COMANDO LINUX · HOJA DE REFERENCIA

GREP · Cheatsheet

Busca texto, filtra resultados y encuentra información en segundos.

¿Para qué sirve?

grep recorre la entrada línea a línea y muestra las líneas que coinciden con un patrón. Por defecto distingue mayúsculas, busca coincidencias parciales y no modifica archivos.

Sintaxis

```
grep [opciones] 'patrón' archivo
comando | grep [opciones] 'patrón'
```

opciones modifican la búsqueda · **patrón** es el texto o regex · **archivo** es el fichero donde buscar · **|** filtra la salida de otro comando

Opciones más utilizadas

OPCIÓN	QUÉ HACE
`-i`	Ignora diferencias entre mayúsculas y minúsculas.
`-n`	Añade el número de línea original.
`-v`	Invierte la búsqueda: muestra las líneas que NO coinciden.
`-r / -R`	Busca recursivamente; -R sigue enlaces simbólicos encontrados.
`-c`	Cuenta líneas coincidentes (no todas las apariciones).
`-w / -x`	Palabra completa / patrón que ocupa toda la línea.
`-o`	Muestra solo la parte que coincide.
`-E / -F`	Regex extendida / búsqueda de texto literal.
`-B N / -A N / -C N`	N líneas antes / después / a ambos lados.
`-m N / -q`	Máximo N líneas coincidentes / búsqueda sin imprimir resultados.

Expresiones regulares esenciales

PATRÓN	SIGNIFICADO	EJEMPLO
`^`	Comienzo de línea	'^DB_'
`$`	Final de línea	' 500$'
`|`	O lógico (con -E)	'500|503'
`[0-9]{3}`	Tres dígitos (con -E)	'[0-9]{3}$'
`.*`	Cualquier secuencia de caracteres	'ERROR.*Redis'

PISTA Escribe patrones entre comillas simples. Puedes unir opciones como -rni. El resultado 0 significa coincidencia; 1, ninguna; 2, error de ejecución.

01 / PRÁCTICA CON ARCHIVO REAL

API · logs/api.log

Filtra peticiones y códigos HTTP sin procesar archivos completos.

CONTENIDO DEL ARCHIVO

```
2026-11-09 08:00:10 INFO GET /api/users 200
2026-11-09 08:00:20 INFO GET /api/products 200
2026-11-09 08:01:10 WARN GET /api/orders 404
2026-11-09 08:02:00 ERROR POST /api/login 500
2026-11-09 08:02:30 INFO GET /api/users 200
2026-11-09 08:03:00 ERROR GET /api/orders 503
```

Comandos y salidas reales

01. Filtrar por método Selecciona únicamente las peticiones POST.

```
$ grep 'POST' logs/api.log
2026-11-09 08:02:00 ERROR POST /api/login 500
```

02. Buscar un código concreto El signo $ obliga a que 404 esté al final de la línea.

```
$ grep ' 404$' logs/api.log
2026-11-09 08:01:10 WARN GET /api/orders 404
```

03. Localizar errores del servidor Con -E y la barra vertical buscamos HTTP 500 o 503.

```
$ grep -E ' (500|503)$' logs/api.log
2026-11-09 08:02:00 ERROR POST /api/login 500
2026-11-09 08:03:00 ERROR GET /api/orders 503
```

04. Contar errores -c muestra cuántas líneas cumplen la condición.

```
$ grep -Ec ' (500|503)$' logs/api.log
2
```

05. Extraer solo los códigos -o imprime solo la coincidencia; [0-9]{3} son tres dígitos.

```
$ grep -oE '[0-9]{3}$' logs/api.log
200
200
404
500
200
503
```

06. Excluir las respuestas correctas -v invierte el filtro: aquí oculta las respuestas 200.

```
$ grep -v ' 200$' logs/api.log
2026-11-09 08:01:10 WARN GET /api/orders 404
2026-11-09 08:02:00 ERROR POST /api/login 500
2026-11-09 08:03:00 ERROR GET /api/orders 503
```

07. Limitar los resultados -m 2 detiene la lectura tras dos líneas coincidentes.

```
$ grep -m 2 'GET' logs/api.log
2026-11-09 08:00:10 INFO GET /api/users 200
2026-11-09 08:00:20 INFO GET /api/products 200
```

02 / PRÁCTICA CON ARCHIVO REAL

Usuarios · usuarios.csv

Localiza registros, nombres y roles usando coincidencias y expresiones.

