TL;DR

Step 1 – Install & Configure

Install R from CRAN and ensure it works (R, Rscript, Rtools as needed). Download and install the Cursor IDE from the official site. Open your R project or scripts directly in Cursor (or use the Cursor CLI).

Test Prompt Example:

> Write R code to load a CSV, remove NAs, summarize data, and create a ggplot histogram.

Step 2 – Generate Scripts

Use Cursor to generate R code for tasks like reading CSV files, ggplot2 plots, and data cleaning pipelines.

- Review and adjust prompts as needed.

Example Script:

```
data <- read.csv("sales_data.csv")
data <- na.omit(data)
summary(data)
```

Step 3 – Reuse & Automate

Save templates for recurring tasks to streamline workflow.

- Reusable snippets for cleaning, transforming, and summarizing data.

Step 4 – Review & Execute

Validate all generated code in your R environment.

- Adjust parameters, libraries, and variable names.
- Cursor assists code generation; review every output for accuracy.

Cursor Setup Tips

- Set default libraries: `dplyr`, `ggplot2`, `data.table`.
- Customize templates for repeated tasks.
- Ensure IDE integration and syntax highlighting.
- Avoid version mismatches and PATH issues.

Step-by-Step R Workflow with Cursor

Step 1: Import Data

```
library(readr)
sales_data <- read_csv("sales_data.csv")
head(sales_data)
```

- Always inspect column types and missing values.

Step 2: Clean & Transform Data

```
clean_data <- na.omit(sales_data)
clean_data$Region <- as.factor(clean_data$Region)
names(clean_data) <- tolower(names(clean_data))
```

- Save these snippets as templates for consistency.

Step 3: Generate EDA Insights

```
library(ggplot2)
ggplot(clean_data, aes(x = Sales)) + geom_histogram(binwidth = 1000) +
theme_minimal()
summary(clean_data)
```

Step 4: Build Models

```
model <- lm(Sales ~ MarketingSpend + Region, data = clean_data)
summary(model)
```

- Classification templates: decision trees, logistic regression.

Step 5: Automate Reports

```
write.csv(clean_data, "clean_sales_data.csv")
ggsave("sales_distribution.png")
```

Quick Checklist

1. Verify Data Import
2. Clean and Transform Data
3. Perform EDA
4. Save Reusable Code & Templates
5. Validate Outputs

Time Savings Comparison

Task	Traditional Time	Cursor Time Saved
Data cleaning (100k rows)	20 min	7 min
Plotting & EDA	15 min	5 min
Script generation	10 min	3 min

Advanced Features

- Reusable templates for repeated tasks
- Multi-step pipelines
- Git integration for version control

Best Practices

1. Keep code modular using functions.
2. Validate all generated code step by step.
3. Document workflows and save runbooks.

Example Function:

```
clean_dataset <- function(df) {
  df <- na.omit(df)
  names(df) <- tolower(names(df))
  return(df)
}
cleaned_data <- clean_dataset(raw_data)
```

Validation:

```
head(cleaned_data)
str(cleaned_data)
```