In [1]:
```
import pandas as pd
```

In [2]:
```
dat = pd.read_csv('priority1calls.csv')
dat.head()
```

Out[2]:

	Incident_Code	Key_Month	Incident_Date	Incident_Priority	Incident_Problem	Incident_Zip_Coc
0	11005617	201301	2013-01-01 00:00:00	1	Unconscious Pri 1	78748
1	11005641	201301	2013-01-01 00:00:00	1	Diabetic Pri 1	78704
2	11005807	201301	2013-01-01 00:00:00	1	Unconscious Pri 1	78753
3	11005817	201301	2013-01-01 00:00:00	1	Unconscious Pri 1	78705
4	11006522	201301	2013-01-01 00:00:00	1	Respiratory Pri 1	78660

5 rows × 35 columns

In [3]:
```
dat.assign(
    call_region = lambda x: (x['Incident_ESD'] == 'City of Austin').replace({
        True: 'City of Austin',
        False: 'Travis County outside Austin'
    })
).call_region.value_counts().apply(
    '{:,.0f}'.format
)
```

Out[3]:
```
City of Austin                35,279
Travis County outside Austin   6,205
Name: call_region, dtype: object
```

In [4]:
```
dat.query(
    'Incident_ESD != "City of Austin"'
)[
    'Response_Interval_Goal_Met (1=Y; " "=N)'
].value_counts(
    normalize=True
).apply('{:.1%}'.format)
```

Out[4]:
```
1    76.8%
0    23.2%
Name: Response_Interval_Goal_Met (1=Y; " "=N), dtype: object
```

Ambulances responding to calls outside Austin from 2013 to 2017 complied with the 12-minute goal for priority one calls 76.9 percent of time.

In [5]:
```
dat.Incident_ESD.value_counts().to_frame().assign(
    calls_per_year = lambda x: x['Incident_ESD'] / len(pd.to_datetime(dat.Incident_Date).apply(lambda y: y.year).dropna().unique())
).join(
    pd.crosstab(
        dat.Incident_ESD,
        dat['Response_Interval_Goal_Met (1=Y; " "=N)']
    ).apply(
        lambda x: 100 * (x / sum(x)), axis=1
    ).round(2).rename(
        columns = {
            0: 'goal_not_met',
            1: 'goal_met'
        }
    )
)
```

Out[5]:

	Incident_ESD	calls_per_year	goal_not_met	goal_met
City of Austin	35279	7055.8	8.46	91.54
ESD02	2172	434.4	8.70	91.30
ESD06	901	180.2	33.85	66.15
ESD12	761	152.2	18.79	81.21
ESD11	701	140.2	39.51	60.49
ESD01	423	84.6	33.81	66.19
ESD04	342	68.4	15.50	84.50
ESD03	234	46.8	32.91	67.09
ESD05	206	41.2	25.73	74.27
ESD09	161	32.2	16.15	83.85
ESD08	112	22.4	34.82	65.18
ESD10	73	14.6	58.90	41.10
ESD13	69	13.8	81.16	18.84
ESD14	50	10.0	70.00	30.00

In [6]:
```
esd_by_year = dat.assign(
    year = pd.to_datetime(dat.Incident_Date).apply(lambda y: y.year)
).groupby([
    'Incident_ESD',
    'year'
])[
    'Response_Interval_Goal_Met (1=Y; " "=N)'
].agg([
    sum,
    pd.Series.count
]).assign(
    pct_met = lambda x: (
        100 * (x['sum'] / x['count'])
    ).round(2)
).rename(
    columns = {
        'sum': 'county_goal_met',
        'count': 'p1_calls'
    }
).unstack()[[
    'p1_calls',
    'county_goal_met',
    'pct_met'
]]

esd_by_year
```

Out[6]:

	p1_calls					county_goal_met					pct_met		
year	2013	2014	2015	2016	2017	2013	2014	2015	2016	2017	2013	2014	2015
Incident_ESD													
City of Austin	6175	6185	7049	7602	8268	5685	5746	6434	6961	7468	92.06	92.90	91.28
ESD01	82	81	74	82	104	54	51	57	49	69	65.85	62.96	77.03
ESD02	446	430	519	500	277	406	391	471	455	260	91.03	90.93	90.75
ESD03	47	43	48	39	57	38	34	30	22	33	80.85	79.07	62.50
ESD04	66	70	53	69	84	55	57	43	59	75	83.33	81.43	81.13
ESD05	44	41	34	47	40	31	27	25	37	33	70.45	65.85	73.53
ESD06	133	137	175	202	254	88	84	116	138	170	66.17	61.31	66.29
ESD08	20	23	15	21	33	15	19	7	12	20	75.00	82.61	46.67
ESD09	31	32	32	37	29	26	29	27	32	21	83.87	90.62	84.38
ESD10	12	16	16	16	13	6	6	7	6	5	50.00	37.50	43.75
ESD11	139	136	125	141	160	77	73	70	84	120	55.40	53.68	56.00
ESD12	133	138	154	182	154	99	121	123	151	124	74.44	87.68	79.87
ESD13	7	13	17	21	11	2	4	1	4	2	28.57	30.77	5.88
ESD14	10	8	7	14	11	5	3	1	4	2	50.00	37.50	14.29

In [ ]: