`\L`, `\U`, `\l`, `\u` — these are **special case conversion escapes** used inside **sed** (and also in some other tools like `perl`) **only when using `-E` (extended regex)** or advanced features.

In short:

Escape	Meaning	Example
`\L`	Lowercase all following characters until `\E`	`s/.*/\L&/` turns the whole line into lowercase
`\U`	Uppercase all following characters until `\E`	`s/.*/\U&/` turns the whole line into uppercase
`\l`	Lowercase **only** the next character	`s/.*/\l&/` lowercases the first character
`\u`	Uppercase **only** the next character	`s/.*/\u&/` uppercases the first character

`\E` is used to "end" the effect of `\L` or `\U`, if you want (sometimes optional if at end of replacement).

Tiny Examples

Suppose a line says:

nginx
CopyEdit
```
hELLO wORLD
```

- `s/.*/\L&/` → `hello world`
  (everything lowercase)
- `s/.*/\U&/` → `HELLO WORLD`
  (everything uppercase)

- `s/(^.)/\u\1/` → `HELLO WORLD` → `HELLO WORLD`
  (uppercases only first character — but already upper)

- `s/(^.)/\l\1/` → `hELLO WORLD` → `hELLO WORLD`
  (lowercases first character)

What is gsub in awk?

- `gsub` stands for **global substitution**.
- It **searches** for **all matches** of a pattern in a string and **replaces** them.
- It modifies the **string in place** (awk strings are mutable!).
- Syntax:

awk
CopyEdit
```
gsub(REGEX, REPLACEMENT, [TARGET])
```

Part	Meaning
`REGEX`	The pattern you want to search
`REPLACEMENT`	What you want to replace it with
`TARGET` (optional)	The string variable or field (default is `$0`, the whole line)

Detailed Parts

- If **`TARGET` is omitted**, `gsub` operates on the **whole line** (`$0`).
- It returns the **number of substitutions made** (you can use it if needed).

Examples

1. Basic substitution on entire line

bash
CopyEdit
```
awk '{ gsub(/apple/, "orange"); print }' input.txt
```

- Replace every `apple` with `orange` in every line.
- If a line has `"apple apple"`, both are changed.

---

2. Substitution inside a variable

bash
CopyEdit
```
awk '{ str="hello world"; gsub(/world/, "awk", str); print str
}'
```

Output:

nginx
CopyEdit
```
hello awk
```

- It changes only the `str` variable, not the whole line.

---

3. Substitution inside a specific field

bash
CopyEdit

```
awk '{ gsub(/foo/, "bar", $2); print }'
```

- Replace `foo` with `bar` **only in the second field** (`$2`).

Behavior points:

- `gsub` **changes the string** directly (in-place).
- It **always replaces all occurrences** (not just first).
- **If no match is found**, string remains unchanged.
- It **returns** number of replacements.

Example:

bash
CopyEdit
```
awk '{ n = gsub(/a/, "x", $1); print n, $1 }'
```

- Counts how many `a` replaced in `$1`, prints number and modified `$1`.

Special things inside gsub:

- You can use **&** inside replacement → it means "the matched text".

Example:

bash

CopyEdit
```
awk '{ gsub(/dog/, "&s"); print }'
```

turns:

rust
CopyEdit
```
dog -> dogs
hotdog -> hotdogs
```

Tiny difference with `sub`

Function	Behavior
`sub`	Replace **only first** match
`gsub`	Replace **all** matches