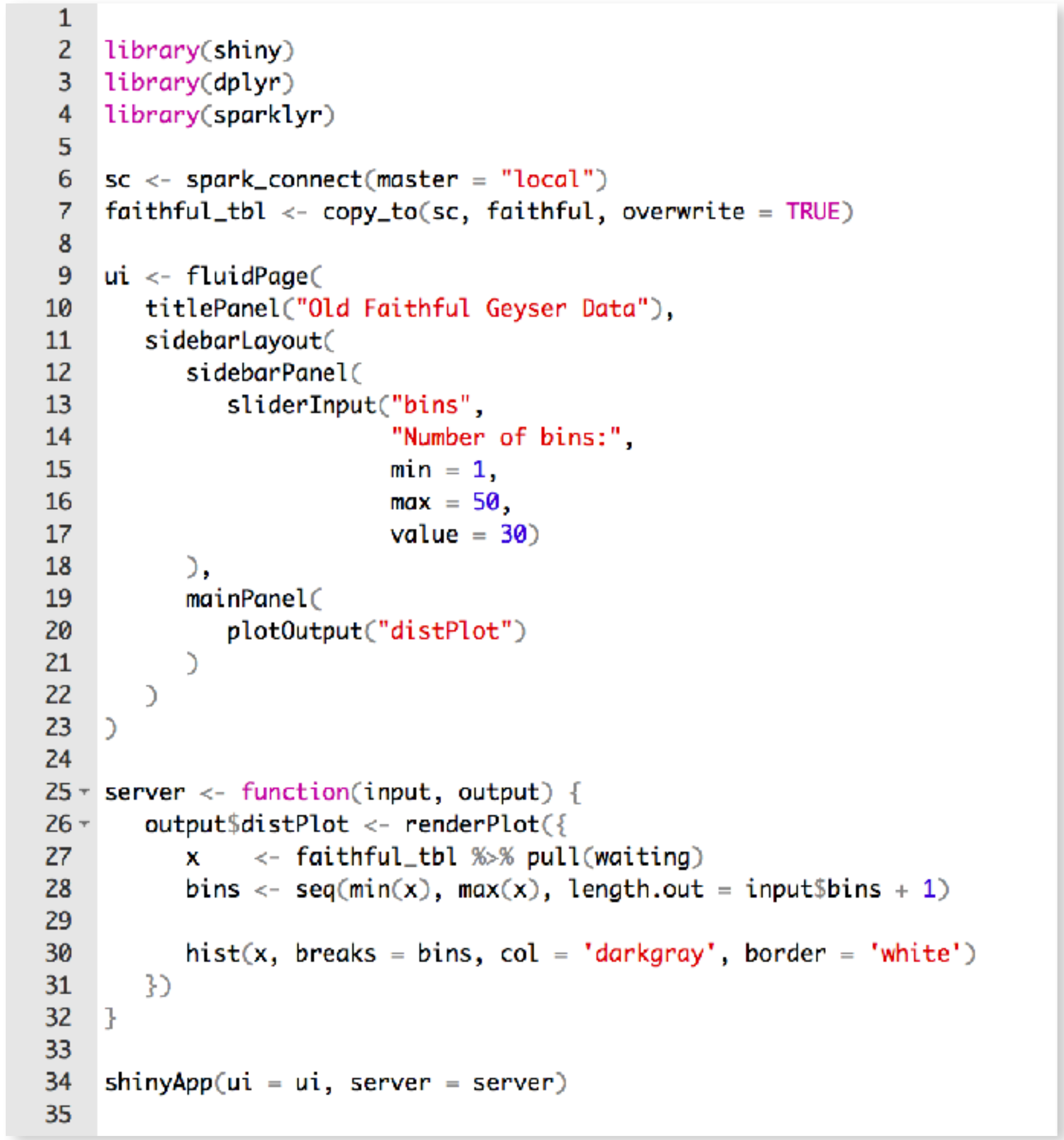
```r
config <- livy_config("<username>",
                      "<password>")

sc <- spark_connect(master = "<address>",
                    method = "livy",
                    config = config)
```

