In [1]:
```
import pandas as pd
import geopandas as gpd
from shapely.geometry import Point

import matplotlib.pyplot as plt
```

In [2]:
```
grid = gpd.read_file('grid_1mile.json').assign(
    idx = lambda x: 'hex-' + x.index.astype(str)
)
```

In [3]:
```
dat_points = pd.read_csv('priority1calls.csv').assign(
    geometry = lambda x: x.apply(
        lambda row: Point(
            row['Incident_Longitude'],
            row['Incident_Latitude']
        ), axis=1
    )
)
```

In [12]:
```
# Added by Steve
dat_points.head()
```

Out[12]:

| | Incident_Code | Key_Month | Incident_Date | Incident_Priority | Incident_Problem | Incident_Zip_Code | Incident_E |
|---|---|---|---|---|---|---|---|
| **0** | 11005617 | 201301 | 2013-01-01 00:00:00 | 1 | Unconscious Pri 1 | 78748.0 | City of Aus |
| **1** | 11005641 | 201301 | 2013-01-01 00:00:00 | 1 | Diabetic Pri 1 | 78704.0 | City of Aus |
| **2** | 11005807 | 201301 | 2013-01-01 00:00:00 | 1 | Unconscious Pri 1 | 78753.0 | City of Aus |
| **3** | 11005817 | 201301 | 2013-01-01 00:00:00 | 1 | Unconscious Pri 1 | 78705.0 | City of Aus |
| **4** | 11006522 | 201301 | 2013-01-01 00:00:00 | 1 | Respiratory Pri 1 | 78660.0 | ESD |

5 rows × 36 columns

In [14]:
```
dat_points = gpd.GeoDataFrame(
    dat_points
)

dat_points.crs = grid.crs
```

In [16]:
```
dat_points_grid = gpd.sjoin(
    dat_points,
    grid,
    how='left',
    op='within'
)
```

In [6]:
```
breakdown_priority = dat_points_grid[[
    'idx',
    'Incident_Priority'
]].pivot_table(
    index='idx',
    columns='Incident_Priority',
    aggfunc=pd.Series.count
).apply(
    lambda x: round(
        (x / sum(x)) * 100,
        2
    ), axis=1
).fillna(0).rename(
    columns = lambda x: 'priority-' + str(x)
)
```

In [7]:
```
breakdown_seconds = dat_points_grid.groupby('idx')['Incident_Priority'].agg([
        pd.np.size
    ]
)
```

In [8]:
```
breakdown_goal = dat_points_grid.groupby('idx')[
    'Response_Interval_Goal_Met (1=Y; " "=N)'
].agg([
    sum,
    pd.Series.count
]).rename(
    columns={
        'sum': 'goal_met',
        'count': 'total_count'
    }
).assign(
    pct_goal_met = lambda x: round(
        (x['goal_met'] / x['total_count']) * 100,
        2
    )
)
```

In [9]:
```
grid_merge = grid.set_index('idx').join(
    breakdown_seconds.join(
        breakdown_priority
    ).join(
        breakdown_goal
    )
).fillna(0).query('size > 0').rename(
    columns = lambda x: str(x)
)
```

In [10]:
```
grid_merge.to_file(
    'points-grid_1mile.geojson',
    driver='GeoJSON'
)
```

In [ ]:
```

```