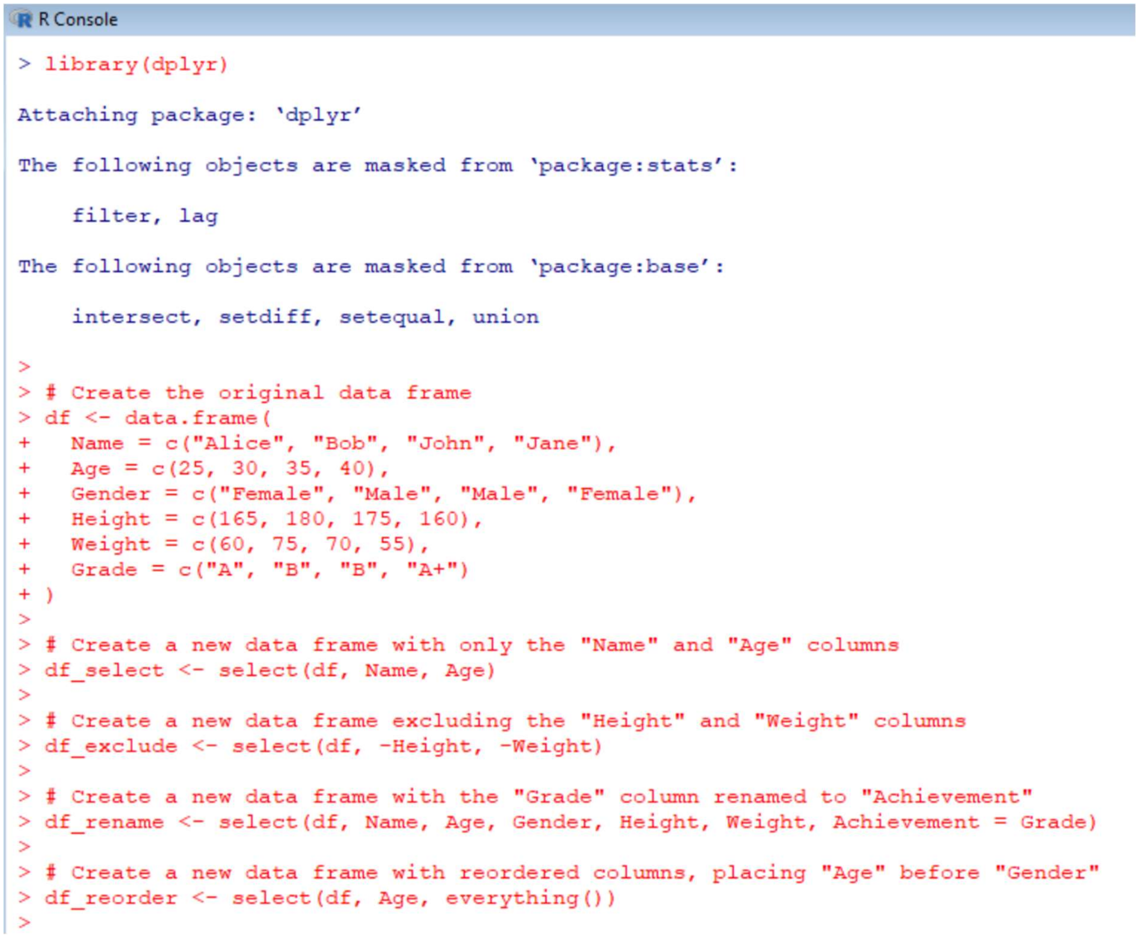
FDA Lab-9

KHAN MOHD OWAIS RAZA
20BCD7138

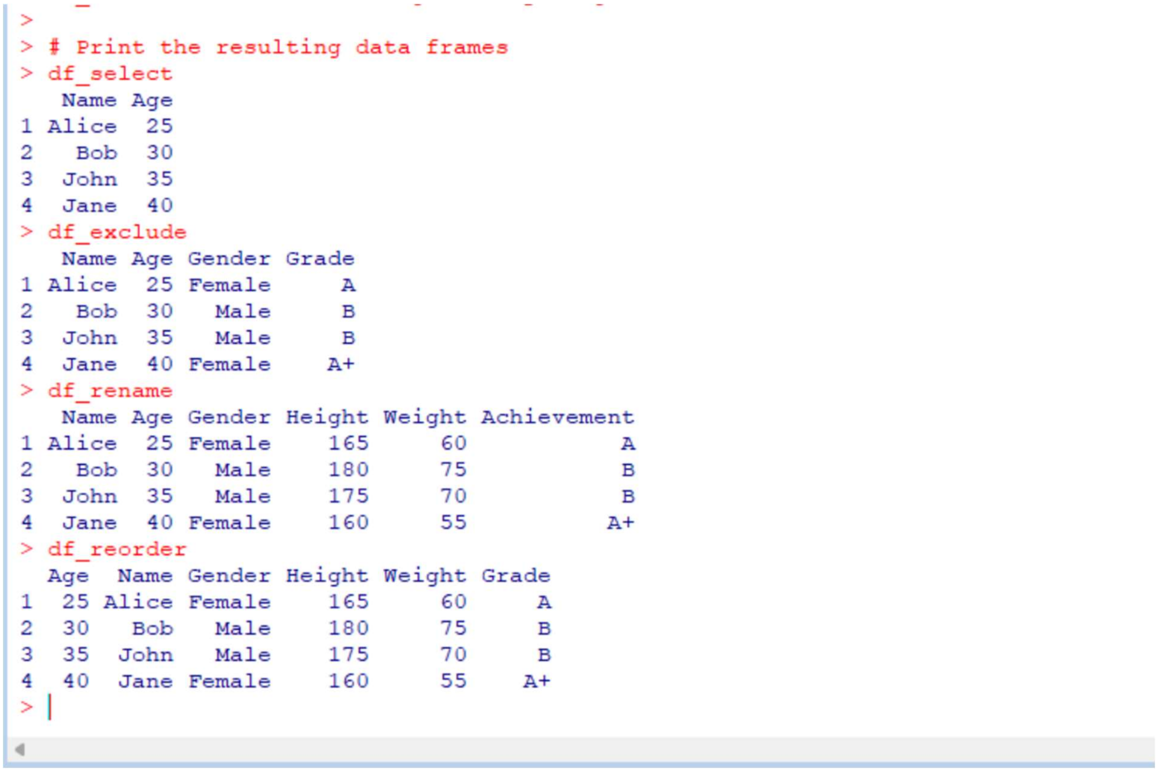
Q1] Consider a data frame "df" with the following columns: "Name", "Age", "Gender", "Height", "Weight", "Grade".

```
df <- data.frame(
  Name = c("Alice", "Bob", "John", "Jane"),
  Age = c(25, 30, 35, 40),
  Gender = c("Female", "Male", "Male", "Female"),
  Height = c(165, 180, 175, 160),
  Weight = c(60, 75, 70, 55),
  Grade = c("A", "B", "B", "A+")
)
```

- Use the **select()** function in R to create a new data frame called "df_select" that includes only the "Name" and "Age" columns from the original data frame.
- Use the **select()** function to create a new data frame called "df_exclude" that excludes the "Height" and "Weight" columns from the original data frame.
- Use the **select()** function to create a new data frame called "df_rename" that renames the "Grade" column to "Achievement" in the original data frame.
- Use the **select()** function to create a new data frame called "df_reorder" that reorders the columns of the original data frame, with "Age" appearing before "Gender".



```
> library(dplyr)

Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union

>
> # Create the original data frame
> df <- data.frame(
+   Name = c("Alice", "Bob", "John", "Jane"),
+   Age = c(25, 30, 35, 40),
+   Gender = c("Female", "Male", "Male", "Female"),
+   Height = c(165, 180, 175, 160),
+   Weight = c(60, 75, 70, 55),
+   Grade = c("A", "B", "B", "A+")
+ )
>
> # Create a new data frame with only the "Name" and "Age" columns
> df_select <- select(df, Name, Age)
>
> # Create a new data frame excluding the "Height" and "Weight" columns
> df_exclude <- select(df, -Height, -Weight)
>
> # Create a new data frame with the "Grade" column renamed to "Achievement"
> df_rename <- select(df, Name, Age, Gender, Height, Weight, Achievement = Grade)
>
> # Create a new data frame with reordered columns, placing "Age" before "Gender"
> df_reorder <- select(df, Age, everything())
>
```



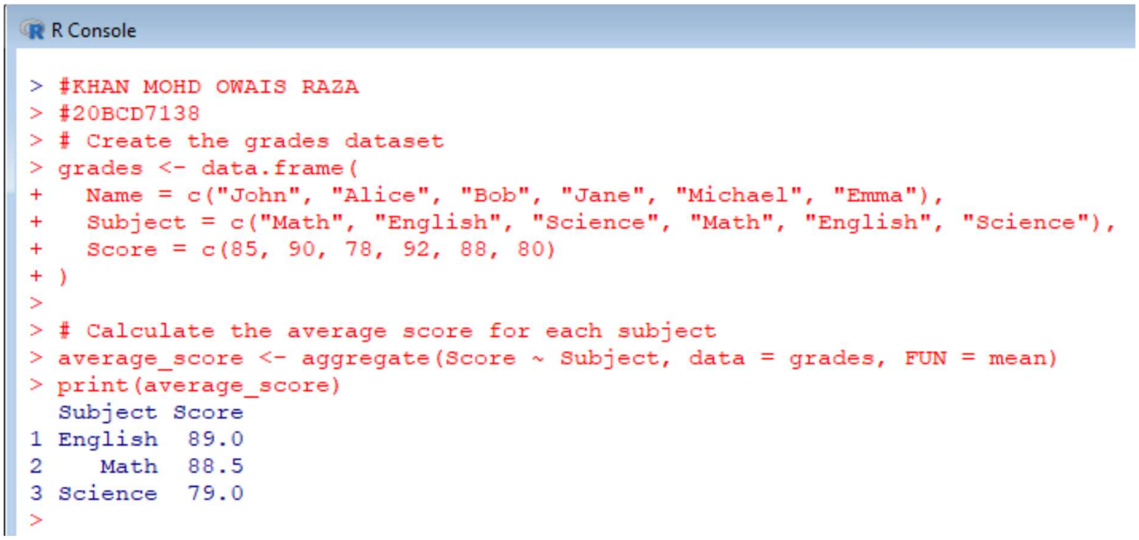
```
>
> # Print the resulting data frames
> df_select
   Name Age
1 Alice  25
2   Bob  30
3  John  35
4  Jane  40
> df_exclude
   Name Age Gender Grade
1 Alice  25 Female     A
2   Bob  30   Male     B
3  John  35   Male     B
4  Jane  40 Female    A+
> df_rename
   Name Age Gender Height Weight Achievement
1 Alice  25 Female    165     60           A
2   Bob  30   Male    180     75           B
3  John  35   Male    175     70           B
4  Jane  40 Female    160     55          A+
> df_reorder
  Age  Name Gender Height Weight Grade
1  25 Alice Female    165     60     A
2  30   Bob   Male    180     75     B
3  35  John   Male    175     70     B
4  40  Jane Female    160     55    A+
> |
```

Q2] Consider a dataset "grades" with the following columns: "Name", "Subject", and "Score".

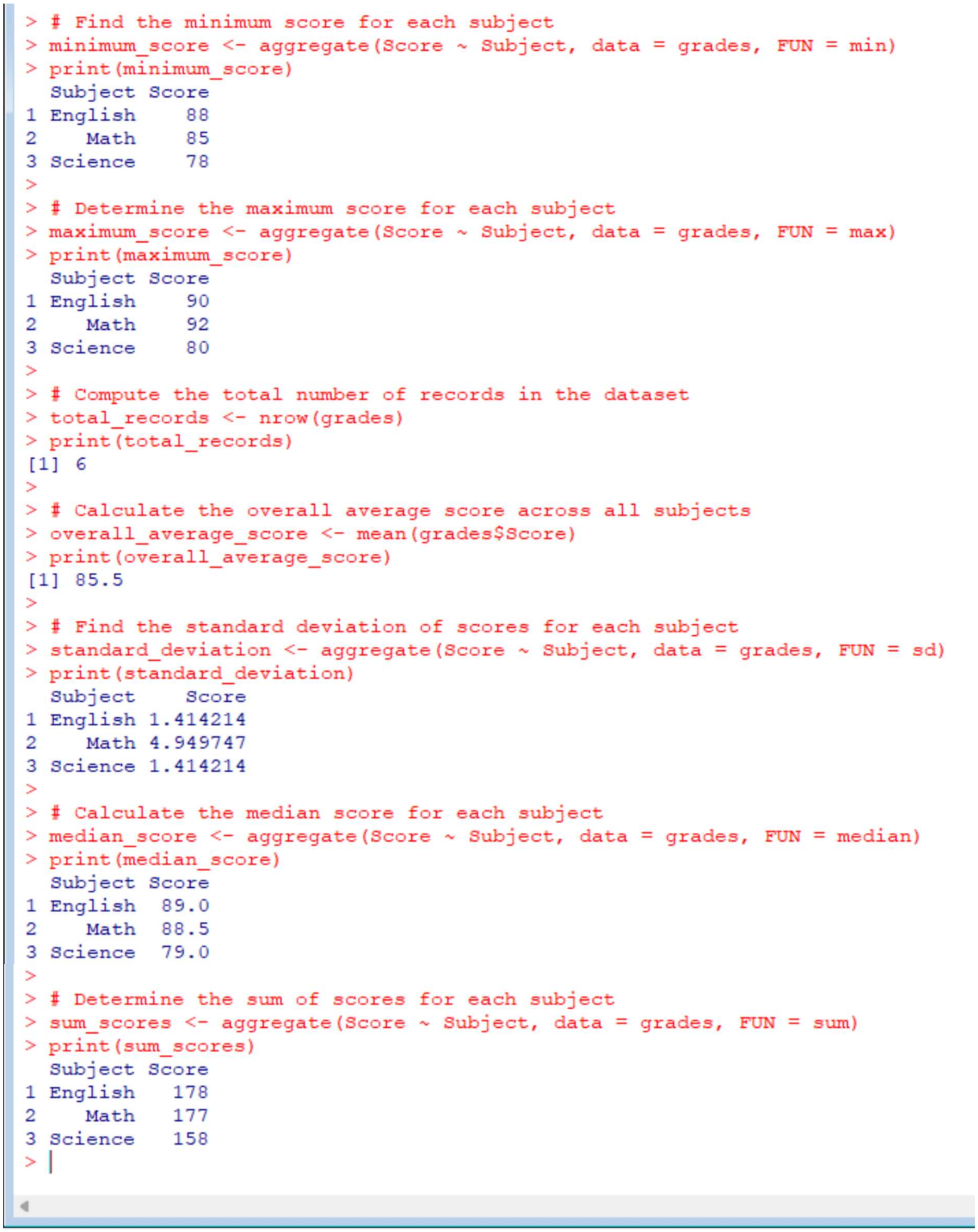
```
grades <- data.frame(
  Name = c("John", "Alice", "Bob", "Jane", "Michael", "Emma"),
  Subject = c("Math", "English", "Science", "Math", "English", "Science"),
  Score = c(85, 90, 78, 92, 88, 80)
)
```

Write R code to perform the following summarization tasks:

- Calculate the average score for each subject.
- Find the minimum score for each subject.
- Determine the maximum score for each subject.
- Compute the total number of records in the dataset.
- Calculate the overall average score across all subjects.
- Find the standard deviation of scores for each subject.
- Calculate the median score for each subject.
- Determine the sum of scores for each subject.



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Create the grades dataset
> grades <- data.frame(
+   Name = c("John", "Alice", "Bob", "Jane", "Michael", "Emma"),
+   Subject = c("Math", "English", "Science", "Math", "English", "Science"),
+   Score = c(85, 90, 78, 92, 88, 80)
+ )
>
> # Calculate the average score for each subject
> average_score <- aggregate(Score ~ Subject, data = grades, FUN = mean)
> print(average_score)
  Subject Score
1 English  89.0
2    Math  88.5
3 Science  79.0
>
```



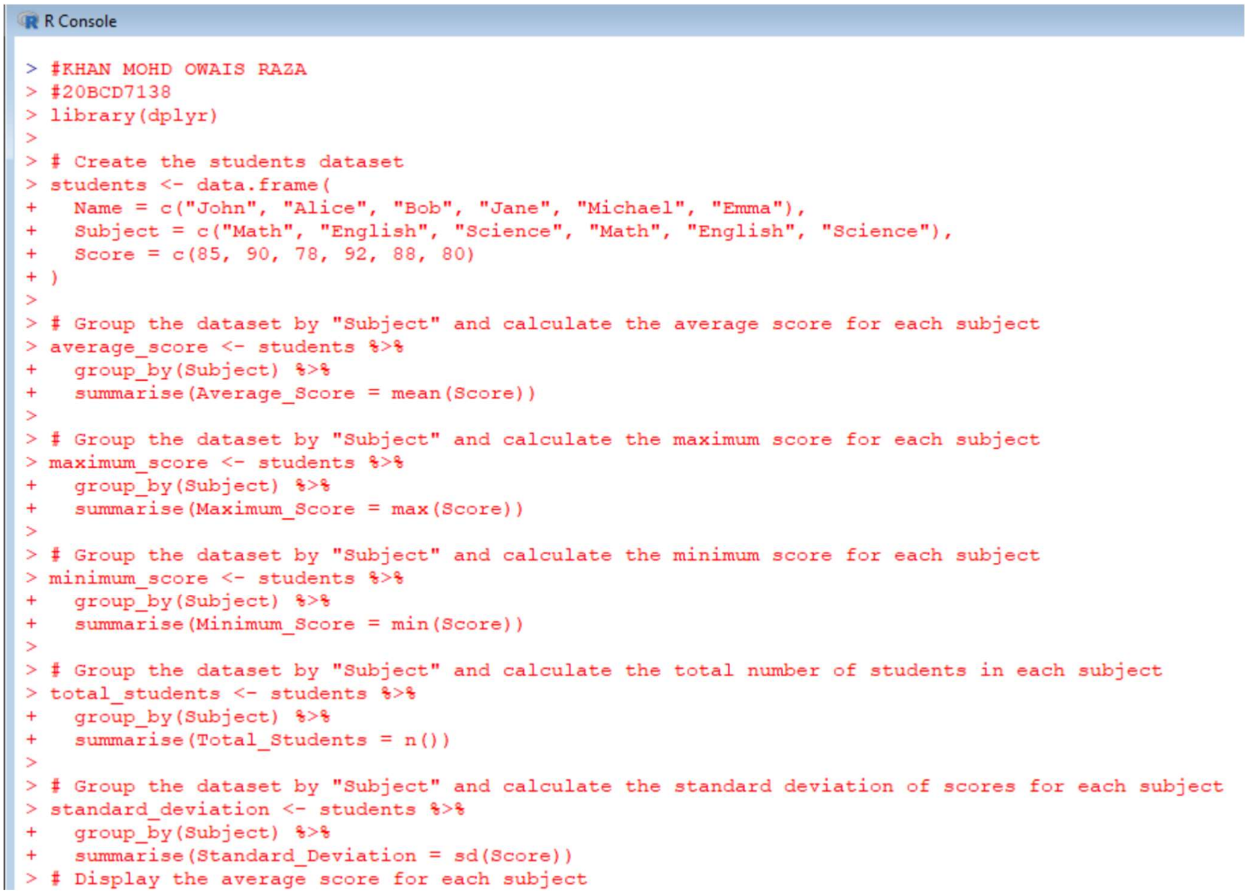
```
> # Find the minimum score for each subject
> minimum_score <- aggregate(Score ~ Subject, data = grades, FUN = min)
> print(minimum_score)
  Subject Score
1 English    88
2    Math    85
3 Science    78
>
> # Determine the maximum score for each subject
> maximum_score <- aggregate(Score ~ Subject, data = grades, FUN = max)
> print(maximum_score)
  Subject Score
1 English    90
2    Math    92
3 Science    80
>
> # Compute the total number of records in the dataset
> total_records <- nrow(grades)
> print(total_records)
[1] 6
>
> # Calculate the overall average score across all subjects
> overall_average_score <- mean(grades$Score)
> print(overall_average_score)
[1] 85.5
>
> # Find the standard deviation of scores for each subject
> standard_deviation <- aggregate(Score ~ Subject, data = grades, FUN = sd)
> print(standard_deviation)
  Subject    Score
1 English 1.414214
2    Math 4.949747
