test area

Table of contents

load

```
from pathlib import Path
import os
import seaborn as sb
import pandas as pd
import sys
import numpy as np
# from datetime import datetime
# from connection_helper import sql

from pandas_plots import tbl, pls, ven , hlp
import duckdb as ddb

hlp.show_package_version(["matplotlib_venn", "dataframe_image", "plotly", "kaleido", "seaborn"], )
df = sb.load_dataset('taxis')

# if os.getenv("RENDERER") in ('png', 'svg'):
#     os.environ['THEME'] = 'light'
# else:
#     os.environ['THEME'] = 'dark'

hlp.set_theme("dark")

dir_db=Path("C://temp") if hlp.get_os(hlp.OperatingSystem.WINDOWS) else Path(os.path.expanduser("~/tmp"))

file_db_clin = dir_db/'workflow/2025-10-20_data_clin.duckdb'

if not file_db_clin.is_file():
    print(f"File does not exist: {file_db_clin}")
    raise FileNotFoundError

print(hlp.get_os(hlp.OperatingSystem.MAC))
```

```
🐍 3.12.9 | 📦 matplotlib_venn: 0.11.10 | 📦 dataframe_image: 0.2.7 | 📦 plotly: 6.2.0 | 📦 kaleido: 1.0.0 | 📦 seaborn: 0.13.2 | 📦 pandas: 2.3.1
| 📦 numpy: 1.26.4 | 📦 duckdb: 1.3.2 | 📦 pandas-plots: 0.20.1 | 📦 connection-helper: 0.13.1
True
```

```
con = ddb.connect(file_db_clin, read_only=True)
```

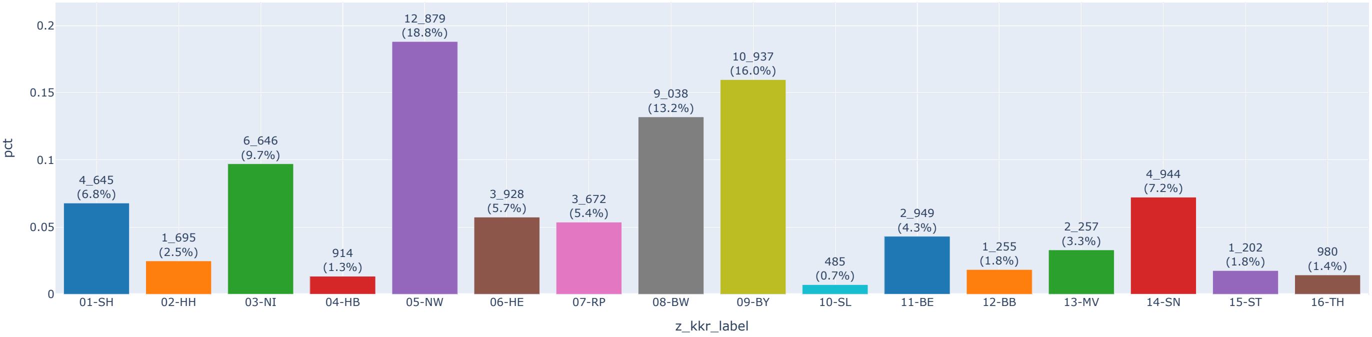
```
db_delay = con.sql("""--sql
    select
            first(z_kkr_label) as z_kkr_label,
            first(z_first_treatment_after_days) as z_first_treatment_after_days,
            first(z_first_treatment) as z_first_treatment
    from Tumor
    where ifnull(z_first_treatment,'') <> ''
    and z_dy = 2023 and z_icd10_3d = 'C81'
    group by z_tum_id
    """)
```

pls

bars

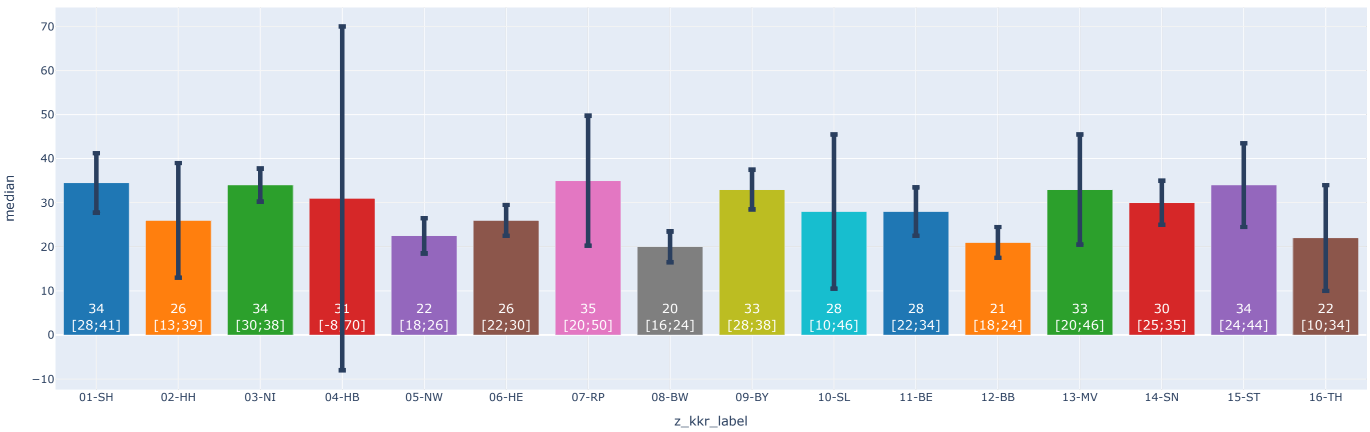
```
_ = pls.plot_bars(
    db_delay.to_df().iloc[:,:2],
    # height=400,
    # width=1000,
    # use_ci=True,
    # ci_agg='median',
)
_ = pls.plot_bars(
    db_delay.to_df().iloc[:,:2],
    height=600,
    use_ci=True,
    ci_agg='median',
)
```

[z_first_treatment_after_days] by [z_kkr_label], n=1_724 (68_426)



[z_first_treatment_after_days] by [z_kkr_label], NULL excluded, n=1_724)

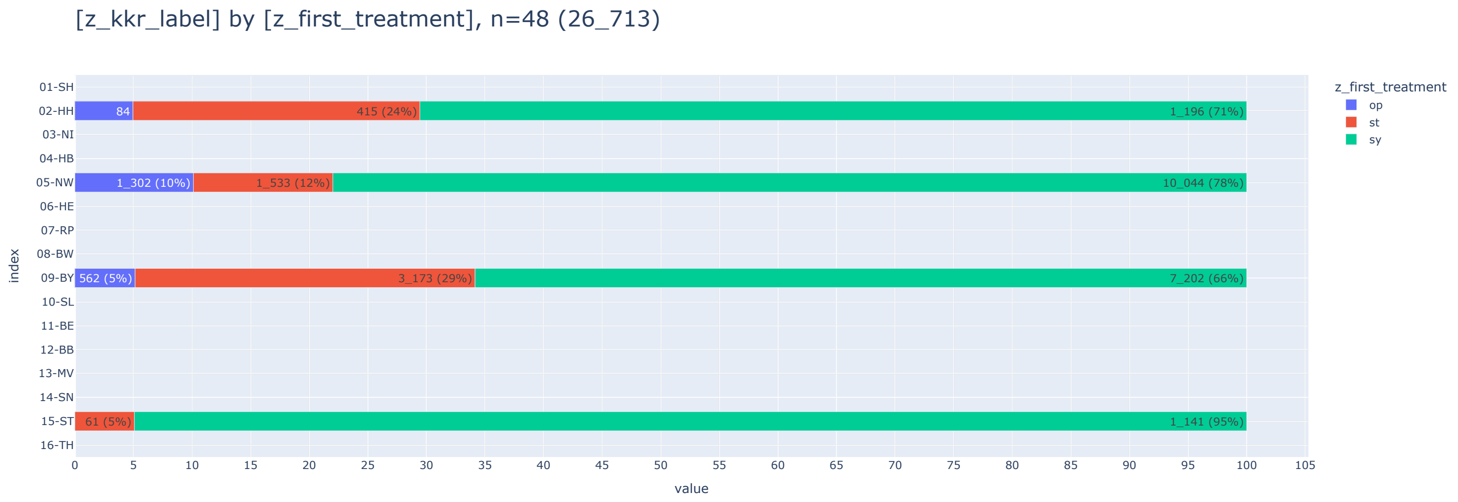
ci(95) on medians




```
_df = db_delay.filter("left(z_kkr_label,2)::int8 in (2,5,9,15)").to_df().iloc[:,[0,2,1]]
display(_df)
pls.plot_stacked_bars(
    _df,
    height=600,
    width=1600,
    orientation="h",
    relative=True,
    show_pct_bar=True,
    kkr_col="z_kkr_label",
    # renderer="png",
    )
```

	z_kkr_label	z_first_treatment	z_first_treatment_after_days
0	15-ST	sy	22
1	09-BY	sy	2
2	05-NW	op	0
3	05-NW	sy	20
4	05-NW	st	130
...	...	...	...
656	02-HH	op	0
657	05-NW	sy	28
658	09-BY	sy	14
659	05-NW	op	0
660	09-BY	sy	17

