Data Science Workshop

Safouene Kallel

2023-12-07

Quizz

SQL is a ... language.

... clause is used to choose the columns

... clause is used to filter the rows

... gets rid of the duplicates

... is used to create aggregated summaries

A Primary key is ...

Quizz

Please complete the Query:

```
SELECT Sales.*, Customers.*
FROM Sales
LEFT JOIN Customers
... Customers.CustomerID = Sales.CustomerID
```

Quizz

{...} helps us connect to SQL
{...} helps us handle data
{...} helps us visualize data

```
pulsedIV <- pulsedIV ...
  filter(Example==1)
```

Quizz

```r
ggplot(pulsedIV,aes(x=V2,
                    y=I,
                    ... = V1)) +
  geom_point() +
  geom_line()
```

Quizz

```r
pulsedIV_summary <- pulsedIV %>%
  ...(V1) %>%
  ...(maxI=max(I, na.rm = TRUE))
```

Quizz

What is the difference between numpy and pandas?

Which libraries did we use to connect to SQL?

Which libraries did we use to make a plot?

Please finish the command

```
df1952 = ...
```

Summary

We have seen SQL (a neat way to store data but not only).

We have seen 2 open source programming languages:

- We have seen R, packages, SQL connection, pipe, ggplot
- We have seen Python, libraries, SQL connection, seaborn

We add another building block today.

Web Applications

What we are going to create today:

- Create a Web Application that displays the PCM data.
- Select one wafer from the list
- Make a button to apply log scale.

Let's look at the data

```r
library(readr)

PCM <- read_csv("./Data/PCM2.csv")

View(PCM)
```

Let's visualize the data

Please filter the data according to this rule:

Wafer = 1

```
PCM_example <- PCM %>%
  filter(Wafer == 1)
```

Is this scalable?

```r
ggplot(PCM_example,
       aes(x=X_COORD,
           y=Y_COORD,
           fill= value)) +
  geom_tile() +
  scale_fill_gradient(low="red",high="green")
```

Is this scalable?

What if I want to see another wafer? We can filter the data again, and visualize it.

```
PCM_example <- PCM %>%
  filter(Wafer == 2)
```

We can apply a for loop that goes through the waferIDs one by one and generate .png charts for each one.

Is this scalable?

or we can use this code:

```
WaferList <- c(1,2)
PCM_example <- PCM %>%
  filter(Wafer %in% WaferList)

ggplot(PCM_example, aes(x=X_COORD,
                        y=Y_COORD,
                        group= Wafer,
                        fill= value)) +
  geom_tile() +
  scale_fill_gradient(low="red",high="green") +
  facet_grid(.~Wafer)
```

Is this scalable?

What if we have new data coming from additional tests?

We need to log in, generate the new plots...

We need something else..

A web Application that is running 24/7. It checks for new data every time we launch it.

Welcome to Shiny

Figure 1: https://shiny.rstudio.com

Let's start

https://shiny.rstudio.com/gallery/widget-gallery.html

We need a selection button and a check box for log10 graphs.

Let’s develop a PCM app

Loading the usual suspects

```r
library("readr")
library("dplyr")
library('ggplot2')
library('shiny')
```

Loading the data

```r
PCM <- read_csv('./Data/PCM2.csv')
```

Let’s develop a PCM app

```r
WaferList <- c(1,2)
PCM_example <- PCM %>%
  filter(Wafer %in% WaferList)

ggplot(PCM_example, aes(x=X_COORD,
                        y=Y_COORD,
                        group= Wafer,
                        fill= value)) +
  geom_tile() +
  scale_fill_gradient(low="red",high="green") +
  facet_grid(.~Wafer)
```

Let's develop the ui side

the UI side

```
ui <- fluidPage(
  selectInput("Wafer", label = h3("Select Wafer"),
              sort(WaferList)),
  plotOutput('plot')
)
```

Let’s develop the server side

the Server side

```r
server <- function(input, output) {
  WaferList <- c(1,2)
  PCM_example <- PCM %>%
    filter(Wafer %in% WaferList)

  plot <- ggplot(PCM_example, aes(x=X_COORD,
                                  y=Y_COORD,
                                  group=Wafer,
                                  fill= value)) +
    geom_tile() +
    scale_fill_gradient(low="red",high="green") +
    facet_grid(.~Wafer)
}
```

Let’s develop the server side

the Server side

```r
server <- function(input, output) {
  PCM_example <- reactive({PCM %>%
      filter(Wafer %in% WaferList)
  })

  output$plot <- renderPlot({ggplot(PCM_example(), aes(x=X_COO
                             y=Y_COORD,
                             group=Wafer,
                             fill= value)) +
    geom_tile() +
    scale_fill_gradient(low="red",high="green") +
    facet_grid(.~Wafer)
  })
}
```

Let's call the app

```r
shinyApp(ui = ui, server = server)
```

Let's continue working on the Server side

```r
server <- function(input, output) {
  PCM_example <- reactive({PCM %>%
      filter(Wafer %in% WaferList)
  })

  output$plot <- renderPlot({ggplot(PCM_example(),
                                    aes(x=X_COORD,
                                        y=Y_COORD,
                                        group=Wafer,
                                        fill= value)) +
      geom_tile() +
      scale_fill_gradient(low="red",high="green") +
      facet_grid(.~Wafer)
  })
}
```

Let’s continue working on the ui side

Let’s add the ON/OFF button for Log10

```
ui <- fluidPage(
  selectInput("Wafer",
              label = h3("Select Wafer"),
              sort(WaferList),
              multiple = TRUE,
              selected = WaferList[1]),
  checkboxInput("checkbox", label = "log10", value = FALSE),
  plotOutput('plot')
)
```

Let's continue working on the Server side

```
if (input$checkbox){
  ggplot(PCM_example(), aes(x=X_COORD,
                            y=Y_COORD,
                            group=Wafer,
                            fill= value)) +
    geom_tile() +
    scale_fill_gradient(low="red",
                        high="green",
                        trans = "log10") +
    facet_grid(.~Wafer)
}
```

Let's continue working on the ui side

```r
ui <- fluidPage(
  selectInput("Wafer", label = h3("Select Wafer"),
              sort(unique(PCM$Wafer)),
              multiple = TRUE,
              selected = WaferList[1]),
  checkboxInput("checkbox", label = "log10", value = FALSE),
  plotOutput('plot')
)
```