3 Science 1.414214
>
> # Calculate the median score for each subject
> median_score <- aggregate(Score ~ Subject, data = grades, FUN = median)
> print(median_score)
  Subject Score
1 English  89.0
2    Math  88.5
3 Science  79.0
>
> # Determine the sum of scores for each subject
> sum_scores <- aggregate(Score ~ Subject, data = grades, FUN = sum)
> print(sum_scores)
  Subject Score
1 English   178
2    Math   177
3 Science   158
> |
```

Q3] Consider the following dataset "students" representing students' scores in different subjects:

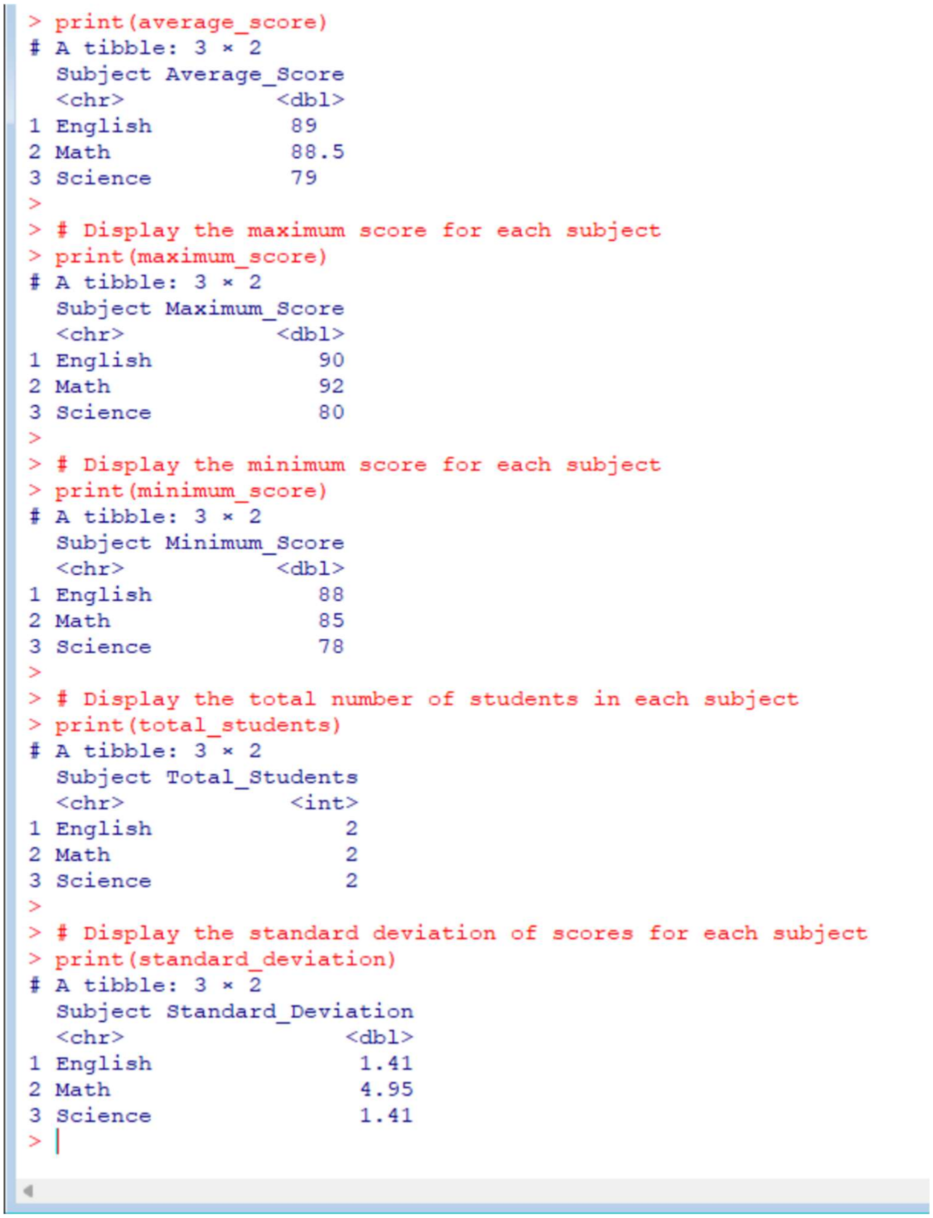
students <- data.frame(
  Name = c("John", "Alice", "Bob", "Jane", "Michael", "Emma"),
  Subject = c("Math", "English", "Science", "Math", "English", "Science"),
  Score = c(85, 90, 78, 92, 88, 80)
)

1. Group the dataset by "Subject" and calculate the average score for each subject.
2. Group the dataset by "Subject" and calculate the maximum score for each subject.
3. Group the dataset by "Subject" and calculate the minimum score for each subject.
4. Group the dataset by "Subject" and calculate the total number of students in each subject.
5. Group the dataset by "Subject" and calculate the standard deviation of scores for each subject.

For each question, write the necessary R code using the **group_by()** function and appropriate summarization functions such as **mean()**, **max()**, **min()**, **n()**, and **sd()**.



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> library(dplyr)
>
> # Create the students dataset
> students <- data.frame(
+   Name = c("John", "Alice", "Bob", "Jane", "Michael", "Emma"),
+   Subject = c("Math", "English", "Science", "Math", "English", "Science"),
+   Score = c(85, 90, 78, 92, 88, 80)
+ )
>
> # Group the dataset by "Subject" and calculate the average score for each subject
> average_score <- students %>%
+   group_by(Subject) %>%
+   summarise(Average_Score = mean(Score))
