title: “Day 2 – 01 Explore Data (Exercises)” author: “Seminar practice worksheet” output: html_document: toc: true toc_depth: 2 —

This self-paced worksheet mirrors the guided exploration but uses `second_day_part2/data/dataset2_subset_long.csv`, which contains three genomes (Akkermansia, Bacteroides, Turicimonas). Students should type their own code into the chunks below. Hints are provided in the text; the code blocks are left with `# TODO` comments.

> **Tip:** Run `second_day_part2/00_prepare.Rmd` once to install packages, and `second_day_part2/data/00_prepare_dat` only if you need to regenerate the three-genome dataset.

1. Load and preview the dataset

Goal: use `read.csv()` to load the file, then report the number of rows/columns and display the first few entries.

```
# TODO: load dataset2_subset_long.csv into an object named long_df
# TODO: print nrow(long_df) and ncol(long_df)
# TODO: call head(long_df)
```

Hint: remember to set `stringsAsFactors = FALSE` and `check.names = FALSE` so the column headers stay untouched.

2. Enumerate genomes and SNPs

Goal: list all genomes present, count how many rows each contributes, and count unique SNP identifiers per genome.

```
# TODO: unique() to list genomes
# TODO: table() to count rows per genome
# TODO: tapply() + length(unique()) to count SNPs per genome
```

Hint: model your solution after the formula `tapply(long_df$snp_id, long_df$Genome, ...)`.

3. Summaries by genome

Goal: compute summary statistics of `value` for each genome (min/median/mean, etc.) using `aggregate(value ~ Genome, ...)`. Also, show a quick `summary()` of the first few values to illustrate what the anonymous function does.

```
# TODO: sample_values <- long_df$value[1:10]; summary(sample_values)
# TODO: aggregate(value ~ Genome, data = long_df, function(x) summary(x))
```

Hint: read `value ~ Genome` as “value grouped by Genome”. The column to the left of `~` must be numeric.

4. Focus on Turicimonas

Goal: subset the data frame to `Turicimonas_muris_YL45` only, show `head()` of the subset, and build a `table(mouse_id, day)` for that genome.

```
# TODO: turicimonas_df <- long_df[long_df$Genome == 'Turicimonas_muris_YL45', ]
# TODO: head(turicimonas_df)
# TODO: table(turicimonas_df$mouse_id, turicimonas_df$day)
```

Hint: inspect `head(long_df$Genome == 'Turicimonas_muris_YL45')` to see the TRUE/FALSE mask created by the comparison.

5. Allele-frequency focus (Turicimonas vs all genomes)

Goal: plot the allele-frequency (`value`) histogram for `Turicimonas_muris_YL45`, then for the entire dataset. Report `summary()` for both vectors and note whether Turicimonas shows more high-frequency SNPs.

```
# TODO: turicimonas_values <- long_df$value[long_df$Genome == 'Turicimonas_muris_YL45']
# TODO: hist(turicimonas_values, breaks = 20, main = 'Turicimonas AF', xlab = 'value')
# TODO: summary(turicimonas_values)
# TODO: hist(long_df$value, breaks = 30, main = 'All genomes AF', xlab = 'value')
# TODO: summary(long_df$value)
# Question: Does Turicimonas concentrate at higher or lower allele frequencies compared to the mix?
```

Hint: reuse the histogram idea from the guided notebook and jot down your interpretation.

6. Missing values

Goal: check whether the `value` column contains any `NA`s overall and per mouse/day combination.

```
# TODO: na_total <- sum(is.na(long_df$value))
# TODO: na_by_mouse_day <- with(long_df, tapply(value, list(mouse_id, day), function(x) sum(is.na(x))))
# TODO: print both objects and interpret (all zeros means no missing data).
```

Question: If you do detect NA values, which genome(s) or treatment groups do they belong to? State explicitly whether Turicimonas introduces any missing allele frequencies.

7. Treatment groups

Goal: use the `treatment_group` column to understand how samples are distributed across treatments.

```
# TODO: list unique(long_df$treatment_group)
# TODO: table(long_df$treatment_group)
# TODO: table(long_df$mouse_id, long_df$treatment_group)
# TODO: subset long_df for a specific day (e.g., day 30) and inspect treatment groups
```

Hint: use `table()` for quick counts and `unique()` to see combinations of mouse/day/group.

8. Stretch idea (optional)

Can you reuse these skills to answer: "Which genome has the highest median value on day 30?" Outline your approach below before coding it up.

```
# Notes / pseudo-code:
# 1. Filter long_df to day == 30
# 2. Split by Genome and compute median(value)
# 3. Identify the maximum
```

These practice tasks mirror the guided walkthrough but force you to re-create the analysis yourself. Once satisfied, continue with `scripts/02_simple_heatmap.Rmd` to generate the heatmap.