# Small Errors, Big Consequences: Lessons from Data Mismanagement in Business

Karla Hernández - 400916125 - hernandez.karla@stud.hs-fresenius.de

**Scan me**

## Why Smalls Errors Matter

- Data drives decisions in production, hiring, marketing & forecasting
- Most business data failures are not dramatic bugs
- They come from tiny, unnoticed mistakes
- If the data is wrong, the decision is wrong
- RStudio does not warn when results are logically wrong

### What Counts as a "Small Error"?

| Error | What it Means |
|---|---|
| "Mexico" vs "México" | R treats them as two different categories |
| Number stored as text | Math doesn't work the way we expect |
| Missing values | Averages and totals become wrong |
| Duplicate rows | Customers or employees counted twice |
| Wrong join | Rows multiply or disappear silently |

### Tiny Errors, Big Consequences

| Area | Tiny Error | Big Result |
|---|---|---|
| Retail | Missing value in sales | Too little production that leads to lost on sales |
| HR | Employee duplicated during merge | Over-hired people that leads to unnecessary cost |
| Marketing | Duplicated customers | Overspent on ads |

### Bad Practice vs Good Practice

*Scenario:*

What is the average purchase amount of customers under 30, and how does it differ by location?

- Click to download dataset: CSV shopping_behavior dataset

### Bad Practice Example

```
# BAD import: let R guess names and types
data_bad <- read.csv("shopping_behavior.csv")

# BAD filter (forgetting to handle NA)
young_bad <- data_bad[data_bad$Age < 30, ]

# BAD join using merge() with the same location_info
location_info <- data.frame(
  Location = c("California", "New York", "Texas", "Florida"),
  Region   = c("West", "East", "South", "South"),
  stringsAsFactors = FALSE
```

```
)

merged_bad <- merge(young_bad, location_info)

# BAD summary: mean without na.rm and no cleaning of Location variants
result_bad <- aggregate(Purchase.Amount..USD. ~ Location, data = merged_bad, FUN = mean)

# Rename columns to match good version later
names(result_bad) <- c("Location_clean", "avg_spend_bad")
result_bad$avg_spend_bad <- round(result_bad$avg_spend_bad, 2)

result_bad
print(result_bad, n = Inf)
View(result_bad)
```

### Outcome

```
# BAD import: let R guess names and types
data_bad <- read.csv("shopping_behavior.csv")

# BAD filter (forgetting to handle NA, using base [] correctly this time but no cleaning)
young_bad <- data_bad[data_bad$Age < 30, ]

# BAD join using merge() with the same location_info
location_info <- data.frame(
  Location = c("California", "New York", "Texas", "Florida"),
  Region   = c("West", "East", "South", "South"),
  stringsAsFactors = FALSE
)

merged_bad <- merge(young_bad, location_info)

# BAD summary: mean without na.rm and no cleaning of Location variants
result_bad <- aggregate(Purchase.Amount..USD. ~ Location, data = merged_bad, FUN = mean)

# Rename columns to match good version later
names(result_bad) <- c("Location_clean", "avg_spend_bad")
result_bad$avg_spend_bad <- round(result_bad$avg_spend_bad, 2)

# Bad practice result
library(knitr)
library(kableExtra)
```

Table 3: Bad practice result

| Location_clean | avg_spend_bad |
|---|---|
| California | 66.30 |
| Florida | 53.33 |
| New York | 67.23 |
| Texas | 55.88 |

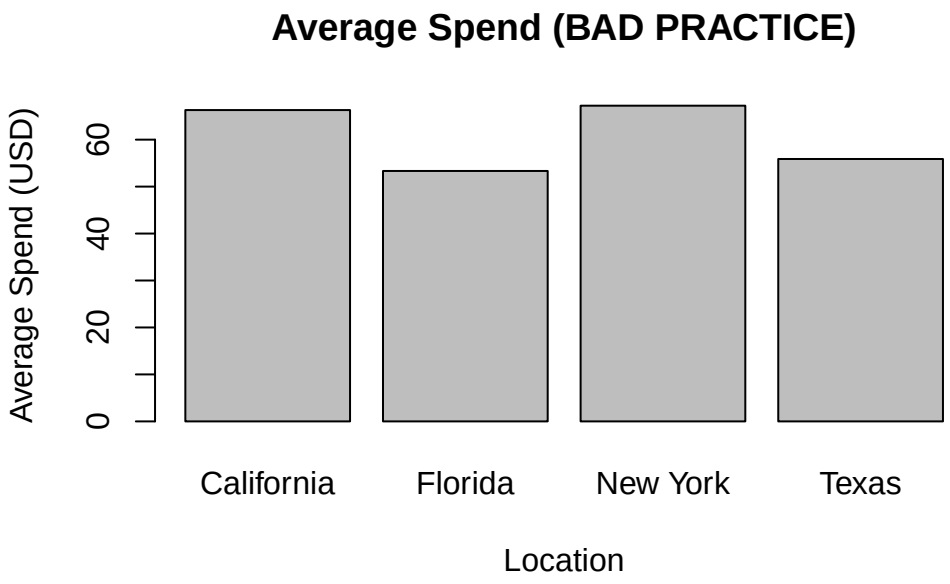
```
result_bad |>
  kbl(caption = "Bad practice result") |>
  scroll_box(height = "600px")
```