>
> # Group the dataset by "Subject" and calculate the maximum score for each subject
> maximum_score <- students %>%
+   group_by(Subject) %>%
+   summarise(Maximum_Score = max(Score))
>
> # Group the dataset by "Subject" and calculate the minimum score for each subject
> minimum_score <- students %>%
+   group_by(Subject) %>%
+   summarise(Minimum_Score = min(Score))
>
> # Group the dataset by "Subject" and calculate the total number of students in each subject
> total_students <- students %>%
+   group_by(Subject) %>%
+   summarise(Total_Students = n())
>
> # Group the dataset by "Subject" and calculate the standard deviation of scores for each subject
> standard_deviation <- students %>%
+   group_by(Subject) %>%
+   summarise(Standard_Deviation = sd(Score))
> # Display the average score for each subject
```



```
> print(average_score)
# A tibble: 3 × 2
  Subject Average_Score
  <chr>           <dbl>
1 English          89  
2 Math             88.5
3 Science          79  
> 
> # Display the maximum score for each subject
> print(maximum_score)
# A tibble: 3 × 2
  Subject Maximum_Score
  <chr>           <dbl>
1 English            90
2 Math               92
3 Science            80
> 
> # Display the minimum score for each subject
> print(minimum_score)
# A tibble: 3 × 2
  Subject Minimum_Score
  <chr>           <dbl>
1 English            88
2 Math               85
3 Science            78
> 
> # Display the total number of students in each subject
> print(total_students)
# A tibble: 3 × 2
  Subject Total_Students
  <chr>            <int>
1 English              2
2 Math                 2
3 Science              2
> 
> # Display the standard deviation of scores for each subject
> print(standard_deviation)
# A tibble: 3 × 2
  Subject Standard_Deviation
  <chr>                <dbl>
1 English               1.41
2 Math                  4.95
3 Science               1.41
> 
```

