

Limpieza de Topónimos y Parseo de Coordenadas y WKT (julia)

Lina María Quintero Fonseca

2026-02-22

0.1 Ejercicio 1

```
j_eval(r"-(
registro_crudo = "    pnn seRRania de la mACaRena    "

texto_sin_espacios = strip(registro_crudo)
texto_titulo = titlecase(lowercase(texto_sin_espacios))
registro_oficial = replace(texto_titulo, "Pnn" => "PNN")

longitud = length(registro_oficial)

println("El área protegida limpia es: $registro_oficial y su nombre tiene $longitud caracteres
) -")
```

```
julia> registro_crudo = "    pnn seRRania de la mACaRena    "
"    pnn seRRania de la mACaRena    "

julia> texto_sin_espacios = strip(registro_crudo)
"pnn seRRania de la mACaRena"

julia> texto_titulo = titlecase(lowercase(texto_sin_espacios))
"Pnn Serrania De La Macarena"

julia> registro_oficial = replace(texto_titulo, "Pnn" => "PNN")
"PNN Serrania De La Macarena"

julia> longitud = length(registro_oficial)
```

```
julia> println("El área protegida limpia es: $registro_oficial y su nombre tiene $longitud caracteres")
El área protegida limpia es: PNN Serrania De La Macarena y su nombre tiene 28 caracteres.
```

0.2 Ejercicio 2

```
j_eval(r"-(
using Printf

mensaje_sensor = "ALERTA_SISMICA:1.221,-77.359:PROFUNDIDAD_5KM"

partes = split(mensaje_sensor, ":")
bloque_coords = partes[2]

latlon = split(bloque_coords, ",")
latitud = parse(Float64, latlon[1])
longitud = parse(Float64, latlon[2])

punto_wkt = @sprintf("POINT(% .4f % .4f)", longitud, latitud)

function miles(n::Integer)
    s = string(n)
    rev = reverse(s)
    trozos = [rev[i:min(i+2, lastindex(rev))] for i in 1:3:lastindex(rev)]
    return reverse(join(trozos, ","))
end

area_m2 = 1250000
area_fmt = miles(area_m2)

println("WKT generado: $punto_wkt")
println("Área afectada: $area_fmt m2")
) -")
```

```
julia> using Printf

julia> mensaje_sensor = "ALERTA_SISMICA:1.221,-77.359:PROFUNDIDAD_5KM"
"ALERTA_SISMICA:1.221,-77.359:PROFUNDIDAD_5KM"

julia> partes = split(mensaje_sensor, ":")
```

```
3-element Vector{SubString{String}}:  
"ALERTA_SISMICA"  
"1.221,-77.359"  
"PROFOUNDIDAD_5KM"  
  
julia> bloque_coords = partes[2]  
"1.221,-77.359"  
  
julia> latlon = split(bloque_coords, ",")  
2-element Vector{SubString{String}}:  
"1.221"  
"-77.359"  
  
julia> latitud = parse(Float64, latlon[1])  
1.221  
  
julia> longitud = parse(Float64, latlon[2])  
-77.359  
  
julia> punto_wkt = @sprintf("POINT(%f %f)", longitud, latitud)  
"POINT(-77.3590 1.2210)"  
  
julia> function miles(n::Integer)  
    s = string(n)  
    rev = reverse(s)  
    trozos = [rev[i:min(i+2, lastindex(rev))] for i in 1:3:lastindex(rev)]  
    return reverse(join(trozos, ","))  
end  
miles (generic function with 1 method)  
  
julia> area_m2 = 1250000  
1250000  
  
julia> area_fmt = miles(area_m2)  
"1,250,000"  
  
julia> println("WKT generado: $punto_wkt")  
WKT generado: POINT(-77.3590 1.2210)  
  
julia> println("Área afectada: $area_fmt m2")  
Área afectada: 1,250,000 m2
```

1 1 - ¿Por qué es una mala práctica depender únicamente de la función “Reemplazar” (.replace() / gsub()) en lugar de usar funciones dedicadas como strip() o trimws()?

Por qué NO depender solo de “Reemplazar” para quitar espacios. Depender solo de reemplazo (replace/gsub) para limpiar espacios de borde en bases de datos grandes es mala práctica porque:

- Es fácil definir un patrón incompleto y dejar casos fuera (tabs, saltos de línea, combinaciones de whitespace).
- Se puede terminar “limpiando de más” y modificar espacios internos que sí son semánticos (topónimos compuestos).
- Funciones dedicadas como strip()/trimws() son más claras, más legibles y reducen riesgo de errores de regex/patrones.

1.1 2 — Impacto de NO convertir texto a Float antes de WKT

Si se construye el WKT concatenando texto sin convertir a número decimal:

Se pierde una validación crítica: parse(Float64, ...) detecta entradas corruptas (caracteres extra, espacios invisibles).

Se puede producir WKT con formatos inconsistentes (p.ej. notación inesperada o caracteres no numéricos), y luego ST_GeomFromText en PostGIS puede fallar o interpretar mal.

Para cálculos de distancia en PostGIS/QGIS, el problema típico no es “la distancia sale distinta por ser texto”, sino que la geometría puede quedar inválida / mal parseada / con precisión no controlada. Convertir y formatear asegura consistencia (4 decimales) y trazabilidad.

1.2 Pregunta General — Split en R vs Julia/Python

En R, strsplit() devuelve siempre una lista (aunque partas un solo string), por eso debes usar [[1]] o unlist() para extraer el vector de strings y poder indexar normal. En Julia (y Python), split() devuelve directamente una colección indexable (Vector/lista) de substrings, así que no necesitas ese paso extra.