661 rows × 3 columns

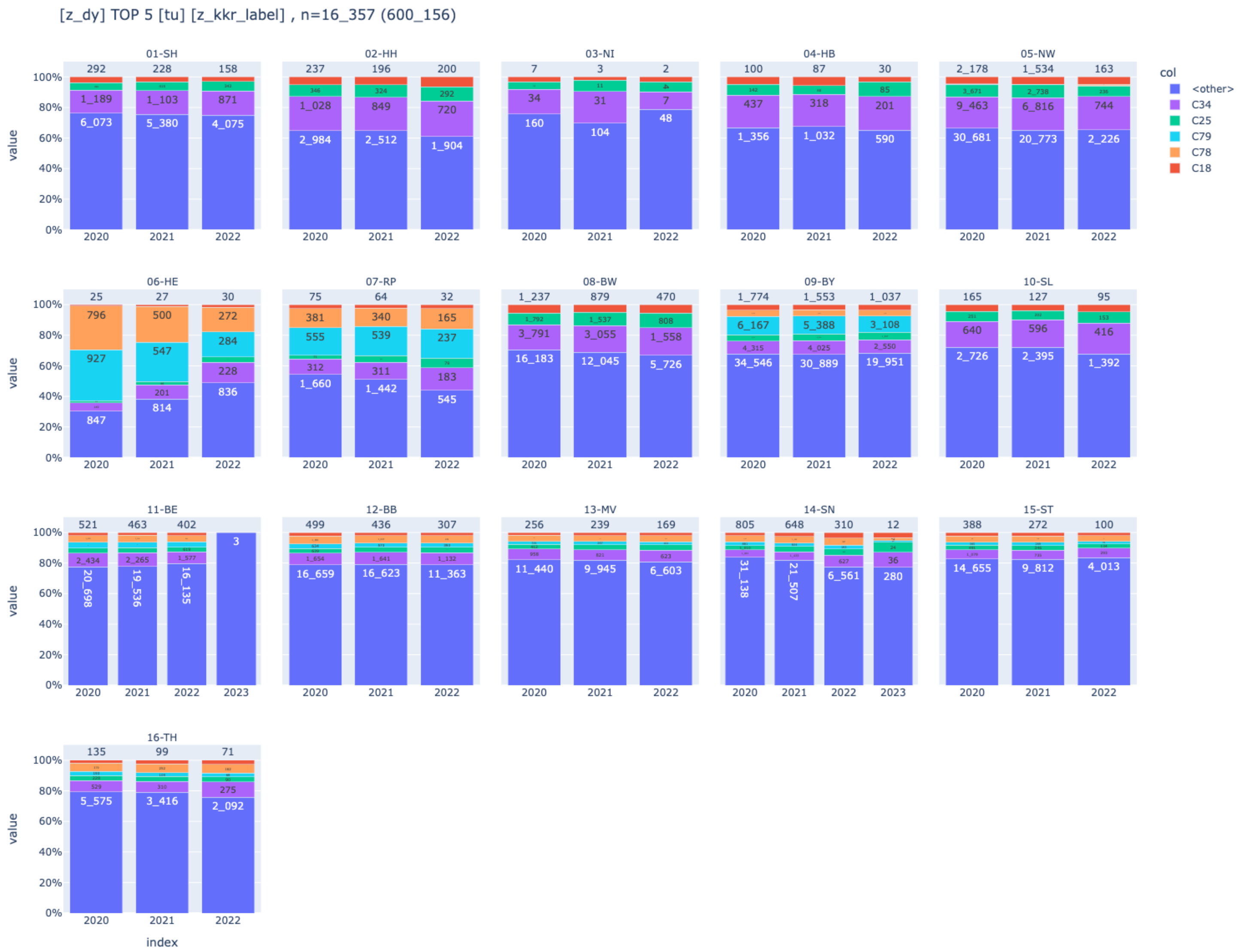



```python
df_facets = pd.read_csv("assets/facets.csv", sep=";")#.astype({"z_dy": "Int64"}).astype({"z_dy": str})
display(df_facets[:None])
print(df_facets.dtypes)

_df = df_facets[
    [
        "z_dy",
        "tu",
        "z_kkr_label",
        "cnt",
    ]
]#[:100]
# _df


l=pls.plot_facet_stacked_bars(
    _df,
    top_n_color=5,
    # subplots_per_row=3,
    # top_n_facet=15,
    # top_n_index=3,
    show_other=True,
    sort_values_color=True,
    # sort_values_facet=True,
    relative=True,
    # show_pct=True,
    annotations=True,
    subplot_size=300,
    subplots_per_row=5,
    # renderer="",
    # show_pct=True,
    )
# l

# ll = pls.plot_facet_stacked_bars(
#     _df,
#     top_n_color=5,
#     # subplots_per_row=8,
#     # top_n_facet=15,
#     # top_n_index=3,
#     show_other=True,
#     sort_values_color=True,
#     # sort_values_facet=True,
#     relative=True,
#     annotations=True,
#     )
# # ll
```

	z_dy	tu	z_kkr_label	cnt
0	2020	06	13-MV	1
1	2020	30	13-MV	1
2	2020	###	13-MV	21
3	2020	18	11-BE	1
4	2020	18.	13-MV	1
...	...	...	...	...
16352	2023	S37	14-SN	1
16353	2023	S72	11-BE	1
16354	2023	Z22	14-SN	1
16355	2023	Z85	14-SN	1
16356	2023	Z96	14-SN	1

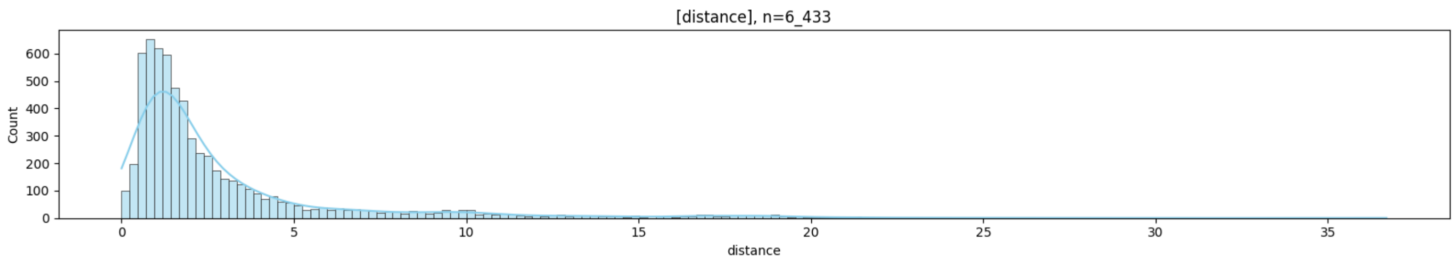
16357 rows × 4 columns

```
z_dy            int64
tu             object
z_kkr_label    object
cnt             int64
dtype: object
```



histo

```
_=pls.plot_histogram_large(
    df.distance,
    # nbins=50,
    height=300,
    summary=True
)
```




```
column   | count |  min  | lower |  q25  | median | mean  |  q75  | upper |  max   |  std  |  cv   |    sum
---------+-------+-------+-------+-------+--------+-------+-------+-------+--------+-------+-------+-----------
distance | 6_433 | 0.000 | 0.000 | 0.980 |  1.640 | 3.025 | 3.210 | 6.550 | 36.700 | 3.828 | 1.266 | 19_457.360
```

```
# _df = df["distance"]
# # _df#.to_frame()
# # _df=df[["fare","distance"]]
# pls.plot_histogram(
#     _df,
#     height=300,
#     width=1000,
#     precision=0,
#     orientation="v",
#     histnorm="",
#     nbins=-1,
#     # barmode="overlay",
#     # renderer="png",
#     # png_path=Path(".local/box5.png"),
#     summary=True,
```

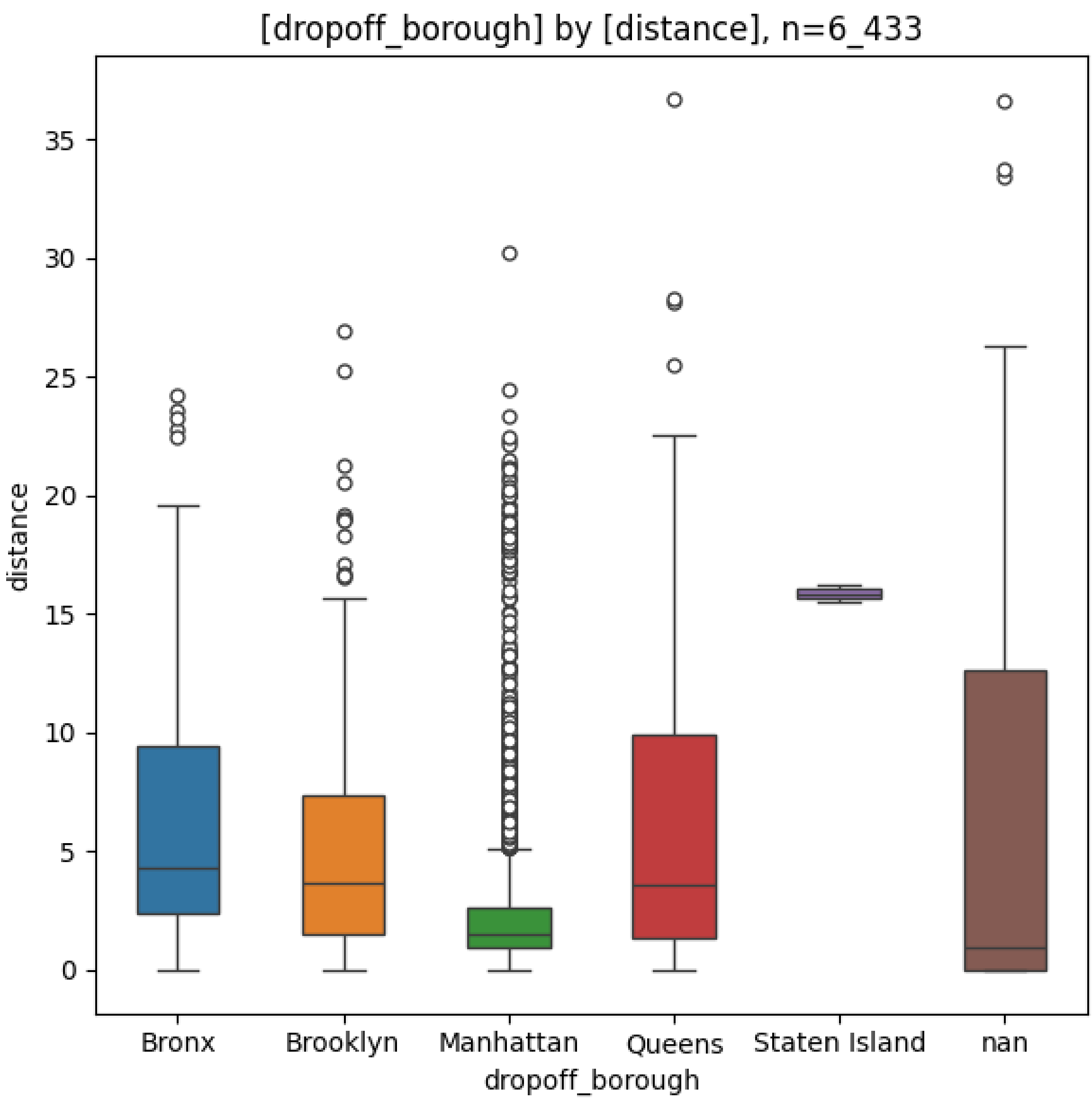
```
#     caption="test lol xd-lmao",
# )
```

