In [1]:
```
import pandas as pd
import numpy as np
from scipy import stats
import plotly.express as px
import plotly.graph_objects as go
from plotly.subplots import make_subplots
from tqdm.notebook import tqdm
import re
pd.set_option('max_columns', None)
pd.options.display.max_colwidth = 100
```

In [2]:
```
df = pd.read_csv('data.csv', index_col=0)
df.shape
```

Out[2]:
```
(6643221, 4)
```

1 Request time

In [3]:
```
df1 = df[['cid','bytes_returned']].groupby('cid').agg(['count','mean'])
df1.columns = df1.columns.get_level_values(1)
df1 = df1.reset_index()
df1['mean'] = df1['mean'].astype(int)
df1 = df1.rename(columns={"mean": "size"})
df1 = df1.sort_values(by=['size'])
df1.head()
```

Out[3]:

	cid	count	size
135800	QmZiZPaXaT4kSJq6gP3GJ8geNSHxEay8U8EigDhr4x39Gb	1	0
106116	QmXEg9JT6dVPMbmYpY8gWKbeD5fJHdUgcZWTHNLPXM9Vxx	1	0
106115	QmXEfjr121xgyXzU7Uu9U3kFeZh8tmThUvzQXtB94pfuAW	1	0
106105	QmXEdbeckJMpEQbpmsANxK7fPQ8LjYjQJA7ZJFRRPQ24ps	1	0
106103	QmXEc8dmxfTBXrUJaYJJEkiwpGXUDifnKQkrkPwv4cUgkY	1	0

In [4]:
```
df2 = pd.DataFrame(df1['count'].value_counts())
df2 = df2.reset_index()
df2 = df2.rename(columns={"index": "request_time"})
df2 = df2.sort_values(by=['request_time'])
df2.head()
```

Out[4]:

	request_time	count
0	1	163887
1	2	35669
2	3	15839
3	4	8458
4	5	5307

In [5]:
```
df3 = pd.DataFrame(columns = ['request_time', 'count'])

def addRow(df3, l, r):
    df_temp = df1[(df1['count'] >= l) & (df1['count'] < r)]
    c = df_temp.count()[0]
    df3 = df3.append({'request_time':'['+str(l)+','+str(r)+')', 'count':c}, ignore_index = True)
    return df3

df3 = addRow(df3, 1, 2)
df3 = addRow(df3, 2, 10)
df3 = addRow(df3, 10, 100)
df3 = addRow(df3, 100, 1000)
df3 = addRow(df3, 1000, 10000)
df3 = addRow(df3, 10000, 100000)

df3 = df3.replace('[1,2)', '1')
df3 = df3.replace('[10000,100000)', '[10000,+∞)')

total = df3['count'].sum()
df3['percentage'] = df3['count']/total
df3.head()
```

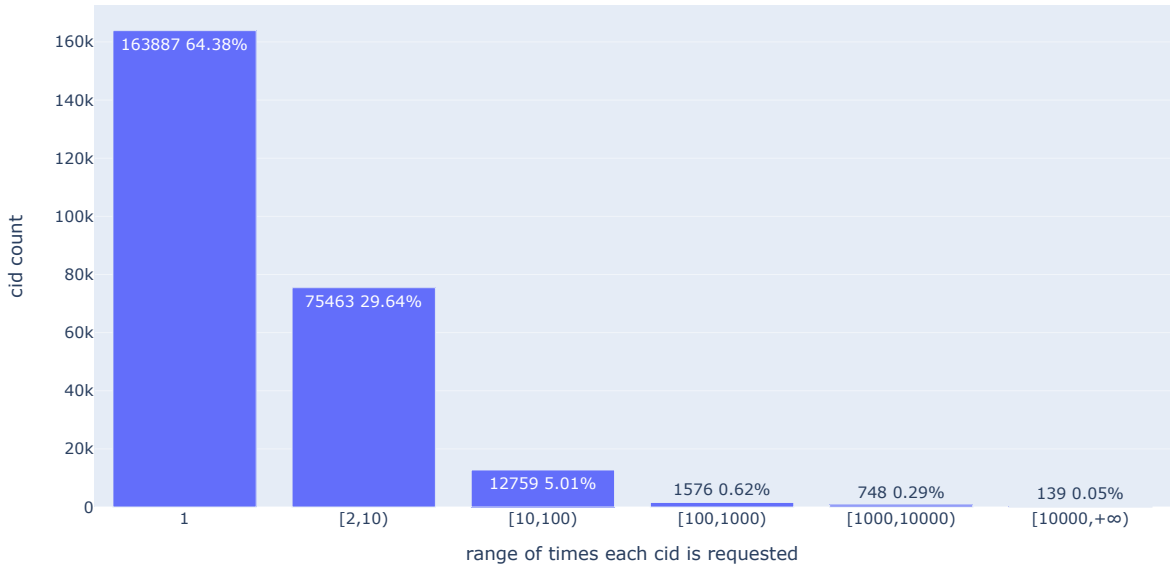
Out[5]:

	request_time	count	percentage
0	1	163887	0.643775
1	[2,10)	75463	0.296431
2	[10,100)	12759	0.050119
3	[100,1000)	1576	0.006191
4	[1000,10000)	748	0.002938

In [6]:
```
fig = px.bar(df3, x='request_time', y='count', text=[str(x[0])+'\n{0:1.2f}%'.format(x[1]*100) for x in zip(df3['count'

fig.update_xaxes(title='range of times each cid is requested')
fig.update_yaxes(title='cid count')

fig.show()
```



In [7]:
```
df5 = df1.copy()

df5['type'] = ''

def addType(l, r):
    df5.loc[(df5['count'] >= l) & (df5['count'] < r), 'type'] = '['+str(l)+','+str(r)+')'

df5.loc[df5['count'] == 1, 'type'] = '1'
addType(2, 10)
addType(10, 100)
addType(100, 1000)
addType(1000, 10000)
df5.loc[df5['count'] > 10000, 'type'] = '[10000,+∞)'

df5.head()
```

Out[7]:

	cid	count	size	type
135800	QmZiZPaXaT4kSJq6gP3GJ8geNSHxEay8U8EigDhr4x39Gb	1	0	1
106116	QmXEg9JT6dVPMbmYpY8gWKbeD5fJHdUgcZWTHNLPXM9Vxx	1	0	1
106115	QmXEfjr121xgyXzU7Uu9U3kFeZh8tmThUvzQXtB94pfuAW	1	0	1
106105	QmXEdbeckJMpEQbpmsANxK7fPQ8LjYjQJA7ZJFRRPQ24ps	1	0	1
106103	QmXEc8dmxfTBXrUJaYJJEkiwpGXUDifnKQkrkPwv4cUgkY	1	0	1

In [8]:
```
def q1(x):
    return x.quantile(0.01)

def q10(x):
    return x.quantile(0.1)

def q90(x):
    return x.quantile(0.9)

def q99(x):
    return x.quantile(0.99)


df6 = df5[['type','size']].groupby('type').agg(['min',q1,q10,'median',q90,q99,'max'])
df6_MB = df6/pow(1024,2)
df6_MB
```

Out[8]:

	size						
	min	q1	q10	median	q90	q99	max
type							
1	0.000000	0.000000	0.000000	0.001406	1.003116	12.203519	2639.354151
[10,100)	0.000000	0.000000	0.000277	0.179795	3.091456	17.519571	69.371615
[100,1000)	0.000000	0.000112	0.000517	0.594907	1.800844	6.266561	41.171799
[1000,10000)	0.000003	0.000156	0.003378	0.801641	1.223510	2.195043	11.552380
[10000,+∞)	0.000088	0.000134	0.370026	0.885081	1.384316	2.066542	2.392586
[2,10)	0.000000	0.000000	0.000079	0.110648	2.210935	14.340477	1590.445610

In [9]:
```
data = df5[['type','size']]
data['size'] = data['size']/1024

fig = px.box(data, x="type", y="size", log_y=True)
fig.update_xaxes(title='range of times each cid is requested')
fig.update_yaxes(title='file size in KB (log scale)')
fig.show()
```

```
/var/folders/gh/hc3npzks3hq9y6jtyp23d8jm0000gn/T/ipykernel_4131/1664295758.py:2: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
```