CONTENIDO DEL ARCHIVO

```
id,nombre,email,rol,estado
1,ana,ana@ejemplo.com,admin,activo
2,anabel,anabel@ejemplo.com,usuario,activo
3,pedro,pedro@ejemplo.com,admin,inactivo
4,luis,luis@ejemplo.com,usuario,activo
5,ANA,ana2@ejemplo.com,usuario,inactivo
```

Comandos y salidas reales

01. Buscar un fragmento grep busca en toda la línea, no solo en la columna nombre.

```
$ grep 'ana' datos/usuarios.csv
1,ana,ana@ejemplo.com,admin,activo
2,anabel,anabel@ejemplo.com,usuario,activo
5,ANA,ana2@ejemplo.com,usuario,inactivo
```

02. Ignorar mayúsculas -i encuentra ana y ANA como nombre, sin importar su escritura.

```
$ grep -i ',ana,' datos/usuarios.csv
1,ana,ana@ejemplo.com,admin,activo
5,ANA,ana2@ejemplo.com,usuario,inactivo
```

03. Buscar el nombre exacto en la columna 2 ^ + id numérico + coma + ana + coma; evita incluir anabel.

```
$ grep -Ei '^[0-9]+,ana,' datos/usuarios.csv
1,ana,ana@ejemplo.com,admin,activo
5,ANA,ana2@ejemplo.com,usuario,inactivo
```

04. Encontrar administradores Las comas ayudan a identificar el valor del campo rol.

```
$ grep ',admin,' datos/usuarios.csv
1,ana,ana@ejemplo.com,admin,activo
3,pedro,pedro@ejemplo.com,admin,inactivo
```

05. Cruzar rol y estado Encontramos administradores cuyo estado final es activo.

```
$ grep ',admin,activo$' datos/usuarios.csv
1,ana,ana@ejemplo.com,admin,activo
```

06. Contar administradores -c cuenta líneas con el campo admin.

```
$ grep -c ',admin,' datos/usuarios.csv
2
```

07. Encontrar inactivos $ exige que la palabra inactivo cierre el registro.

```
$ grep ',inactivo$' datos/usuarios.csv
3,pedro,pedro@ejemplo.com,admin,inactivo
5,ANA,ana2@ejemplo.com,usuario,inactivo
```

03 / PRÁCTICA CON ARCHIVO REAL

Configuración · app.env

Encuentra claves concretas y valores de configuración.

CONTENIDO DEL ARCHIVO

```
APP_ENV=production
DEBUG=false
PORT=8080
DB_HOST=db.internal
DB_USER=aplicacion
REDIS_HOST=redis.internal
ERROR_REPORTING=true
```

Comandos y salidas reales

01. Buscar parámetros de base de datos ^ fija DB_ al comienzo de la línea.

```
$ grep '^DB_' config/app.env
DB_HOST=db.internal
DB_USER=aplicacion
```

02. Localizar la configuración de Redis Muestra la línea que contiene REDIS.

```
$ grep 'REDIS' config/app.env
REDIS_HOST=redis.internal
```

03. Buscar una clave concreta Busca la línea que empieza exactamente por DB_HOST=.

```
$ grep '^DB_HOST=' config/app.env
DB_HOST=db.internal
```

04. Coincidencia de línea completa -x exige que toda la línea sea DEBUG=false.

```
$ grep -x 'DEBUG=false' config/app.env
DEBUG=false
```

05. Buscar texto literal -F trata el punto como un punto real, no como regex.

```
$ grep -F 'db.internal' config/app.env
DB_HOST=db.internal
```

06. Alternativas en una búsqueda -E permite combinar PORT con las claves terminadas en _HOST.

```
$ grep -E '(^PORT|_HOST)=' config/app.env
PORT=8080
DB_HOST=db.internal
REDIS_HOST=redis.internal
```

07. Mostrar la línea de cada ajuste -n indica dónde están las variables de DB y Redis.

```
$ grep -nE '^(DB_|REDIS_)' config/app.env
4:DB_HOST=db.internal
5:DB_USER=aplicacion
6:REDIS_HOST=redis.internal
```