box

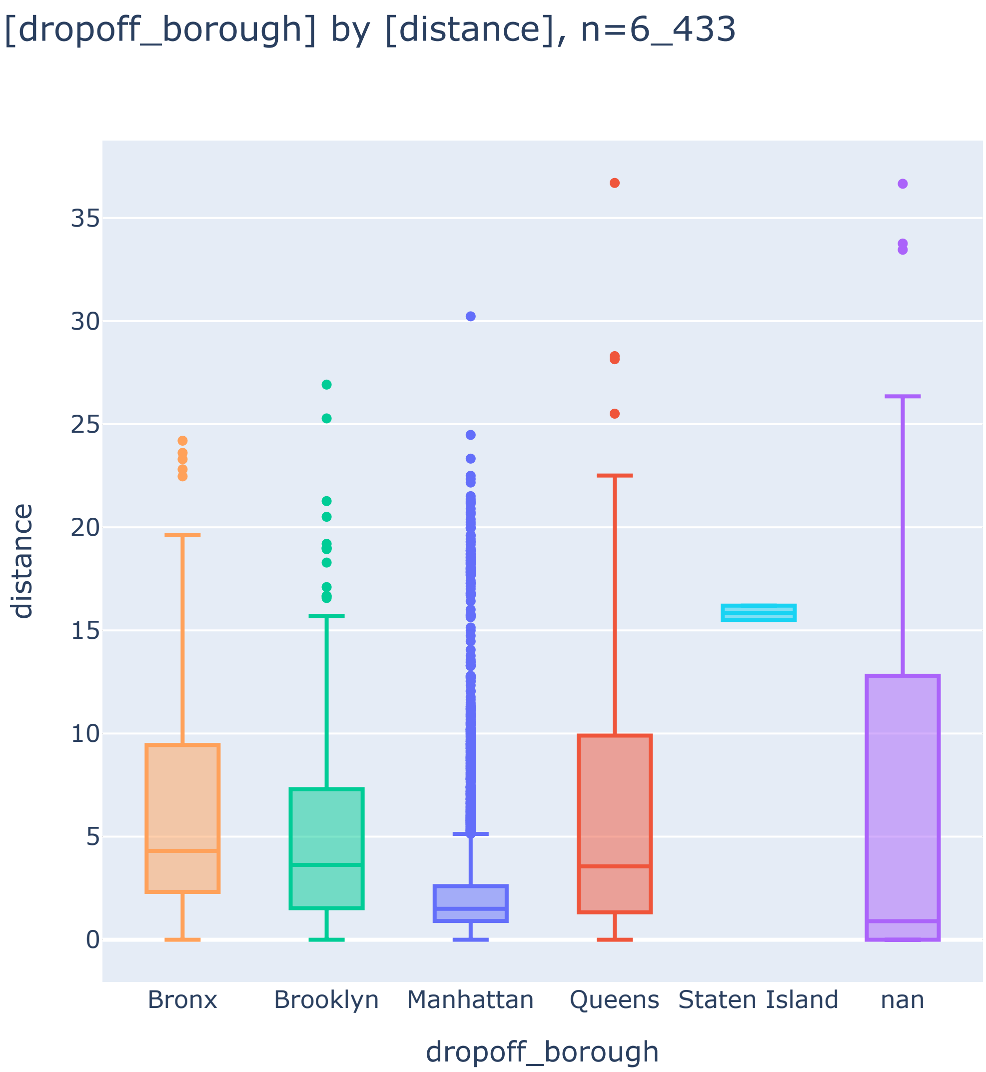
```
# _=pls.plot_box(
#     # df["distance"],
#     df[df["dropoff_borough"].isna()].distance,
#     height=200,
#     # violin=True,
#     # use_log=True,
# )
```

```
_df = df[["dropoff_borough","distance"]]
print(_df)
_=pls.plot_boxes_large(
    _df,
    width=600,
    # violin=True,
    # use_log=True,
)
_=pls.plot_boxes(
    _df,
    width=600,
    # violin=True,
    # use_log=True,
)
```

```
     dropoff_borough  distance
0          Manhattan      1.60
1          Manhattan      0.79
2          Manhattan      1.37
3          Manhattan      7.70
4          Manhattan      2.16
...              ...       ...
6428       Manhattan      0.75
6429           Bronx     18.74
6430        Brooklyn      4.14
6431        Brooklyn      1.12
6432        Brooklyn      3.85

[6433 rows x 2 columns]
```



```
column   | count | min  | lower | q25  | median | mean | q75  | upper |  max  | std  |  cv  |    sum
---------+-------+------+-------+------+--------+------+------+-------+-------+------+------+----------
distance | 6_433 | 0.00 |  0.00 | 0.98 |   1.64 | 3.02 | 3.21 |  6.55 | 36.70 | 3.83 | 1.27 | 19_457.36


column        | count |  min  | lower |  q25  | median | mean  |  q75  | upper |  max  |  std  |  cv  |    sum
--------------+-------+-------+-------+-------+--------+-------+-------+-------+-------+-------+------+----------
Bronx         |   137 |  0.00 |  0.00 |  2.36 |   4.31 |  6.61 |  9.44 | 19.62 | 24.20 |  5.95 | 0.90 |    905.88
Brooklyn      |   501 |  0.00 |  0.00 |  1.53 |   3.63 |  5.03 |  7.30 | 15.70 | 26.92 |  4.49 | 0.89 |  2_519.28
Manhattan     | 5_206 |  0.00 |  0.00 |  0.91 |   1.50 |  2.39 |  2.60 |  5.13 | 30.23 |  2.91 | 1.21 | 12_447.68
Queens        |   542 |  0.00 |  0.00 |  1.34 |   3.56 |  5.96 |  9.89 | 22.51 | 36.70 |  5.89 | 0.99 |  3_230.13
Staten Island |     2 | 15.51 | 15.51 | 15.68 |  15.86 | 15.86 | 16.03 | 16.20 | 16.20 |  0.49 | 0.03 |     31.71
nan           |    45 |  0.00 |  0.00 |  0.00 |   0.90 |  7.17 | 12.60 | 26.35 | 36.66 | 10.71 | 1.49 |    322.68
```



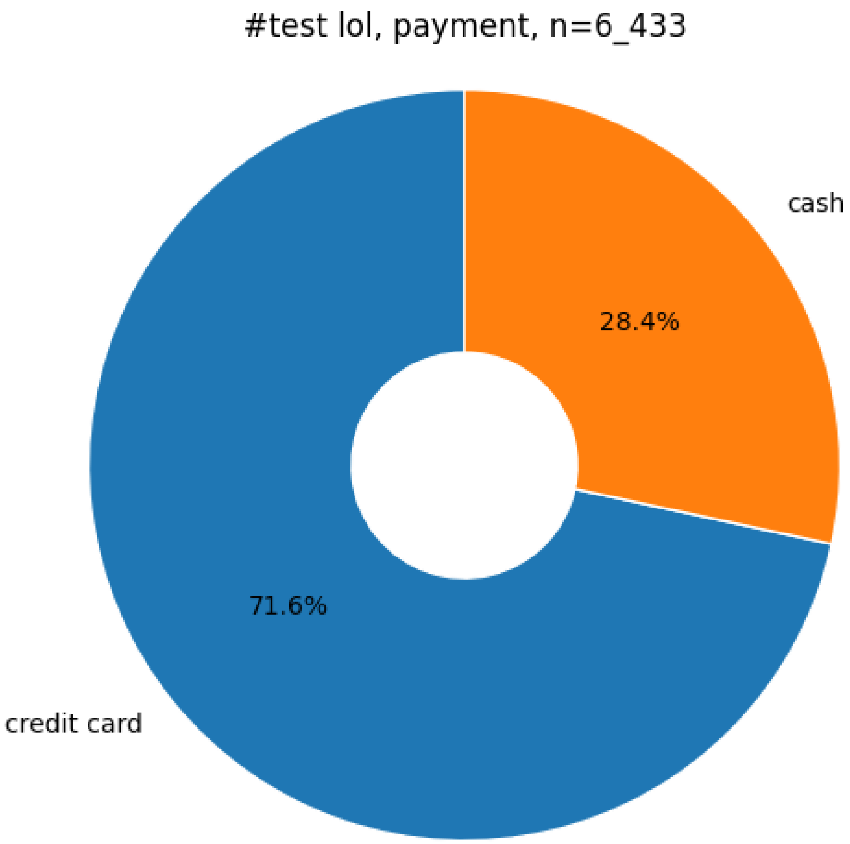

```
column   | count | min  | lower | q25  | median | mean | q75  | upper |  max  | std  |  cv  |    sum
---------+-------+------+-------+------+--------+------+------+-------+-------+------+------+----------
distance | 6_433 | 0.00 |  0.00 | 0.98 |   1.64 | 3.02 | 3.21 |  6.55 | 36.70 | 3.83 | 1.27 | 19_457.36


column        | count |  min  | lower |  q25  | median | mean  |  q75  | upper |  max  |  std  |  cv  |    sum
--------------+-------+-------+-------+-------+--------+-------+-------+-------+-------+-------+------+----------
Bronx         |   137 |  0.00 |  0.00 |  2.36 |   4.31 |  6.61 |  9.44 | 19.62 | 24.20 |  5.95 | 0.90 |    905.88
Brooklyn      |   501 |  0.00 |  0.00 |  1.53 |   3.63 |  5.03 |  7.30 | 15.70 | 26.92 |  4.49 | 0.89 |  2_519.28
Manhattan     | 5_206 |  0.00 |  0.00 |  0.91 |   1.50 |  2.39 |  2.60 |  5.13 | 30.23 |  2.91 | 1.21 | 12_447.68
Queens        |   542 |  0.00 |  0.00 |  1.34 |   3.56 |  5.96 |  9.89 | 22.51 | 36.70 |  5.89 | 0.99 |  3_230.13
Staten Island |     2 | 15.51 | 15.51 | 15.68 |  15.86 | 15.86 | 16.03 | 16.20 | 16.20 |  0.49 | 0.03 |     31.71
nan           |    45 |  0.00 |  0.00 |  0.00 |   0.90 |  7.17 | 12.60 | 26.35 | 36.66 | 10.71 | 1.49 |    322.68
```


```
# pls.plot_box(
#     df.distance,
#     height=400,
#     violin=False,
#     x_min=-2,
#     x_max=50,
#     # summary = False
# )

# _df = df[["dropoff_borough", "distance"]]
# # _df["dropoff_borough"] = _df["dropoff_borough"].astype(str)
# display(_df.dtypes)
# display(_df)
# _=pls.plot_boxes(
#     _df,
#     width=1600,
```

```
#     annotations=False,
#     precision=4,
# )
```

