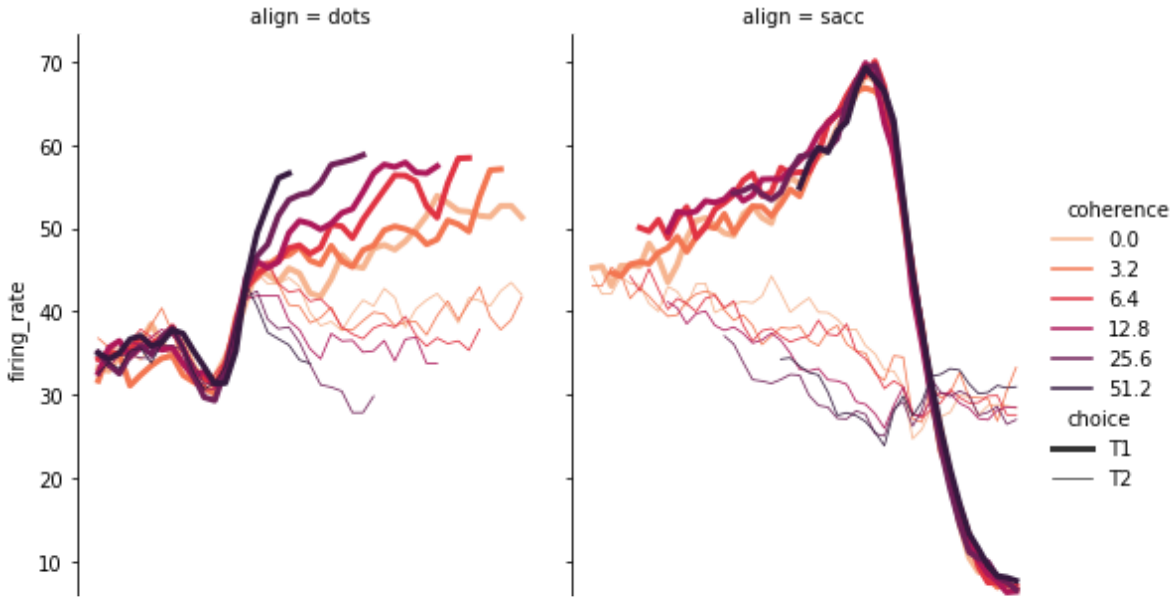
In [4]:
```
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
nuqta=sns.load_dataset("dots")
nuqta.head()
```

Out[4]:

	align	choice	time	coherence	firing_rate
0	dots	T1	-80	0.0	33.189967
1	dots	T1	-80	3.2	31.691726
2	dots	T1	-80	6.4	34.279840
3	dots	T1	-80	12.8	32.631874
4	dots	T1	-80	25.6	35.060487

In [10]:
```
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
nuqta=sns.load_dataset("dots")
#DEFINING A COLOR PALETTE
p=sns.color_palette("rocket_r")
#DRAW A LINEPLOT
sns.relplot(data=nuqta, x="time", y="firing_rate", hue="coherence",
            size="choice", col="align", kind="line", size_order=["T1", "T2"], pale
            height=5, aspect=.75, facet_kws=dict(sharex=False),)
```

