Tab 1

1. Load library dplyr!
```
library(dplyr)
```

2. Tampilin 10 game dgn penjualan terbanyak di US utk game yg rilis pd th 2000 s/d 2012. Tampilin hanya kolom nama & total penjualan di US.
```
video_game_sales %>%
 filter(year >= 2000 & year <= 2012) %>%
 arrange(desc(us_sales)) %>%
 select(name, us_sales) %>%
 head(10)
```

3. Tampilin semua nama kolom pd dataframe tersebut!
```
names(video_game_sales)
```
/pakai fungsi colnames() atau str()

4. Tampilin tipe data yg dimiliki oleh kolom "platform". Pd bag bawah kode, jelasin mengapa kolom "platform" pakai tipe data tsb.
```
- class(video_game_sales$platform)
```
- Kolom "platform" punya tipe data factor karena kolom tsb menyimpan data kategorikal, di mana pd data tsb terdapat 13 macam platform.

5. Tampilin 5 data teratas utk game pd platform "Wii" diurutkan berdasarkan th rilis terbaru. Tampilin semua kolom kec kolom "us_sales"
```
video_game_sales %>%
 filter(platform == "Wii") %>%
 arrange(desc(year)) %>%
 select(-us_sales) %>%
 head(5)
```

6. Buat sebuah klasifikasi pd dtframe tsb berdasarkan penjualan "global_sales" dgn kondisi:
1) Penjualan >= 20 dikategorikan sbg "Untung", 2) Penjualan di antara 10-20 dikategorikan sbg "Biasa aja"
3) Penjualan <= 10 dikategorikan sbg "Rugi". Masukin ke suatu variabel baru -> 'tingkat_penjualan'.
```
- tingkat_penjualan = ifelse(video_game_sales$global_sales >= 20, "Untung", ifelse(video_game_sales$global_sales <= 10, "Rugi", "Biasa aja"))
tingkat_penjualan
- tingkat_penjualan2 = case_when(
 video_game_sales$global_sales >= 20 ~ "untung",
 video_game_sales$global_sales > 10 ~ "biasa aja",
 video_game_sales$global_sales <= 10 ~ "rugi",)
tingkat_penjualan2
```

7. Tambahin kolom "tingkat_penjualan" td sbg kolom baru pd dataframe. Tampilin 8 data teratas berdasarkan global_sales.
```
video_game_sales =
video_game_sales %>%
mutate(tingkat_penjualan)

video_game_sales %>%
 arrange(desc(global_sales)) %>%
 head(8)
```

8. Ambil masing² 5 data pd tiap kategori pada tingkat_penjualan. Gabung data² tsb ke dlm suatu dataframe baru bernama "new_video_game_sales". Setelah itu, Tampilkan isinya!
```
new_video_game_sales =
video_game_sales %>%
 group_by(tingkat_penjualan) %>%
 sample_n(5)
new_video_game_sales
```

9. Tampilin rata² & jumlah dari global_sales dikelompokkan berdasarkan platform!
```
hasil = video_game_sales %>%
 group_by(platform) %>%
 summarise(rerata = mean(global_sales),
   total = sum(global_sales) )
hasil
```

10. Lakukan visualisasi dgn barplot dgn sumbu x yaitu nama platformnya & sumbu y yaitu rata² penjualan dari hasil no 9!
```
barplot(
 hasil$rerata ~ hasil$platform,
 las = 2, # <- buat nge-rotate x label biar semua labelnya keliatan
 xlab = "",
 ylab = "Rata-rata Penjualan",
 main = "Tingkat Penjualan Global berdasarkan Platform")
```

11. Buat histogram dgn sumbu x yaitu th pembuatan game & sumbu y banyaknya game di tahun tsb. Di bawah chunk tulis kesimpulan yg dpt diperoleh dari histogramnya!
```
hist(video_game_sales$year,
 main = "Jumlah Video Game yang Rilis tiap Tahun",xlab = "Tahun Rilis",
 ylab = "Jumlah Video Game")
```
Kesimpulan: Berdasarkan histogram di atas, tahun 2005 s/d 2010 merupakan tahun video game paling banyak rilis.

1) Import dataset airquality1.csv & airquality2.csv menggunakan library here, lalu tampilkan 10 data awal.
```
- data_airquality1 = read.csv(here("airquality1.csv"))
head(data_airquality1, 10)
- data_airquality2 = read.csv(here("airquality2.csv"))
head(data_airquality2, 10)
```
Alter:
```
path1 <- here("data-raw", "airquality1.csv")
airquality1 <- read.csv(path1)
head(airquality1, 10)
path2 <- here("data-raw", "airquality2.csv")
airquality2 <- read.csv(path2)
head(airquality2, 10)
```