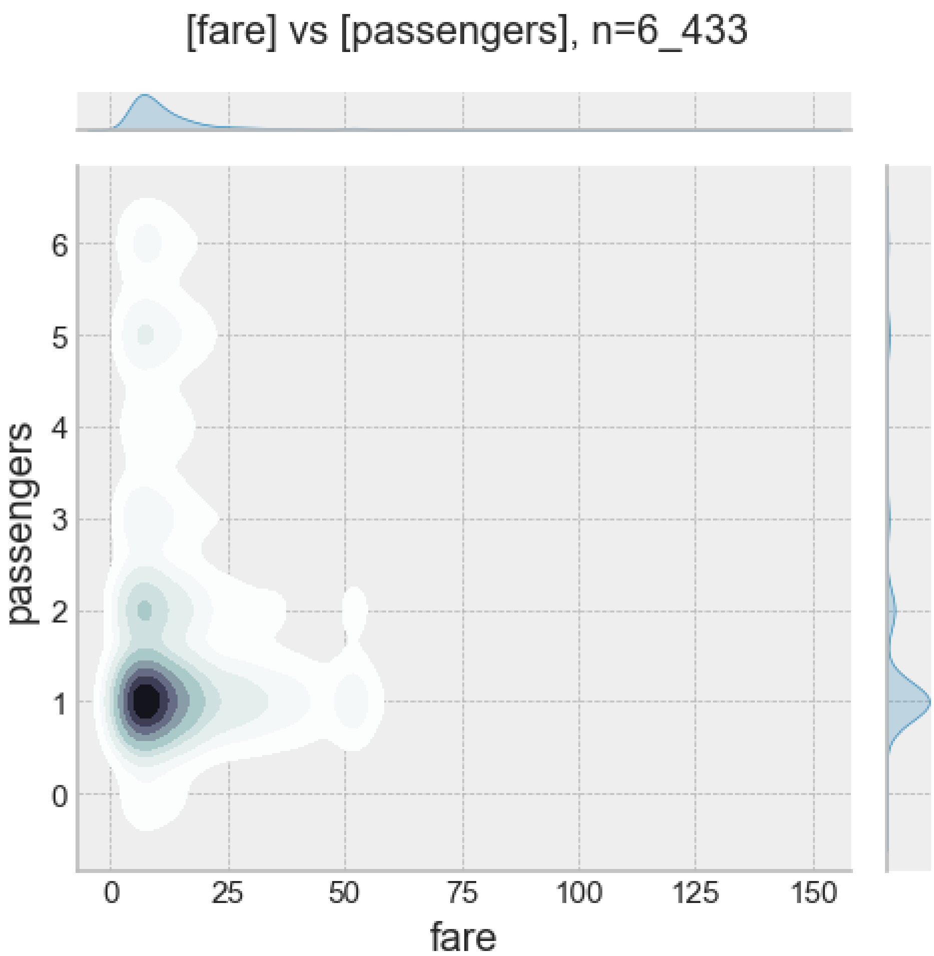
pie

```
df["payment"]
pls.plot_pie(
    data=df["payment"],
    caption="test lol",
    donut_size=0.3,
    # precision=3,
)
```



join

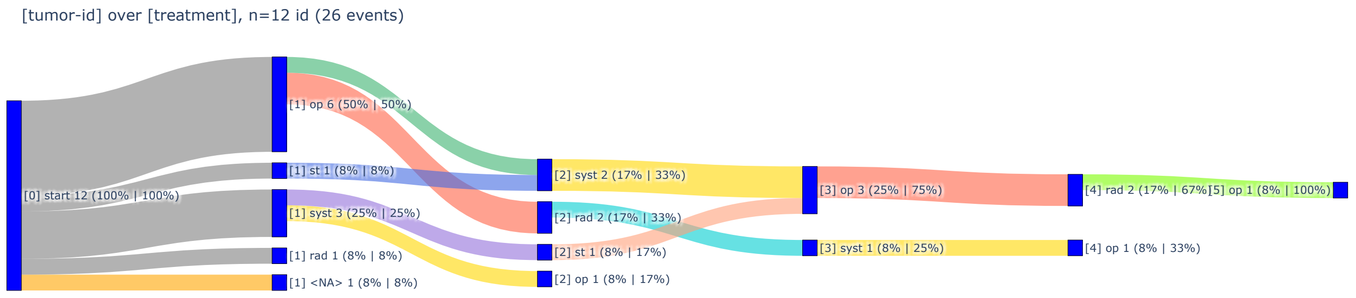
```
_df = df[["fare", "passengers"]]
# pls.plot_joint(_df, precision=0, size=15, kind="reg", caption="")
# pls.plot_joint(_df, precision=0, size=15, kind="hex", png_path=Path(".local/box6.png"))
pls.plot_joint(_df, precision=0, size=5, kind="kde",)
# pls.plot_joint(_df, precision=0, size=5, kind="hist",)
# pls.plot_joint(_df, precision=0, size=5, kind="kde", png_path=Path(".local/box7.png"))
```



sankey

```
pls.plot_sankey(
    width=2000,
    font_size=15,
    # renderer="png",
)
```

```
--- Using demo data (data_demo) ---
   tumor-id diagnosis date treatment
0         1      2020-01-01        op
1         1      2021-02-01      syst
2         1      2022-03-01        op
3         1      2023-04-01       rad
4         1      2024-05-01        op
5         2      2010-01-01      syst
6         2      2011-02-01        st
7         2      2012-03-01        op
8         2      2013-04-01       rad
9         3      2015-01-01        op
10        3      2016-02-01       rad
11        3      2017-03-01      syst
12        3      2018-04-01        op
13        4      2005-01-01        st
14        4      2006-02-01      syst
15        4      2007-03-01        op
16        5      2019-01-01        op
17        5      2020-02-01       rad
18        6      2021-01-01      syst
19        6      2022-02-01        op
20        7
21        7
22        8      2025-01-01        op
23        9      2025-02-01        op
24       10      2025-03-01      syst
25       11      2025-04-01       rad
26       12      2025-05-01        op
----------------------------------
```



tbl

print_summary()

```
_=tbl.print_summary(df)
```

```
column     | count |  min  | lower |  q25   | median |  mean  |  q75   | upper  |   max   |  std   |  cv   |    sum
-----------+-------+-------+-------+--------+--------+--------+--------+--------+---------+--------+-------+------------
passengers | 6_433 |     0 |     0 |  1.000 |  1.000 |  1.539 |  2.000 |      3 |       6 |  1.204 | 0.782 |      9_902
distance   | 6_433 | 0.000 | 0.000 |  0.980 |  1.640 |  3.025 |  3.210 |  6.550 |  36.700 |  3.828 | 1.266 |  19_457.360
fare       | 6_433 | 1.000 | 1.000 |  6.500 |  9.500 | 13.091 | 15.000 | 27.540 | 150.000 | 11.552 | 0.882 |  84_214.870
tip        | 6_433 | 0.000 | 0.000 |  0.000 |  1.700 |  1.979 |  2.800 |  6.960 |  33.200 |  2.449 | 1.237 |  12_732.320
tolls      | 6_433 | 0.000 | 0.000 |  0.000 |  0.000 |  0.325 |  0.000 |  0.000 |  24.020 |  1.415 | 4.351 |   2_092.480
total      | 6_433 | 1.300 | 1.300 | 10.800 | 14.160 | 18.518 | 20.300 | 34.550 | 174.820 | 13.816 | 0.746 | 119_124.970
```

describe_df()

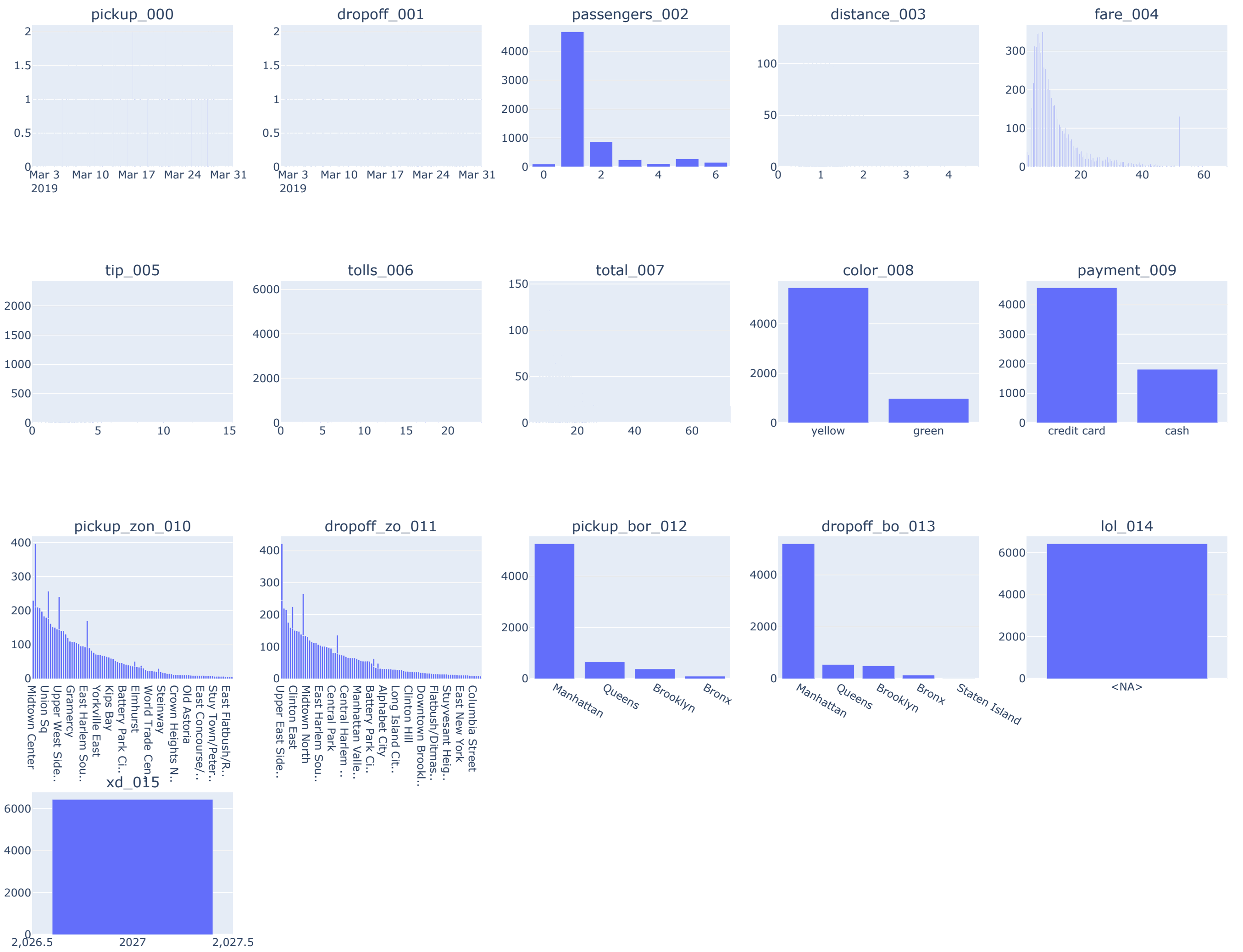
```
df["lol"] = np.nan
df["xd"]=2027
df["lol"] = df["lol"].astype("float64")

display(df)

tbl.describe_df(
    df,
    # "taxis",
    use_columns=False,
    # renderer="svg",
    top_n_uniques=10,
    top_n_chars_in_columns=10,
    top_n_chars_in_index=15,
)
```