Out[10]:
```
<seaborn.axisgrid.FacetGrid at 0x114a783a400>
```

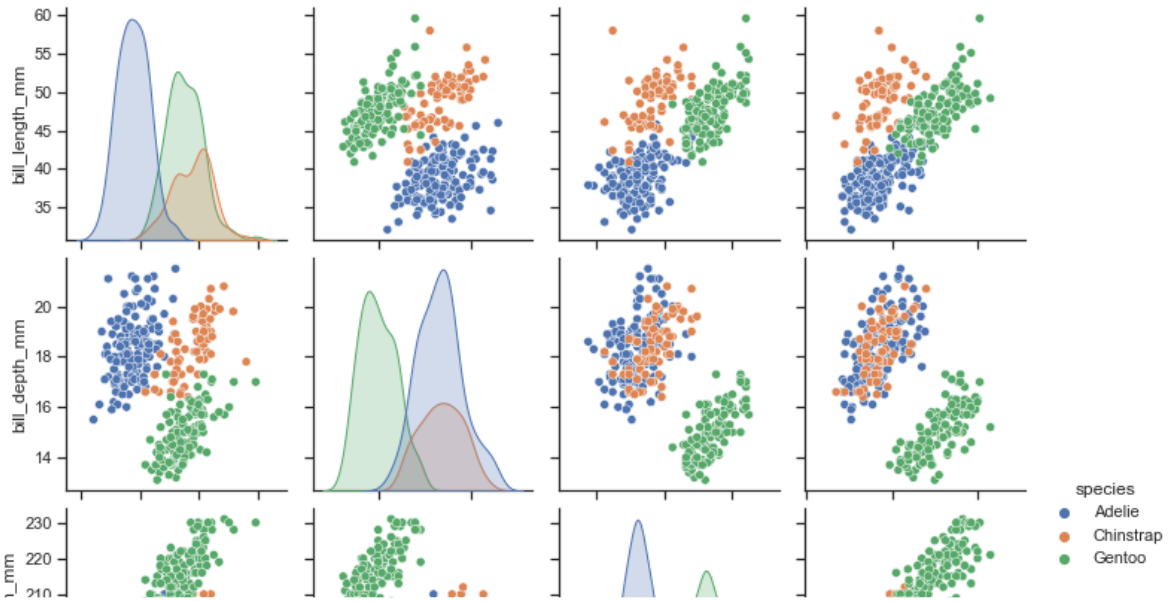


IMPORTING DATA FROM SEABORN LIBRARY ONLINE

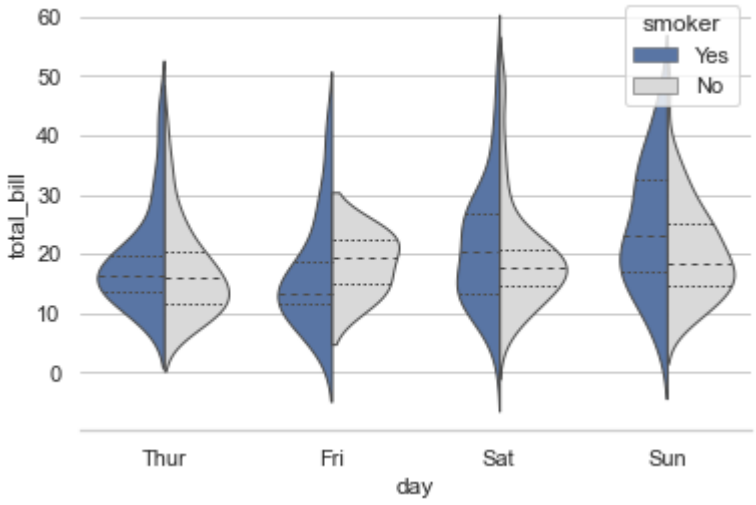
- I CAN ADD VALUE OF MINE OWN DATA TO GET GRAPH LIKE THAT
- BE CAREFUL, ADD NUMERIC VALUE AT PLACE OF NUMERICAL VARIABLE AND VICE VERSA

In [11]:
```
 #IMPORTED DATA FROM SEABORN.PYDATA.ORG
import seaborn as sns
sns.set_theme(style="ticks")

df = sns.load_dataset("penguins")
sns.pairplot(df, hue="species")
```

Out[11]:
```
<seaborn.axisgrid.PairGrid at 0x114a7a2c220>
```



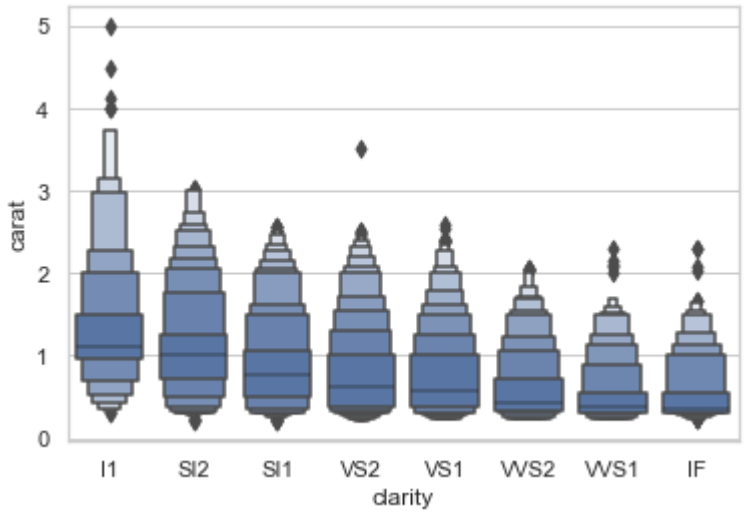
In [13]:
```
import seaborn as sns
sns.set_theme(style="whitegrid")

# Load the example tips dataset
tips = sns.load_dataset("tips")

# Draw a nested violinplot and split the violins for easier comparison
sns.violinplot(data=tips, x="day", y="total_bill", hue="smoker",
               split=True, inner="quart", linewidth=1,
               palette={"Yes": "b", "No": ".85"})
sns.despine(left=True)
```