Q4] Consider the following dataset "students" representing students' information

- Filter the dataset to select students who are older than 20 years.
- Filter the dataset to select students who have a grade of "A".
- Filter the dataset to select students who have a GPA higher than 3.5.
- Filter the dataset to select students who are older than 20 years and have a grade of "A".
- Filter the dataset to select students who have a GPA higher than 3.5 or are younger than 19 years.

For each question, write the necessary R code using the **filter()** function and use appropriate conditions using comparison operators such as <, >, ==, and logical operators such as **&** (AND) and | (OR).

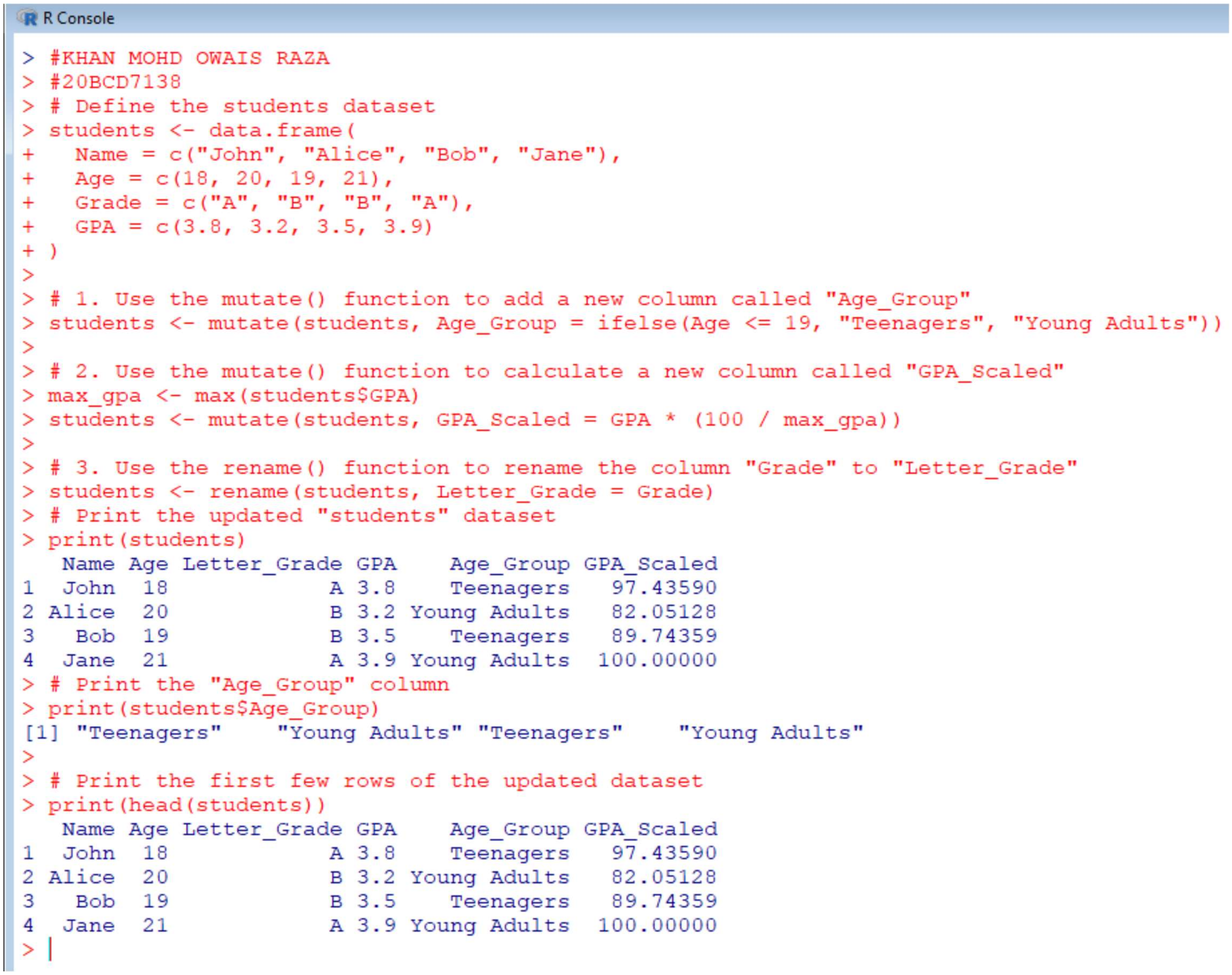
```
# Load the "students" dataset
students <- data.frame(
  Name = c("John", "Alice", "Bob", "Jane", "Michael",
"Emma"),
  Subject = c("Math", "English", "Science", "Math",
"English", "Science"),
  Score = c(85, 90, 78, 92, 88, 80)
)

# Filter the dataset to select students who are older
than 20 years
filtered_students_1 <- students %>% filter(Age > 20)
print("Students older than 20 years:")
print(filtered_students_1)

# Filter the dataset to select students who have a grade
of "A"
filtered_students_2 <- students %>% filter(Score == "A")
print("Students with a score of 'A':")
print(filtered_students_2)

# Filter the dataset to select students who have a GPA
higher than 3.5
filtered_students_3 <- students %>% filter(Score > 3.5)
print("Students with a score higher than 3.5:")
print(filtered_students_3)

# Filter the dataset to select students who are older
than 20 years and have a grade of "A"
filtered_students_4 <- students %>% filter(Age > 20,
Score == "A")
print("Students older than 20 years with a score of
'A':")
print(filtered_students_4)

# Filter the dataset to select students who have a GPA
higher than 3.5 or are younger than 19 years
filtered_students_5 <- students %>% filter(Score > 3.5 |
Age < 19)
print("Students with a score higher than 3.5 or younger
than 19 years:")
print(filtered_students_5)
```