	pickup	dropoff	passengers	distance	fare	tip	tolls	total	color	payment	pickup_zone	dropoff_zone	pickup_borough	dropoff_borough	lol	xd
0	2019-03-23 20:21:09	2019-03-23 20:27:24	1	1.60	7.0	2.15	0.0	12.95	yellow	credit card	Lenox Hill West	UN/Turtle Bay South	Manhattan	Manhattan	NaN	2027
1	2019-03-04 16:11:55	2019-03-04 16:19:00	1	0.79	5.0	0.00	0.0	9.30	yellow	cash	Upper West Side South	Upper West Side South	Manhattan	Manhattan	NaN	2027
2	2019-03-27 17:53:01	2019-03-27 18:00:25	1	1.37	7.5	2.36	0.0	14.16	yellow	credit card	Alphabet City	West Village	Manhattan	Manhattan	NaN	2027
3	2019-03-10 01:23:59	2019-03-10 01:49:51	1	7.70	27.0	6.15	0.0	36.95	yellow	credit card	Hudson Sq	Yorkville West	Manhattan	Manhattan	NaN	2027
4	2019-03-30 13:27:42	2019-03-30 13:37:14	3	2.16	9.0	1.10	0.0	13.40	yellow	credit card	Midtown East	Yorkville West	Manhattan	Manhattan	NaN	2027
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
6428	2019-03-31 09:51:53	2019-03-31 09:55:27	1	0.75	4.5	1.06	0.0	6.36	green	credit card	East Harlem North	Central Harlem North	Manhattan	Manhattan	NaN	2027
6429	2019-03-31 17:38:00	2019-03-31 18:34:23	1	18.74	58.0	0.00	0.0	58.80	green	credit card	Jamaica	East Concourse/Concourse Village	Queens	Bronx	NaN	2027
6430	2019-03-23 22:55:18	2019-03-23 23:14:25	1	4.14	16.0	0.00	0.0	17.30	green	cash	Crown Heights North	Bushwick North	Brooklyn	Brooklyn	NaN	2027
6431	2019-03-04 10:09:25	2019-03-04 10:14:29	1	1.12	6.0	0.00	0.0	6.80	green	credit card	East New York	East Flatbush/Remsen Village	Brooklyn	Brooklyn	NaN	2027
6432	2019-03-13 19:31:22	2019-03-13 19:48:02	1	3.85	15.0	3.36	0.0	20.16	green	credit card	Boerum Hill	Windsor Terrace	Brooklyn	Brooklyn	NaN	2027

6433 rows × 16 columns

```
🔵 *** df: <unknown> ***
🟣 shape: (6_433, 16)
🟣 duplicates: 0
🟠 column stats numeric

column     | count |  min  | lower |   q25    |  median  |   mean   |   q75    | upper  |   max   |  std   |  cv   |     sum
-----------+-------+-------+-------+----------+----------+----------+----------+--------+---------+--------+-------+------------
passengers | 6_433 |     0 |     0 |    1.000 |    1.000 |    1.539 |    2.000 |      3 |       6 |  1.204 | 0.782 |       9_902
distance   | 6_433 | 0.000 | 0.000 |    0.980 |    1.640 |    3.025 |    3.210 |  6.550 |  36.700 |  3.828 | 1.266 |  19_457.360
fare       | 6_433 | 1.000 | 1.000 |    6.500 |    9.500 |   13.091 |   15.000 | 27.540 | 150.000 | 11.552 | 0.882 |  84_214.870
tip        | 6_433 | 0.000 | 0.000 |    0.000 |    1.700 |    1.979 |    2.800 |  6.960 |  33.200 |  2.449 | 1.237 |  12_732.320
tolls      | 6_433 | 0.000 | 0.000 |    0.000 |    0.000 |    0.325 |    0.000 |  0.000 |  24.020 |  1.415 | 4.351 |   2_092.480
total      | 6_433 | 1.300 | 1.300 |   10.800 |   14.160 |   18.518 |   20.300 | 34.550 | 174.820 | 13.816 | 0.746 | 119_124.970
xd         | 6_433 | 2_027 | 2_027 | 2_027.000 | 2_027.000 | 2_027.000 | 2_027.000 |  2_027 |   2_027 |  0.000 |   N/A |  13_039_691
```

	pickup	dropoff	passengers	distance	fare	tip	tolls	total	color	payment	pickup_zone	dropoff_zone	pickup_borough	dropoff_borough	lol	xd
0	2019-03-23 20:21:09	2019-03-23 20:27:24	1	1.60	7.0	2.15	0.0	12.95	yellow	credit card	Lenox Hill West	UN/Turtle Bay South	Manhattan	Manhattan	NaN	2027
1	2019-03-04 16:11:55	2019-03-04 16:19:00	1	0.79	5.0	0.00	0.0	9.30	yellow	cash	Upper West Side South	Upper West Side South	Manhattan	Manhattan	NaN	2027
2	2019-03-27 17:53:01	2019-03-27 18:00:25	1	1.37	7.5	2.36	0.0	14.16	yellow	credit card	Alphabet City	West Village	Manhattan	Manhattan	NaN	2027



descr_db()

```
tbl.descr_db(db_delay, caption="delay", width=200)
```

delay 1_724, 3
("z_kkr_label, z_first_treatment_after_days, z_first_treatment")

z_kkr_label varchar	z_first_treatment_after_days int32	z_first_treatment varchar
03-NI	14	sy
05-NW	13	op
14-SN	90	sy

```
df_dsich = pd.read_csv("assets/dsich.csv", sep=";")#.astype({"z_dy": "Int64"}).astype({"z_dy": str})
display(df_dsich[:3])
# print(df_dsich.dtypes)
```

	id	bl	label	cnt
0	0	01-SH	7 - Histologie Primärtumor	14351
1	22	01-SH	3 - Todesbescheinigung (DCO)	1004
2	23	01-SH	5 - Zytologie	146

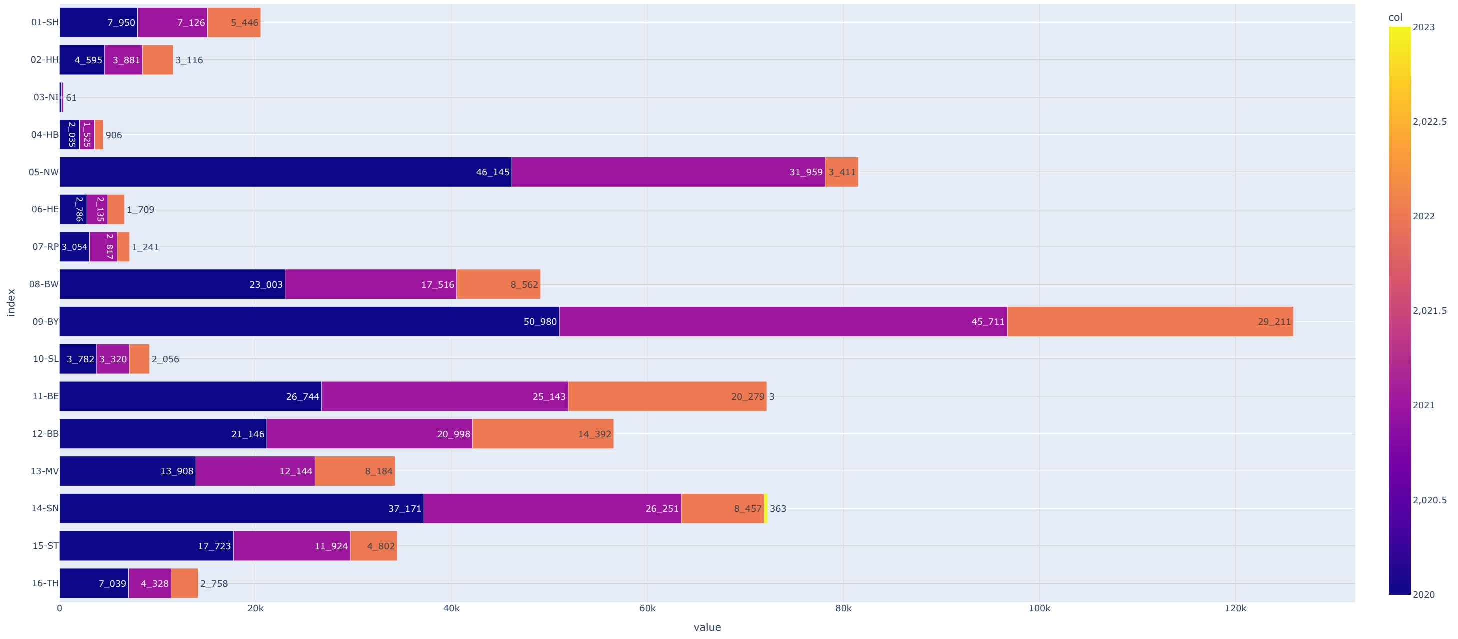
```
_df = df_dsich[
    [
        "bl",
```

```
        "label",
        "cnt",
    ]
].sort_values("bl")

_ = pls.plot_stacked_bars(
    _df,
    # swap=True,
    orientation="h",
    # show_total=True,
    # normalize=True,
    # relative=True,
    height=600,
    # top_n_color=5,
    # sort_values_index=True,
    sort_values_color=True,
    # top_n_index=3,
    # precision=2,
    # show_other=True,
    # top_n_index=2,
    kkr_col="bl",
)
_
```

[bl] by [label], n=160 (188_481)

