



-Polars



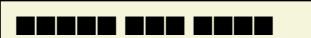
██████:	███████████████████ Polars
████:	███████████████████
████:	████████
███████:	2025-10-29



1. ██████████ ██████████ ██████████
2. ██████████ ██████████ ██████████ ██████████
3. ██████████ ██████████ ██████████
4. ██████████ ██████████ ██████████ ██████████
5. ██████████ ██████████ ██████████ ██████████
6. ██████████ ██████████ ██████████
7. ██████████ ██████████ ██████████
8. ██████████ ██████████ ██████████ ██████████

1.

 Polars .

		
starts_with()		df.filter(pl.col('name').str.starts_with('A'))
ends_with()		df.filter(pl.col('email').str.ends_with('.com'))
contains()		df.filter(pl.col('text').str.contains('word'))
contains_any()		df.filter(pl.col('text').str.contains_any(['a', 'b']))
len_chars()		df.filter(pl.col('text').str.len_chars() > 10)

:

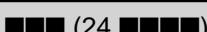
```
df.filter(pl.col('content').str.starts_with('Very'))
.select('content')
.head()
```

2.


Polars .

		
to_date()	Date	<code>pl.col('date_str').str.to_date(format='%Y-%m-%d')</code>
to_time()	Time	<code>pl.col('time_str').str.to_time(format='%H:%M:%S')</code>
to_datetime()	Datetime	<code>pl.col('dt').str.to_datetime(format='%Y-%m-%d %H:%M')</code>
strptime()		<code>pl.col('str').str.strptime(pl.Date, '%Y-%m-%d')</code>

:

		
%Y	 (4  <td>2024</td>	2024
%m	 (2  <td>01-12</td>	01-12
%d	 (2  <td>01-31</td>	01-31
%H	 (24  <td>00-23</td>	00-23
%M		00-59
%S		00-59

3.

 Polars .

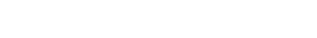
		
slice()		.str.slice(3, 5)
extract()	 Regex (	.str.extract(r'(\d+)')
extract_all()		.str.extract_all(r'(\w+)')
extract_groups()		.str.extract_groups(r'(\d+)-(\d+) ')

 Regex :

	
\d+	
[A-Za-z] +	
\w+	 ( ,  , _)
\s+	
^start	
end\$	

4.

 Polars 

		
	<code>strip_chars()</code>	 / 
	<code>replace_all()</code>	
	<code>to_lowercase()</code>	
	<code>to_uppercase()</code>	
	<code>to_titlecase()</code>	Title Case
	<code>pad_start()</code>	
	<code>pad_end()</code>	

7.

- **literal:** `literal=True` **Regex** - `re.compile()`!
- **NULL values:** `fill_null()` **Series** `str.fillna()`
- **Case Sensitivity:** `str.lower()` **Series** `str.upper()`. `str.replace(?i)` **Regex** `re.compile()`
- **strip, lower, replace:** `str.strip().str.lower().str.replace()`
- **starts_with:** `str.len_chars().describe()` **Series** `str.str_len()`
- **starts_with:** `(starts_with)` **Series** `str.startswith()`
- **NULL values** `str.isnull()`
- **literal=True** `literal=True` **Series** `str.replace()`
- **Regex** `re.compile()` - `re.match()`



A horizontal row of ten solid black squares, followed by a black colon character (:).

- ✓ [REDACTED]
 - ✓ [REDACTED]
 - ✓ [REDACTED] Regex [REDACTED]
 - ✓ [REDACTED]
 - ✓ [REDACTED]
 - ✓ [REDACTED]

A horizontal row of ten solid black squares, followed by a black colon character (:).

- Polars: <https://pola-rs.github.io/polars/>
 - Regex Tester: <https://regex101.com/>
 - Python strftime: <https://strftime.org/>
 - Polars Discord Community

■ ■ ■ ■ ■ PDF ■ ■