Q5] Consider the following dataset "students" representing students' information:

```
            students <- data.frame(
      Name = c("John", "Alice", "Bob", "Jane"),
      Age = c(18, 20, 19, 21),
      Grade = c("A", "B", "B", "A"),
      GPA = c(3.8, 3.2, 3.5, 3.9)
    )
```

1. Use the **mutate()** function to add a new column called "Age_Group" to the dataset, which categorizes the students into different age groups: "Teenagers" (age <= 19) and "Young Adults" (age > 19).
2. Use the **mutate()** function to calculate a new column called "GPA_Scaled" that scales the GPA values to a 100-point scale, where the maximum GPA in the dataset corresponds to 100.
3. Use the **rename()** function to rename the column "Grade" to "Letter_Grade" in the dataset.

For each question, write the necessary R code using the **mutate()** and **rename()** functions to perform the desired operations on the "students" dataset.



```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Define the students dataset
> students <- data.frame(
+   Name = c("John", "Alice", "Bob", "Jane"),
+   Age = c(18, 20, 19, 21),
+   Grade = c("A", "B", "B", "A"),
+   GPA = c(3.8, 3.2, 3.5, 3.9)
+ )
>
> # 1. Use the mutate() function to add a new column called "Age_Group"
> students <- mutate(students, Age_Group = ifelse(Age <= 19, "Teenagers", "Young Adults"))
>
> # 2. Use the mutate() function to calculate a new column called "GPA_Scaled"
> max_gpa <- max(students$GPA)
> students <- mutate(students, GPA_Scaled = GPA * (100 / max_gpa))
>
> # 3. Use the rename() function to rename the column "Grade" to "Letter_Grade"
> students <- rename(students, Letter_Grade = Grade)
> # Print the updated "students" dataset
> print(students)
   Name Age Letter_Grade GPA    Age_Group GPA_Scaled
1  John  18            A 3.8    Teenagers   97.43590
2 Alice  20            B 3.2 Young Adults   82.05128
3   Bob  19            B 3.5    Teenagers   89.74359
4  Jane  21            A 3.9 Young Adults  100.00000
> # Print the "Age_Group" column
> print(students$Age_Group)
[1] "Teenagers"    "Young Adults" "Teenagers"    "Young Adults"
>
> # Print the first few rows of the updated dataset
> print(head(students))
   Name Age Letter_Grade GPA    Age_Group GPA_Scaled
1  John  18            A 3.8    Teenagers   97.43590
2 Alice  20            B 3.2 Young Adults   82.05128
3   Bob  19            B 3.5    Teenagers   89.74359
4  Jane  21            A 3.9 Young Adults  100.00000
> |
```