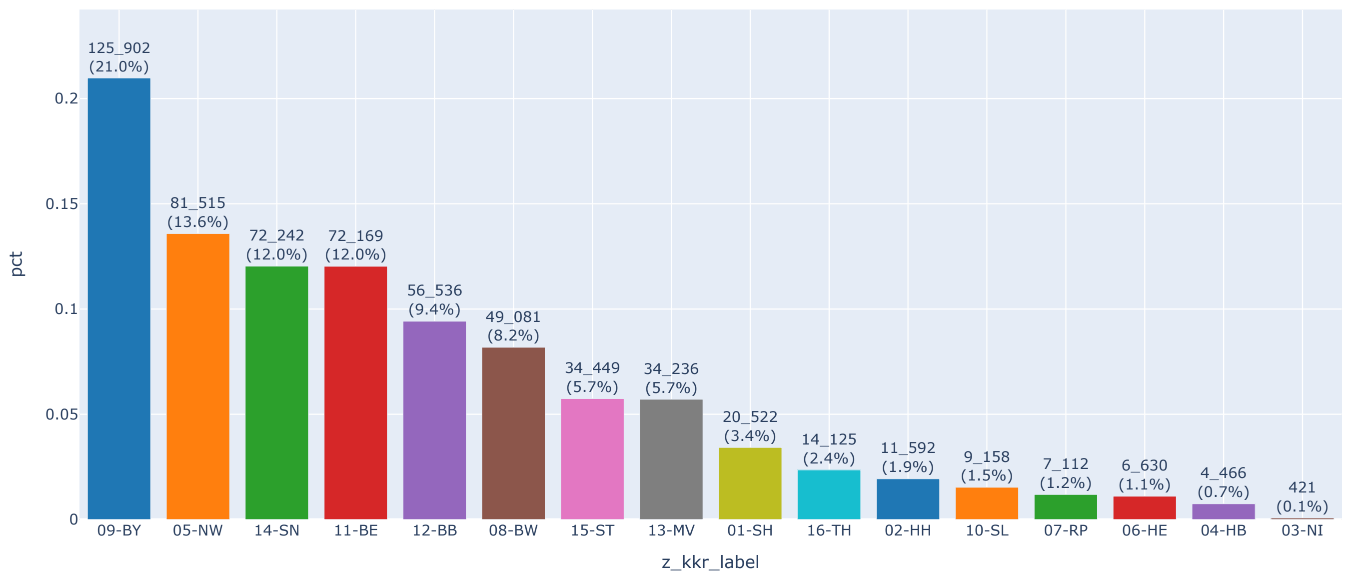

```
_df = df_facets[
    [
        "z_kkr_label",
        "z_dy",
        "cnt",
    ]
]#.astype({"z_dy": str})

_ = pls.plot_stacked_bars(
    _df,
    # swap=True,
    orientation="h",
    # show_total=True,
    # normalize=True,
    # relative=True,
    height=1000,
    top_n_color=10,
    # sort_values_index=True,
    # sort_values_color=True,
    # top_n_index=5,
    # precision=2,
    # show_other=True,
    # top_n_index=2,
    # renderer="png",
)
_
```

[z_kkr_label] by TOP10 [z_dy], n=50 (600_156)




```
_df = df_facets[
    [
        "z_kkr_label",
        "cnt",
    ]
]
_df

_ = pls.plot_bars(
    _df,
    height=600,
    width=1200,
    sort_values=True,
    # renderer="png",
)
```

[cnt] by [z_kkr_label], n=16_357 (600_156)

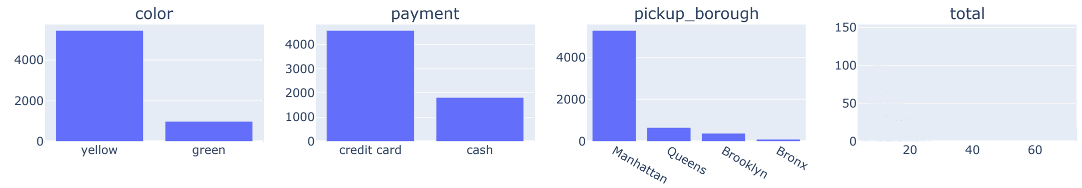



```
_df = df[["color","payment","pickup_borough","total"]]
_df
# _=pls.plot_stacked_bars(_df[["color","payment","total"]],)
tbl.describe_df(_df, "taxis")
```

```
🔵 *** df: taxis ***
🟣 shape: (6_433, 4)
🟣 duplicates: 4_803
🟠 column stats all (dtype | uniques | missings) [values]
- index [0, 1, 2, 3, 4,]
- color (object | 2 | 0 (0%)) ['green', 'yellow',]
- payment (object | 3 | 44 (1%)) ['<NA>', 'cash', 'credit card',]
```

```
- pickup_borough (object | 5 | 26 (0%)) ['<NA>', 'Bronx', 'Brooklyn', 'Manhattan', 'Queens',]
- total (float64 | 898 | 0 (0%)) [1.3, 3.3, 3.31, 3.8, 4.3,]
🟠 column stats numeric

column | count |  min  | lower |  q25   | median |  mean  |  q75   | upper  |   max   |  std   |  cv   |     sum
-------+-------+-------+-------+--------+--------+--------+--------+--------+---------+--------+-------+------------
total  | 6_433 | 1.300 | 1.300 | 10.800 | 14.160 | 18.518 | 20.300 | 34.550 | 174.820 | 13.816 | 0.746 | 119_124.970
```

	color	payment	pickup_borough	total
0	yellow	credit card	Manhattan	12.95
1	yellow	cash	Manhattan	9.30
2	yellow	credit card	Manhattan	14.16




```
db = con.from_df(df)
tbl.descr_db(db, caption="taxis",)
```

```
🗄 taxis    6_433, 16
    ("pickup, dropoff, passengers, distance, fare, tip, tolls, total, color, payment, pickup_zone, dropoff_zone, pickup_borough, dropoff_borough,
lol, xd")
```

pickup timestamp_ns	dropoff timestamp_ns	passengers int64	distance double	fare double	…	pickup_borough varchar	dropoff_borough varchar	lol double	xd int64
2019-03-23 20:21:09	2019-03-23 20:27:24	1	1.6	7.0	…	Manhattan	Manhattan	NULL	2027
2019-03-04 16:11:55	2019-03-04 16:19:00	1	0.79	5.0	…	Manhattan	Manhattan	NULL	2027
2019-03-27 17:53:01	2019-03-27 18:00:25	1	1.37	7.5	…	Manhattan	Manhattan	NULL	2027
3 rows									16 columns (9 shown)

show num

```
from pathlib import Path

_df=df.pivot_table(index="color", columns="payment", values="fare", aggfunc="sum", dropna=False)
tbl.show_num_df(
    _df,
    total_mode="sum",
    total_axis="y",
    data_bar_axis="",
    pct_axis="xy",
    precision=0,
    heatmap_axis="xy",
    # kpi_mode="max_min_x",
    total_exclude=True,
    # kpi_mode="min_max_xy",
    kpi_mode="rag_abs",
    kpi_rag_list=(100, 1000),
    # swap=True,
    font_size_td=12,
    font_size_th=14,
    # png_path=Path("test.png"),
)
```

payment	cash	credit card	nan	Total
color				
green	3_995 (4.7%) 🟥	9_774 (11.6%) 🟥	20 (0.0%) 🟩	13_788 (16.4%) 🟥
yellow	17_012 (20.2%) 🟥	52_907 (62.8%) 🟥	508 (0.6%) 🟨	70_427 (83.6%) 🟥

```
_df=df[["color", "payment", "fare"]]
tbl.pivot_df(
    _df,
    total_mode="sum",
    data_bar_axis="xy",
    pct_axis="xy",
    precision=0,
    heatmap_axis="",
    kpi_mode="max_min_x"
    # kpi_mode="min_max_xy",
    # kpi_mode="rag_abs",
    # kpi_rag_list=(100, 1000),
)
```

payment	nan	cash	credit card	Total
color				
green	20 (0.0%) 🟥	3_995 (4.7%) 🟥	9_774 (11.6%) 🟥	13_788 (16.4%) 🟥

payment	nan	cash	credit card	Total
color				
yellow	508 (0.6%)	17_012 (20.2%)	52_907 (62.8%)	70_427 (83.6%)
Total	528 (0.6%)	21_006 (24.9%)	62_681 (74.4%)	84_215 (100.0%)

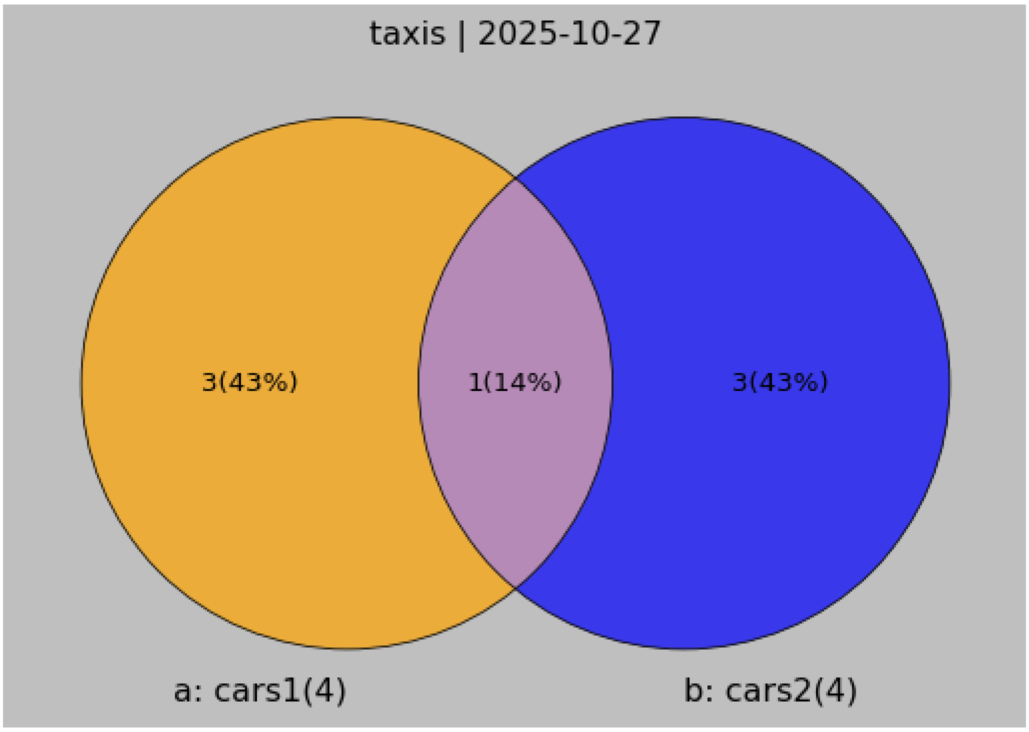
```
_df = df[["distance", "fare", "tip", "tolls"]][:5]
tbl.show_num_df(
    _df,
    data_bar_axis="x",
    pct_axis="xy",
    precision=3,
    total_mode="sum",
)
# _df['distance'].mean()
```

