Cheat Sheet - Polars

Introduction

Library

```
import polars as pl
```

DataFrame

```
df = pl.DataFrame({
   "id": [1, 2, 3, 4]
   ,"name": ["Alice", "Bob", "Charlie", "Diana"]
   ,"age": [25, 30, 35, 28]
   ,"score": [85.5, 90.0, 78.3, 92.1]
})
```

LazyFrame

```
lf = pl.LazyFrame({
   "id": [1, 2, 3, 4]
   ,"name": ["Alice", "Bob", "Charlie", "Diana"]
   ,"age": [25, 30, 35, 28]
   ,"score": [85.5, 90.0, 78.3, 92.1]
})
```

Input/Output

Parquet

```
lf = pl.scan_parquet(
   source = filepath
   ,n_rows = integer / None (default)
)

lf.sink_parquet(
   path = filepath
   ,compression = string (default: 'zstd')
   ,compression_level = int / None (default)
   ,engine = EngineType (default: 'auto')
)
```

CSV

```
lf = pl.scan_csv(
   source = filepath
   ,has_header = bool (default: True)
   ,separator = string (default: ',')
   ,quote_char = string (default: '"')
   ,decimal_comma = bool (default: False)
   ,schema_overrides = dictionary / None (default)
   ,skip_rows = integer (default: 0)
   ,skip_lines = integer (default: 0)
)

lf.sink_csv(
   path = filepath
   ,include_header = bool (default: True)
   ,separator = string (default: ',')
   ,line_terminator = string (default: '\n')
   ,quote_char = string (default: '"')
   ,decimal_comma = bool (default: 'False')
   ,engine = EngineType (default: 'auto')
)
```

Selection

Rows

```
lf.filter(
   (pl.col('age') > 27) | (pl.col('name') == 'Bob')
)

lf.filter(
   (pl.col('age') > 27) & (pl.col('name') == 'Bob')
)

lf.filter(
   pl.col('age') > 27
   ,pl.col('name') == 'Bob'
)
```

Columns

```
lf.select(
   pl.col(['name', 'age', 'score'])
)

Special expressions: pl.all(), pl.exclude('column-
name-01', ..., 'column-name-
n'), pl.col([pl.Int64, pl.Float64, pl.String])
```

Generation

Index Column

```
lf = lf.with_row_index(name = 'index')
```

Custom Column(s)

```
lf.with_columns(
   age_round_01 = pl.col('age').round(0)
   ,pl.col('age').round(0).alias('age_round_02')
   ,pl.col('age').round(0).name.suffix('_round_03')
   ,pl.col('age').name.prefix('original_')
)
```

Conditional Custom Column(s)

```
lf.select(
   pl.when(
      pl.col('score') > 80
   )
   .then('Pass')
   .when(
      pl.col('score') > 60
   )
   .then('Pending')
   .otherwise('Fail')
   .alias('result')
)
```