## Plot

```
# BAD plot result
barplot(
  result_bad$avg_spend_bad,
  names.arg = result_bad$Location,
  main = "Average Spend (BAD PRACTICE)",
  xlab = "Location",
  ylab = "Average Spend (USD)"
)
```

### Plot Outcome

```
# BAD plot result
barplot(
  result_bad$avg_spend_bad,
  names.arg = result_bad$Location,
  main = "Average Spend (BAD PRACTICE)",
  xlab = "Location",
  ylab = "Average Spend (USD)"
)
```




### Good Practice Example

```r
# Example of a good practice
# 1) Import packages
library(dplyr)
library(lubridate)
library(ggplot2)

# 2) Read data with good practices
data_good <- read.csv(
  "shopping_behavior.csv",
  stringsAsFactors = FALSE,
  check.names = FALSE   # keeps "Purchase Amount (USD)" as-is
)

# 3) Parse dates correctly
data_good <- data_good %>%
  mutate(
    Date_good = dmy(Date)
  )

# 4) Remove exact duplicate rows (we know we injected some)
data_nodup <- data_good %>%
  distinct(.keep_all = TRUE)
```

```r
# 5) Keep only customers under 30
young_good <- data_nodup %>%
  filter(Age < 30)

# 6) Location lookup table (regions are just for context)
location_info <- data.frame(
  Location = c("California", "New York", "Texas", "Florida"),
  Region   = c("West", "East", "South", "South"),
  stringsAsFactors = FALSE
)

# 7) Safe join: keep all young customers, add Region if Location matches
merged_good <- young_good %>%
  left_join(location_info, by = "Location")

# 8) Clean up Location variants into one standard name
final_clean <- merged_good %>%
  mutate(
    Location_clean = case_when(
      Location %in% c("California", "california", "Californi", "CA") ~ "California",
      Location %in% c("New York", "NewYork", "N. York")              ~ "New York",
      TRUE ~ Location
    )
  ) %>%
  group_by(Location_clean) %>%
  summarise(
    avg_spend_young = mean(`Purchase Amount (USD)`, na.rm = TRUE),
    n_customers     = n(),
    .groups = "drop"
  ) %>%
  arrange(Location_clean)

# 9) Round numbers to make the table nice for printing
final_clean$avg_spend_young <- round(final_clean$avg_spend_young, 2)

# 10) Print final result nicely
final_clean

print(final_clean, n = Inf)
View(final_clean)
```

### Outcome

```
# Example of a good practice
# 1) Import packages
library(dplyr)
```

```
Attaching package: 'dplyr'

The following object is masked from 'package:kableExtra':

    group_rows

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

    filter, lag

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

    intersect, setdiff, setequal, union
```

```
library(lubridate)
```

```
Attaching package: 'lubridate'

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

    date, intersect, setdiff, union
```

```
library(ggplot2)
```

```
Warning: package 'ggplot2' was built under R version 4.5.2
```

```
# 2) Read data with good practices
data_good <- read.csv(
  "shopping_behavior.csv",
  stringsAsFactors = FALSE,
  check.names = FALSE   # keeps "Purchase Amount (USD)" as-is
)
```

```r
# 3) Parse dates correctly
data_good <- data_good %>%
  mutate(
    Date_good = dmy(Date)
  )
```

```
Warning: There was 1 warning in `mutate()`.
i In argument: `Date_good = dmy(Date)`.
Caused by warning:
!  2 failed to parse.
```

```r
# 4) Remove exact duplicate rows (we know we injected some)
data_nodup <- data_good %>%
  distinct(.keep_all = TRUE)

# 5) Keep only customers under 30
young_good <- data_nodup %>%
  filter(Age < 30)

# 6) Location lookup table (regions are just for context)
location_info <- data.frame(
  Location = c("California", "New York", "Texas", "Florida"),
  Region   = c("West", "East", "South", "South"),
  stringsAsFactors = FALSE
)

# 7) Safe join: keep all young customers, add Region if Location matches
merged_good <- young_good %>%
  left_join(location_info, by = "Location")

# 8) Clean up Location variants into one standard name
final_clean <- merged_good %>%
  mutate(
    Location_clean = case_when(
      Location %in% c("California", "california", "Californi", "CA") ~ "California",
      Location %in% c("New York", "NewYork", "N. York")              ~ "New York",
      TRUE ~ Location
    )
  ) %>%
  group_by(Location_clean) %>%
  summarise(
    avg_spend_young = mean(`Purchase Amount (USD)`, na.rm = TRUE),
    n_customers     = n(),
    .groups = "drop"
```

```
  ) %>%
  arrange(Location_clean)

# 9) Round numbers to make the table nice for printing
final_clean$avg_spend_young <- round(final_clean$avg_spend_young, 2)

# 10) Print final result nicely
final_clean
```

```
# A tibble: 50 x 3
   Location_clean avg_spend_young n_customers
   <chr>                    <dbl>       <int>
 1 Alabama                   52.2          20
 2 Alaska                    66.9          18
 3 Arizona                   65.5          12
 4 Arkansas                  67.4          19
 5 California                66.1          25
 6 Colorado                  58.4          13
 7 Connecticut               54.7          15
 8 Delaware                  59.8          11
 9 Florida                   53.3          15
10 Georgia                   52.8          20
# i 40 more rows
```