	distance	fare	tip	tolls	Total
0	1.600 (2.0%)	7.000 (8.7%)	2.150 (2.7%)	0	10.750 (13.3%)
1	0.790 (1.0%)	5.000 (6.2%)	0	0	5.790 (7.2%)
2	1.370 (1.7%)	7.500 (9.3%)	2.360 (2.9%)	0	11.230 (13.9%)
3	7.700 (9.5%)	27.000 (33.4%)	6.150 (7.6%)	0	40.850 (50.5%)
4	2.160 (2.7%)	9.000 (11.1%)	1.100 (1.4%)	0	12.260 (15.2%)
Total	13.620 (16.8%)	55.500 (68.6%)	11.760 (14.5%)	0	80.880 (100.0%)

venn

```
set_a = {'ford','ferrari','mercedes', 'bmw'}
set_b = {'opel','bmw','bentley','audi'}
_df, _details = ven.show_venn2(
    title="taxis",
    a_set=set_a,
    a_label="cars1",
    b_set=set_b,
    b_label="cars2",
    verbose=0,
    size=8,
)
```

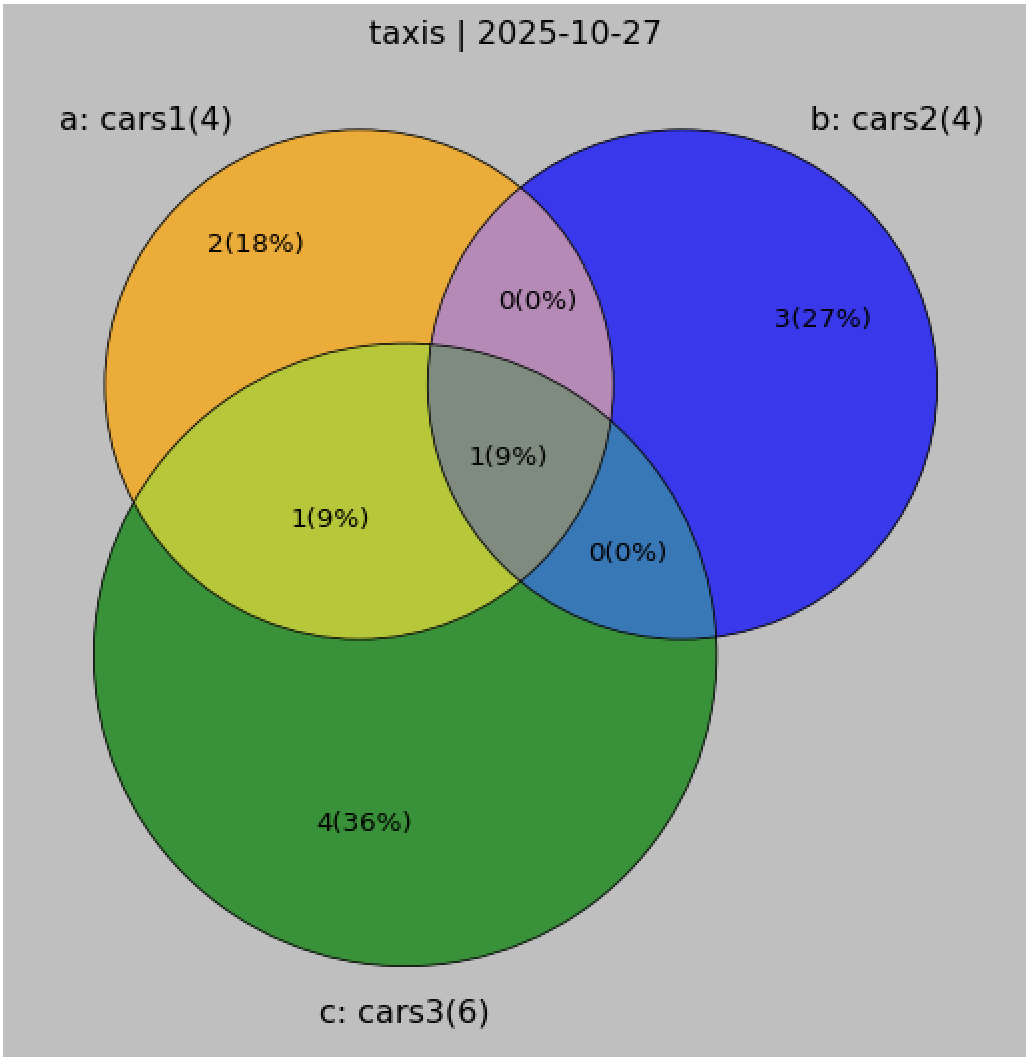
```
ab ⟶ cars1 | cars2 ⟶ len: 7
```




```
set_a = {'ford','ferrari','mercedes', 'bmw'}
set_b = {'opel','bmw','bentley','audi'}
set_c = {'ferrari','bmw','chrysler','renault','peugeot','fiat'}
_df, _details = ven.show_venn3(
    title="taxis",
    a_set=set_a,
    a_label="cars1",
    b_set=set_b,
    b_label="cars2",
    c_set=set_c,
    c_label="cars3",
    verbose=0,
    size=8,
)
```

```
abc ⟶ cars1 | cars2 | cars3 ⟶ len: 11
ab ⟶ cars1 | cars2 ⟶ len: 7
```

```
ac ⟶ cars1 | cars3 ⟶ len: 8
bc ⟶ cars2 | cars3 ⟶ len: 9
```



hlp

```
db_ops_kkr = con.sql("""--sql
    select
            z_kkr_label,
            ops.Code as ops_code,
            count(distinct OPSId) as cnt_ops,
            dim_ops.name as ops_name,
    from Tumor tum
    left join OP op on tum.z_tum_id = op.z_tum_id
    left join OPS ops on op.OPId = ops.OP_TypId
    left join dim_ops on dim_ops.code = ops.Code
    group by ops_code, z_kkr_label, ops_name
    """)
# tbl.descr_db(db_ops_kkr, "ops_kk")

# print(db_ops_kkr.aggregate("ops_code || ' - ' || ops_name as ops, sum(cnt_ops)::int as cnt_ops").order("cnt_ops desc").limit(10).to_df())
db_ops_kkr.aggregate("ops_code || ' - ' || ops_name as ops, sum(cnt_ops)::int as cnt_ops").order("cnt_ops desc").limit(10).show(max_width=120)
db_ops_kkr.aggregate("left(ops_code || ' - ' || ops_name,100) as ops, sum(cnt_ops)::int as cnt_ops").order("cnt_ops desc").limit(10).show()
```

ops varchar	cnt_ops int32
NULL	165429
5-401.11 - Exzision einzelner Lymphknoten und Lymphgefäße: Axillär: Mit Radionuklidmarkierung (Sentinel-…	135826
5-573.40 - Transurethrale Inzision, Exzision, Destruktion und Resektion von (erkranktem) Gewebe der Harn…	94266
5-987.0 - Anwendung eines OP-Roboters: Komplexer OP-Roboter	82761
5-870.a1 - Partielle (brusterhaltende) Exzision der Mamma und Destruktion von Mammagewebe: Partielle Res…	65040
5-870.a2 - Partielle (brusterhaltende) Exzision der Mamma und Destruktion von Mammagewebe: Partielle Res…	52410
5-984 - Mikrochirurgische Technik	47640
5-604.52 - Radikale Prostatovesikulektomie: Laparoskopisch, gefäß- und nervenerhaltend: Mit regionaler L…	39369
5-895.14 - Radikale und ausgedehnte Exzision von erkranktem Gewebe an Haut und Unterhaut: Ohne primären …	35085
5-573.41 - Transurethrale Inzision, Exzision, Destruktion und Resektion von (erkranktem) Gewebe der Harn…	34869
10 rows	2 columns

ops varchar	cnt_ops int32
NULL	165429
5-401.11 - Exzision einzelner Lymphknoten und Lymphgefäße: Axillär: Mit Radionuklidmarkierung (Senti	135826
5-573.40 - Transurethrale Inzision, Exzision, Destruktion und Resektion von (erkranktem) Gewebe der	94266
5-987.0 - Anwendung eines OP-Roboters: Komplexer OP-Roboter	82761
5-870.a1 - Partielle (brusterhaltende) Exzision der Mamma und Destruktion von Mammagewebe: Partielle	65040

5-870.a2 - Partielle (brusterhaltende) Exzision der Mamma und Destruktion von Mammagewebe: Partielle	52410
5-984 - Mikrochirurgische Technik	47640
5-604.52 - Radikale Prostatovesikulektomie: Laparoskopisch, gefäß- und nervenerhaltend: Mit regional	39369
5-895.14 - Radikale und ausgedehnte Exzision von erkranktem Gewebe an Haut und Unterhaut: Ohne primä	35085
5-573.41 - Transurethrale Inzision, Exzision, Destruktion und Resektion von (erkranktem) Gewebe der	34869
10 rows	2 columns

```
df.pickup.to_series()
```

```
0      2019-03-23 20:21:09
1      2019-03-04 16:11:55
2      2019-03-27 17:53:01
3      2019-03-10 01:23:59
4      2019-03-30 13:27:42
               ...
6428   2019-03-31 09:51:53
6429   2019-03-31 17:38:00
6430   2019-03-23 22:55:18
6431   2019-03-04 10:09:25
6432   2019-03-13 19:31:22
Name: pickup, Length: 6433, dtype: datetime64[ns]
```

```
hlp.get_tum_details("df6bc655-e0ca-47f1-900c-ff2c749b3c7d", con)
```

pat

z_pat_id varchar	z_sex varchar	z_age double	z_ag05 varchar	…	Geburtsdatum date	Geburtsdatum_Genau… varchar	DatumVitalStatus date	DatumVitalStatus_G… varchar
450b0462-3c82-411a…	W	83.25	a80b84	…	1939-12-15	T	2023-06-15	T
1 rows								9 columns (8 shown)