In [14]:
```
import seaborn as sns
sns.set_theme(style="whitegrid")

# Load the brain networks dataset, select subset, and collapse the multi-index
df = sns.load_dataset("brain_networks", header=[0, 1, 2], index_col=0)

used_networks = [1, 5, 6, 7, 8, 12, 13, 17]
used_columns = (df.columns
                  .get_level_values("network")
                  .astype(int)
                  .isin(used_networks))
df = df.loc[:, used_columns]

df.columns = df.columns.map("-".join)

# Compute a correlation matrix and convert to long-form
corr_mat = df.corr().stack().reset_index(name="correlation")

# Draw each cell as a scatter point with varying size and color
g = sns.relplot(
    data=corr_mat,
    x="level_0", y="level_1", hue="correlation", size="correlation",
    palette="vlag", hue_norm=(-1, 1), edgecolor=".7",
    height=10, sizes=(50, 250), size_norm=(-.2, .8),
)

# Tweak the figure to finalize
g.set(xlabel="", ylabel="", aspect="equal")
g.despine(left=True, bottom=True)
g.ax.margins(.02)
for label in g.ax.get_xticklabels():
    label.set_rotation(90)
for artist in g.legend.legendHandles:
    artist.set_edgecolor(".7")
```

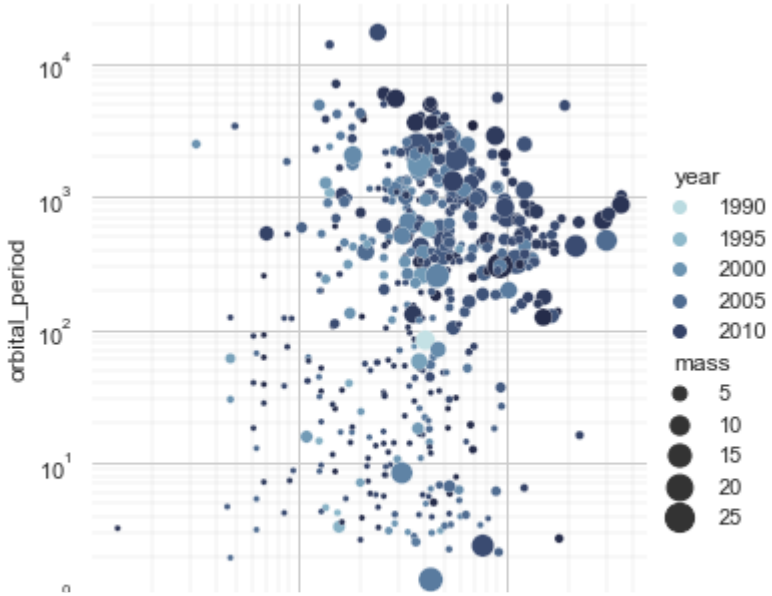
In [16]:
```
import seaborn as sns
sns.set_theme(style="whitegrid")

diamonds = sns.load_dataset("diamonds")
clarity_ranking = ["I1", "SI2", "SI1", "VS2", "VS1", "VVS2", "VVS1", "IF"]

sns.boxenplot(x="clarity", y="carat",
              color="b", order=clarity_ranking,
              scale="linear", data=diamonds)
```

Out[16]: <AxesSubplot:xlabel='clarity', ylabel='carat'>



In [18]:
```
import seaborn as sns
sns.set_theme(style="whitegrid")

# Load the example planets dataset
planets = sns.load_dataset("planets")

cmap = sns.cubehelix_palette(rot=-.2, as_cmap=True)
g = sns.relplot(
    data=planets,
    x="distance", y="orbital_period",
    hue="year", size="mass",
    palette=cmap, sizes=(10, 200),
)
g.set(xscale="log", yscale="log")
g.ax.xaxis.grid(True, "minor", linewidth=.25)
g.ax.yaxis.grid(True, "minor", linewidth=.25)
g.despine(left=True, bottom=True)
```

Out[18]: <seaborn.axisgrid.FacetGrid at 0x114a9d16910>



In [25]:
```
#DARWING GRAPH OF "DOTS" DATA
import seaborn as sns
sns.set_theme(style="whitegrid")

nuqta = sns.load_dataset("dots")
sns.boxenplot(x="choice", y="firing_rate",
              color="b",
              scale="linear", data=nuqta)
```

Out[25]: <AxesSubplot:xlabel='choice', ylabel='firing_rate'>