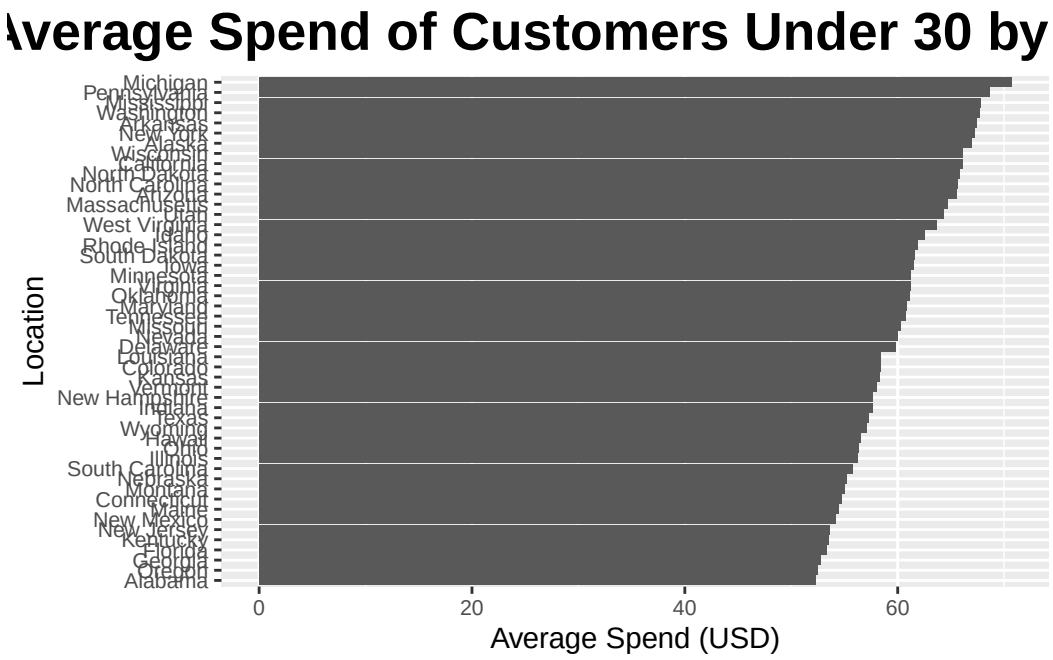
### Plot

```
# 11) Plot final results
plot_data <- final_clean %>%
  arrange(avg_spend_young) %>%
  mutate(Location_clean = factor(Location_clean, Location_clean))

ggplot(plot_data, aes(x = Location_clean, y = avg_spend_young)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  labs(
    title = "Average Spend of Customers Under 30 by Location",
    x = "Location",
    y = "Average Spend (USD)"
  ) +
  theme(
    plot.title = element_text(face = "bold", size = 20, hjust = 0.5),
    axis.text.y = element_text(size = 8),
```

```
    axis.text.x = element_text(size = 8)
  )
```

### Plot Outcome

```
# 11) Plot final results
plot_data <- final_clean %>%
  arrange(avg_spend_young) %>%
  mutate(Location_clean = factor(Location_clean, Location_clean))

ggplot(plot_data, aes(x = Location_clean, y = avg_spend_young)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  labs(
    title = "Average Spend of Customers Under 30 by Location",
    x = "Location",
    y = "Average Spend (USD)"
  ) +
  theme(
    plot.title = element_text(face = "bold", size = 20, hjust = 0.5),
    axis.text.y = element_text(size = 8),
    axis.text.x = element_text(size = 8)
  )
```



## Lessons from Data Mismanagement in Business

1. A *tiny mistake* in data can cause a **huge mistake** in a business decision.
2. RStudio won't warn you when the result is *wrong*, only when the **code** is.
3. Never *assume* the data is clean, always **check it**.
4. Good analysis starts **before** modeling with data validation.
5. *Being explicit* in your code **prevents disasters**.

6. Documentation and reproducible scripts **protect** you from *mistakes*.
7. *Joins* are one of the most dangerous operations, **check them twice**.
8. **Data types** matter more than most people realize.
9. *Peer* review is **essential**.
10. ***Great analysts aren't those who make no mistakes, they are the ones who catch mistakes early.***