tod

TodesursacheId varchar	Code varchar	Version varchar	IsGrundleiden boolean
0 rows			

tum1

z_kkr_label varchar	z_icd10 varchar	Diagnosedatum date	Diagnosedatum_Gena… varchar	…	z_tum_sy_count int16	z_tum_fo_count int16	z_first_treatment varchar	z_first_treatment_… int32
15-ST	C44.2	2023-03-15	T	…	0	5	op	0
1 rows								10 columns (8 shown)

tum2

z_event_order varchar	z_events varchar	Anzahl_Tage_Diagno… int32	…	z_period_diag_psa_… int64	z_last_tum_status varchar	z_class_hpv varchar	z_tum_order int8
fo-op-fo-op-fo	op|fo	NULL	…	NULL	V - Vollremission …	NULL	4
1 rows							9 columns (7 shown)

op

OPId varchar	Intention varchar	Lokale_Beurteilung… varchar	Anzahl_Tage_Diagno… int32	…	Datum_OP_Genauigkeit varchar	z_period_diag_op_day int32	z_op_order int64
b9336bec-c334-412d…	K	R1	0	…	T	0	1
17ed629d-0f14-4dbe…	K	R1	1	…	T	1	2
da08a401-df47-4ae5…	K	R1	29	…	T	29	3
a6b2e505-5d59-460c…	K	R1	50	…	T	50	4
ab1e8069-adf8-46aa…	K	R0	78	…	T	78	5
5 rows							8 columns (7 shown)

ops

OPSId varchar	Code varchar	Version varchar	OP_TypId varchar
1081731c-9595-41ac-9f31-90c1117fe47c	5-181.1	2023	da08a401-df47-4ae5-bfbc-67790d50bdfc
48e2ac46-268f-4cd3-9773-372f176a5afd	5-182.1	2023	a6b2e505-5d59-460c-a271-eafa9e392f18
6910a319-976a-4749-af8a-85f12864c473	5-925.24	2023	ab1e8069-adf8-46aa-adcc-a4ffb8c00b75
cb135ccc-7c90-4658-8a28-48d0e2bd830d	5-181.1	2023	b9336bec-c334-412d-93b5-79b8b23fba90
52be54a2-a0a2-4ba9-a907-1d59506d9f5e	5-182.0	2023	ab1e8069-adf8-46aa-adcc-a4ffb8c00b75
741c8ace-2bdd-4c40-a5b5-e3fac0a8c60b	5-181.4	2023	17ed629d-0f14-4dbe-9d88-610ab6b2a291
93f1b8fe-df4f-4ab6-be4d-b0e217776797	5-903.54	2023	17ed629d-0f14-4dbe-9d88-610ab6b2a291
f58dd5f5-fd6c-481a-9fd7-069e6b0a99b2	5-916.74	2023	b9336bec-c334-412d-93b5-79b8b23fba90

st

STId varchar	Intention varchar	Stellung_OP varchar
0 rows		

be

BestrahlungId	Anzahl_Tage_Diagno…	Anzahl_Tage_ST_Dauer	Datum_Beginn_Bestr…	…	STId	z_period_diag_best…	z_bestr_order
varchar	int32	int32	date		varchar	int32	int64
0 rows							

app

BestrahlungId	TypeOfST_TypBestra…	Seite_Zielgebiet	…	Stereotaktisch	Atemgetriggert	CodeVersion2014	CodeVersion2021
varchar	varchar	varchar		varchar	varchar	varchar	varchar
0 rows							

syst

SYSTId	Intention	Stellung_OP	Therapieart	…	Datum_Beginn_SYST	Datum_Beginn_SYST_…	z_period_diag_syst…	z_syst_order
varchar	varchar	varchar	varchar		date	varchar	int32	int64
0 rows								

fo

FolgeereignisId	Gesamtbeurteilung_…	Verlauf_Lokaler_Tu…	…	Datum_Folgeereigni…	z_fo_order	z_period_diag_fo_day
varchar	varchar	varchar		varchar	int64	int32
1e58d32d-06a5-4a06…	T	T	…	T	1	0
fbd74efe-266c-430c…	T	T	…	T	2	0
13a7ccd4-9133-409d…	T	T	…	T	3	31
04ed157d-787b-419a…	X	NULL	…	T	4	31
927c89f6-d53e-4783…	V	K	…	T	5	92
5 rows						9 columns (6 shown)

fo_tnm

TNMId	FolgeereignisId	Version	y_Symbol	r_Symbol	…	L	V	Pn	S	UICC_Stadium
varchar	varchar	varchar	varchar	varchar		varchar	varchar	varchar	varchar	varchar
fbd74efe-266c-430c…	fbd74efe-266c-430c…	NULL	NULL	NULL	…	NULL	NULL	NULL	NULL	NULL
927c89f6-d53e-4783…	927c89f6-d53e-4783…	NULL	NULL	NULL	…	NULL	NULL	NULL	NULL	NULL
13a7ccd4-9133-409d…	13a7ccd4-9133-409d…	NULL	NULL	NULL	…	NULL	NULL	NULL	NULL	NULL
04ed157d-787b-419a…	04ed157d-787b-419a…	Item8	NULL	r	…	L0	V0	Pn0	NULL	NULL
1e58d32d-06a5-4a06…	1e58d32d-06a5-4a06…	NULL	NULL	NULL	…	NULL	NULL	NULL	NULL	NULL
5 rows										18 columns (10 shown)

fo_fm

FolgeereignisId	FernmetastaseId	Lokalisation
varchar	varchar	varchar
0 rows		

fo_weitere

WeitereKlassifikationId	Name	Stadium	FolgeereignisId
varchar	varchar	varchar	varchar
0 rows			

diag_fm

FernmetastaseId	Lokalisation
varchar	varchar
0 rows	

diag_weitere

WeitereKlassifikationId	Name	Stadium
varchar	varchar	varchar
0 rows		

```
hlp.add_measures_to_pyg_config(".local/pygwalker_spec_.json", strict=False)
```

```
from pandas_plots import hlp
url="https://github.com/robert-koch-institut/Bundesweiter_klinischer_Krebsregisterdatensatz-Datenschema_und_Klassifikationen"
# url="http://google.com"

# hlp.create_barcode_from_url(
#     url=url,
#     output_path=".local/zfkd-repo.png",
#     show_image=True,
# )
```

```
hlp.find_cols(df, [
    "Tot",
    "DIS",
    ])
```

```
['total', 'distance']
```

```
# df
# from pandas_plots import hlp
```

```
hlp.add_datetime_columns(df,"dropoff")
# df
```

```
⌛ Adding datetime columns basing off of: dropoff
```

	pickup	dropoff	passengers	distance	fare	tip	tolls	total	color	payment	...	dropoff_borough	lol	xd	YYYY	MM	Q	YYYY-MM	YYYYQ	YYYY-WW	DDD
0	2019-03-23 20:21:09	2019-03-23 20:27:24	1	1.60	7.0	2.15	0.0	12.95	yellow	credit card	...	Manhattan	NaN	2027	2019	3	1	2019-03	2019Q1	2019-W12	Sat
1	2019-03-04 16:11:55	2019-03-04 16:19:00	1	0.79	5.0	0.00	0.0	9.30	yellow	cash	...	Manhattan	NaN	2027	2019	3	1	2019-03	2019Q1	2019-W10	Mon
2	2019-03-27 17:53:01	2019-03-27 18:00:25	1	1.37	7.5	2.36	0.0	14.16	yellow	credit card	...	Manhattan	NaN	2027	2019	3	1	2019-03	2019Q1	2019-W13	Wed
3	2019-03-10 01:23:59	2019-03-10 01:49:51	1	7.70	27.0	6.15	0.0	36.95	yellow	credit card	...	Manhattan	NaN	2027	2019	3	1	2019-03	2019Q1	2019-W10	Sun
4	2019-03-30 13:27:42	2019-03-30 13:37:14	3	2.16	9.0	1.10	0.0	13.40	yellow	credit card	...	Manhattan	NaN	2027	2019	3	1	2019-03	2019Q1	2019-W13	Sat
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
6428	2019-03-31 09:51:53	2019-03-31 09:55:27	1	0.75	4.5	1.06	0.0	6.36	green	credit card	...	Manhattan	NaN	2027	2019	3	1	2019-03	2019Q1	2019-W13	Sun
6429	2019-03-31 17:38:00	2019-03-31 18:34:23	1	18.74	58.0	0.00	0.0	58.80	green	credit card	...	Bronx	NaN	2027	2019	3	1	2019-03	2019Q1	2019-W13	Sun
6430	2019-03-23 22:55:18	2019-03-23 23:14:25	1	4.14	16.0	0.00	0.0	17.30	green	cash	...	Brooklyn	NaN	2027	2019	3	1	2019-03	2019Q1	2019-W12	Sat
6431	2019-03-04 10:09:25	2019-03-04 10:14:29	1	1.12	6.0	0.00	0.0	6.80	green	credit card	...	Brooklyn	NaN	2027	2019	3	1	2019-03	2019Q1	2019-W10	Mon
6432	2019-03-13 19:31:22	2019-03-13 19:48:02	1	3.85	15.0	3.36	0.0	20.16	green	credit card	...	Brooklyn	NaN	2027	2019	3	1	2019-03	2019Q1	2019-W11	Wed

6433 rows × 23 columns

```
lol="""
The most important themes of the draft referendum are:
1. Improving the use of health data for research and innovation, particularly through the establishment of a National Data Access Point;
2. Enhancing the coordination and networking of different stakeholders involved in health data management;
3. Expanding the scope of the General Data Protection Regulation (GDPR) to cover all areas of healthcare;
4. Providing better access to health data for patients and researchers, including through the development of a new law on health data protection;
5. Facilitating the exchange of health data between different countries and regions, particularly within the European Union (EU);
6. Ensuring that data are used in a way that promotes sustainable development and achieves the United Nations' Sustainable Development Goals (SDGs)."""

print(
hlp.wrap_text(
    text=lol,
    # max_items_in_line=40,
    use_sep=True,
    use_apo=True,
)
)
```

```
[The most important themes of the draft referendum are:
1. Improving the use of health data for research and innovation particularly through
the establishment of a National Data Access Point;
2. Enhancing the coordination and networking of different stakeholders involved
in health data management;
3. Expanding the scope of the General Data Protection Regulation (GDPR) to cover all
areas of healthcare;
4. Providing better access to health data for patients and researchers including through
the development of a new law on health data protection;
5. Facilitating the exchange of health data between different countries and regions
particularly within the European Union (EU);
6. Ensuring that data are used in a way that promotes sustainable development and achieves
the United Nations Sustainable Development Goals (SDGs).]
```