```r
# ---- Install roadDB ----
# Github installation
install.packages("devtools")
devtools::install_github("sommergeo/roadDB")

# CRAN installation (will be made available later)
# install.packages(roadDB)

# ---- Load library ----
# Load the library
library(roadDB)

# ---- Install external packages ----
# Install packages (needs to be done only once)
install.packages("tidyverse")
install.packages("sf")
install.packages("tmap")
install.packages("rcarbon")

# ---- Load dependencies ----
# Load libraries (every time you run this tutorial)
library(roadDB)
library(tidyverse)
library(sf)
library(tmap)
library(rcarbon)

# ---- First query ----
# Now you are ready to run your first query and inspect the results directly in the R console:
my_query <- road_get_localities(cultural_period='Middle Paleolithic')

# ---- Argument examples ----
# Single argument
road_get_localities(cultural_period='MSA')

# Multiple arguments
road_get_localities(cultural_period='MSA', countries='South Africa')

# Lists of arguments
road_get_localities(cultural_period=c('MSA','LSA'), countries=c('South Africa','Namibia'))

# ---- Helper functions ----
# Examples
help(road_get_localities)
help(road_get_lithic_typology)
help(road_get_dates)

# Examples of listing valid argument values
road_list_argument_values("countries")
road_list_argument_values("cultural_periods")
road_list_argument_values("human_genus")

# Example summarizing term appearance
road_summarize_archaeology('egg shell')
head(road_summarize_archaeology('egg shell'), 4)

# ---- Example #1: Localities ----
# Query all localities
localities <- road_get_localities()
head(localities)  # head() shows the first 6 rows of a dataset

# Query selected localities
african_msa_localities <- road_get_localities(continents = "Africa", cultural_periods = "MSA")
head(african_msa_localities)
```

```
# Load geospatial libraries
library(sf)     # for spatial objects
library(tmap)   # for visualization

# Use coordinate columns to create a spatial dataset
african_msa_sf <- st_as_sf(african_msa_localities,
                           coords = c("coord_x", "coord_y"),
                           crs = 4326)  # WGS84 (lat/lon)

# Choose mapping engine
tmap_mode("plot")  # static map mode

# Create map
tm_shape(african_msa_sf) +
  tm_basemap("OpenStreetMap") +
  tm_dots()

# ---- Technocomplex map ----
road_list_argument_values("technocomplexes")

# Query Mousterian sites
mousterian <- road_get_localities(technocomplexes = c("MP/ Mousterian - Eurasia", "MP/ Mousterian
- Levant"))

# Create spatial dataset
mousterian_sf <- st_as_sf(mousterian,
                          coords = c("coord_x", "coord_y"),
                          crs = 4326)  # WGS84 (lat/lon)

# Display map
tmap_mode("plot")  # static map mode

tm_shape(mousterian_sf) +
  tm_basemap("OpenStreetMap") +
  tm_dots(
    col = "technocomplexes",
    palette = "Set2") +
  tm_layout(legend.outside = TRUE)

# ---- Example #2: Assemblages ----
# We can query all assemblages in ROAD
assemblages <- road_get_assemblages()

# Or we can make more specific queries
lithic_typology <- road_get_lithic_typology()
lithic_raw_mat <- road_get_lithic_raw_material()
organic_tools <- road_get_organic_tools()
symbolic <- road_get_symbolic_artifacts()
feature <- road_get_feature()
mix <- road_get_miscellaneous_finds()
people <- road_get_human_remains()
animals <- road_get_paleofauna()
plants <- road_get_plantremains()

# ---- Lithic Tools ----
techno <- road_list_argument_values("technocomplexes")

# Filter for Mousterian technocomplexes
df_mousterian <- techno[grepl("mousterian", techno$technocomplex, ignore.case = TRUE), ]

# Gather lithic typology data for Mousterian contexts
mous_tools <- road_get_lithic_typology(technocomplexes = df_mousterian)

# List available tool types
```

```
road_list_argument_values("tool_list")

# Gather Levallois technology examples
mous_levallois <- road_get_lithic_typology(
  technocomplexes = df_mousterian,
  tool_list = "levallois"
)

# Label data sources
mous_levallois$source <- "Levallois"
mous_tools$source     <- "Mousterian"

# Combine data
combined <- rbind(mous_levallois, mous_tools)

# Plot comparison
ggplot(combined, aes(x = fct_infreq(country),
                     fill = source)) +
  geom_bar(position = "dodge") +
  coord_flip() +
  labs(
    title = "Levallois vs. Mousterian",
    x = NULL,
    y = "Assemblages (n=)",
    fill = "Key"
  ) +
  scale_y_continuous(breaks = scales::pretty_breaks(n = 10)) +
  theme_minimal(base_size = 12)

# ---- Ostrich Egg Shells ----
road_summarize_archaeology('ostrich egg shell')

# Query from miscellaneous finds
eggshell_misc <- road_get_miscellaneous_finds()
eggshell_misc <- eggshell_misc[grepl("ostrich egg shell",
eggshell_misc$miscellaneous_finds_material), ]

# Query from symbolic artifacts
eggshell_symbolic <- road_get_symbolic_artifacts()
eggshell_symbolic <- eggshell_symbolic[grepl("ostrich egg shell",
eggshell_symbolic$symbolic_artifacts_material), ]

# Combine datasets
eggshell_combined <- merge(
  eggshell_misc,
  eggshell_symbolic,
  by = c("locality_id", "assemblage_id", "continent", "subcontinent", "country",
         "locality_types", "coord_x", "coord_y", "assemblage_name", "categories",
         "age_min", "age_max", "cultural_periods", "technocomplexes"),
  all = TRUE
)

# Plot ostrich egg shell distribution
ggplot(eggshell_combined, aes(y = cultural_periods)) +
  geom_bar() +
  labs(
    title = "Ostrich egg shells",
    x = "Number of assemblages",
    y = "Cultural period"
  )

# ---- Example #3: Dates ----
# Load all dates
all_dates <- road_get_dates()
```

```
# Filter for "Hohle Fels"
hohle_fels_dates <- all_dates[grepl("Hohle Fels", all_dates$locality_id), ]

# Filter for archaeological layer Vb and symbolic artifacts
hohle_fels_Vb <- hohle_fels_dates[grepl("AH Vb", hohle_fels_dates$archlayer), ]
hohle_fels_Vb <- hohle_fels_Vb[grepl("symbolic artifacts", hohle_fels_Vb$categories), ]

# Load radiocarbon library
library(rcarbon)

# Calibrate radiocarbon ages
hohle_fels_Vb_cal <- calibrate(
  x = hohle_fels_Vb$age,
  errors = hohle_fels_Vb$positive_standard_deviation,
  calCurves = 'intcal20'
)

# Display calibrated ages
summary(hohle_fels_Vb_cal)

# Visualize first date
plot(hohle_fels_Vb_cal, 1)

# Visualize all calibrated dates
multiplot(
  hohle_fels_Vb_cal,
  decreasing = TRUE,
  rescale = TRUE,
  HPD = TRUE,
  label.pos = 0.9,
  label.offset = -200
)
```