Shiny

SPARKLYR CONNECTIONS

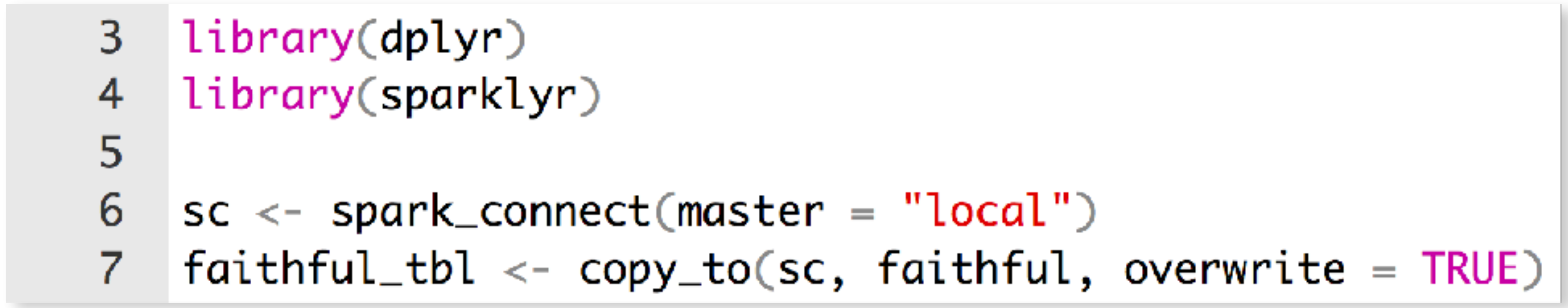

sparklyr
APACHE Spark
tbl_cache() → writeBin() ↔ df.cache()
tbl_cache() readBin() → df.cache()
tbl_cache() ... ...
tbl_cache() ← readBin() ← df.cache()
Shiny

SHINY AND SPARKLYR



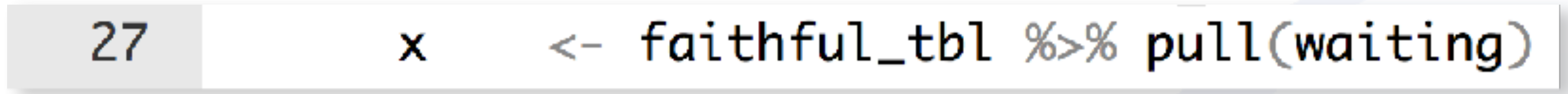
```r
 1
 2  library(shiny)
 3  library(dplyr)
 4  library(sparklyr)
 5
 6  sc <- spark_connect(master = "local")
 7  faithful_tbl <- copy_to(sc, faithful, overwrite = TRUE)
 8
 9  ui <- fluidPage(
10     titlePanel("Old Faithful Geyser Data"),
11     sidebarLayout(
12        sidebarPanel(
13           sliderInput("bins",
14                       "Number of bins:",
15                       min = 1,
16                       max = 50,
17                       value = 30)
18        ),
19        mainPanel(
20           plotOutput("distPlot")
21        )
22     )
23  )
24
25  server <- function(input, output) {
26     output$distPlot <- renderPlot({
27        x    <- faithful_tbl %>% pull(waiting)
28        bins <- seq(min(x), max(x), length.out = input$bins + 1)
29
30        hist(x, breaks = bins, col = 'darkgray', border = 'white')
31     })
32  }
33
34  shinyApp(ui = ui, server = server)
35
```

Setup



```r
3  library(dplyr)
4  library(sparklyr)
5
6  sc <- spark_connect(master = "local")
7  faithful_tbl <- copy_to(sc, faithful, overwrite = TRUE)
```

Server



```r
27        x    <- faithful_tbl %>% pull(waiting)
```

Useful Websites

RStudio and Spark:
spark.rstudio.com

Shiny:
shiny.rstudio.com

Apache Livy:
livy.incubator.apache.org/

Sparklyr Github:
github.com/rstudio/sparklyr

Questions

RStudio®

Open Source & Free

Desktop: http://www.rstudio.com/products/rstudio/download/

RStudio Server: http://www.rstudio.com/products/rstudio/download-server/

Shiny Server: http://www.rstudio.com/products/shiny/download-server/

shinyapps.io beta: https://www.shinyapps.io/admin/#/signup

45 Day Evaluation of Pro Products

RStudio Server Pro: http://www.rstudio.com/products/rstudio-server-pro/evaluation/

Shiny Server Pro: http://www.rstudio.com/products/shiny-server-pro/evaluation/

Please stay in touch

- Blog - http://rviews.rstudio.com/
- Blog - http://blog.rstudio.org/
- Twitter - @rstudio #rstats http://twitter.com/rstudio/
- GitHub - https://github.com/rstudio/
- LinkedIn - https://linkedin.com/company/rstudio-inc
- Facebook - https://www.facebook.com/pages/RStudio-inc
- Google+ - https://plus.google.com/110704473211154995841/posts