2) Dari soal sebelumnya, dpt dilihat bahwa dataset airquality1 memiliki nilai N/A pd beberapa kolom. Hapus nilai N/A atau lakukan imputasi data sederhana utk mengisi nilai N/A, lalu tampilkan 10 data pertama.
```
data_airquality1$Ozone[is.na(data_airquality1$Ozone)] = mean (data_airquality1$Ozone, na.rm=TRUE)
```

data_airquality1$Solar.R[is.na(data_airquality1$Solar.R)] = mean(data_airquality1$Solar.R, na.rm=TRUE)
head(data_airquality1, 10)
Alter:
#airquality1$Ozone[is.na(airquality1$Ozone)] = median(#airquality1$Ozone[!is.na(airquality1$Ozone)])
#airquality1$Solar.R[is.na(airquality1$Solar.R)] = median(#airquality1$Solar.R[!is.na(airquality1$Solar.R)])
airquality1 = na.omit(airquality1)
head(airquality1,10)

3) Dataset airquality1 & airquality2, memiliki 1 kolom yg sama. Gunakan kolom itu utk menyatukan dataset ke variabel baru bernama airquality. Tampilkan 6 data terakhirnya.
airquality <- data_airquality1 %>% inner_join(data_airquality2, by = "X")
tail(airquality, 6)
Alter :airquality = inner_join(airquality1, airquality2, by = 'X')
tail(airquality)

4) Buat kolom baru bernama Date yg menyimpan kombinasi tgl dari kolom Month dan Day dgn format yyyy-MM-dd (tahun = 1973). Gunakan fungsi paste utk gabungin string & fungsi as.POSIXct untuk mengubah string menjadi tanggal.
airquality$Date <- as.POSIXct(paste("1973", airquality$Month, airquality$Day, sep = "-"), format = "%Y-%m-%d") head(airquality)
Alter: airquality = airquality %>% mutate(
Date = as.POSIXct(paste('1973',Month,Day,sep='-'), format = "%Y-%m-%d"))
head(airquality)

5) Setelah itu, buang kolom X, Day, & Month yg tidak akan digunakan utk buat model. Kemudian, ubah nama kolom Solar.R menjadi Solar_Radiation. Gunakan operator pipeline.
airquality %>% select(-X, -Day, -Month) %>% rename(Solar_Radiation = Solar.R)

6) Gambarin perubahan kualitas udara (Ozone) setiap harinya dgn ggplot2. Kombinasiin 2 jenis geom yg sesuai dengan data yang ada.
library(ggplot2)
ggplot(airquality, aes(x = Date, y = Ozone)) + geom_line(color = "navy") + geom_point(color = "tan") + labs(title = "Perubahan Kualitas Udara Harian (Ozone)", x = "Tanggal", y = "Kadar Ozone")

7) Variabel pada dataset ini memiliki range yg berbeda. Lakukan scaling agar berada di range yang sama.
airquality <- airquality %>% mutate(across(where(is.numeric), scale)) head(airquality)
Alter: airquality = airquality %>% mutate(across(c(Solar_Radiation, Wind, Temp),scale)) head(airquality)

8) Bagi dataset untuk training & testing dgn proporsi training 80%. Pastikan dataset diacak sebelum dibagi & pastiin hasil acak akan konsisten walaupun dijalankan berulang dari perangkat berbeda.
set.seed(42)
split <- initial_split(airquality, prop = 0.8, strata = Ozone)
airqual_train <- training(split)
airqual_test <- testing(split)
dim(airqual_train)
dim(airqual_test)

9) Buat resep untuk training data. Tentuin 3 variabel yang jd prediktor & 1 variabel yang jd outcome. Biarin variabel Date sebagai ID.
recipe_airquality <- recipe(Ozone ~ Solar.R + Wind + Temp, data = airqual_train) %>% step_normalize(all_numeric())
recipe_airquality
train_prep <- prep(recipe_airquality, training = airqual_train) %>% juice()
test_prep <- prep(recipe_airquality, training = airqual_train) %>% bake(new_data = airqual_test)
Alter: airquality_recipe = training(airquality_split) %>% recipe() %>% update_role(Solar_Radiation, Wind, Temp, new_role = 'predictor') %>% update_role(Ozone, new_role = 'outcome') %>% update_role(Date, new_role = 'ID') %>% step_corr(all_predictors())
summary(airquality_recipe)

10) Training model berdasarkan data yang sudah diolah.
model_airquality <- lm(Ozone ~ Solar.R + Wind + Temp, data = airqual_train)
summary(model_airquality)
Alter:#penerapan resep
airquality_training = airquality_recipe %>% prep() %>% bake(training(airquality_split))
airquality_testing = airquality_recipe %>% prep() %>% bake(testing(airquality_split))
#training airquality_model = linear_reg(mode = "regression") %>% set_engine("lm") %>% fit(Ozone ~ Solar_Radiation + Wind + Temp, data = airquality_training)
airquality_model

11) Evaluasi performa model menggunakan data testing
predictions <- predict(model_airquality, newdata = airqual_test) library(Metrics)
rmse_value <- rmse(airqual_test$Ozone, predictions) print(paste("RMSE: ", rmse_value))
Alter: airquality_model %>% predict(airquality_testing) %>% bind_cols(airquality_testing) %>% metrics(truth = Ozone, estimate = .pred)