Mile stone

Team Name: 29DataSets

Jenine Rogel u0468294 u0468294@utah.edu (mailto:u0468294@utah.edu)

Ro Rague u1343169 u1343169@utah.edu (mailto:u1343169@utah.edu)

Repository: https://github.com/Bonampak1/29DataSets (https://github.com/Bonampak1/29DataSets)

Trends of crime data over the years across Utah cities

Background and Motivation (Revised after feedback)

The idea for this project came after some thought about using a combination of the datasets with an Opioid dataset to evaluate if there were any specific trends involved with Opioid use and crime. We would use the location and years and see if there was an overlap. The revised project idea is to see if there are general trends of crime data within specific cities and specific years and crime per day and time and what type of crime per day and build visualizations on that data. We were both intersted in this topic because we were both from Utah but did not know much about the crime in UT.

New Hypothesis: Does a specific day and time of day impact what time of crime is committed? Does the location also impact this result?

Data

We will be using data from https://opendata.utah.gov/ (https://opendata.utah.gov/), because there were multiple dataset CSVs we have to use, they were all downloaded and uploaded into our repository from above. The data is in the form of 29 CSV files.

Data Processing

We do anticipate some data clean up with these files. For example, not all the files have the same columns, in the case of 'Cache_County_Sheriff_Police_Crime_Data' and 'Brigham_City_Police_Crime_Data', the 'Cache_County' CSV file contains an extra column 'country'. On that note, some of the CSV files are per county and others are per city. If we plan to do trends across different counties, we would have to merge the data accordingly on city per county. There is also a 'state' column which we likely not need because the datasets are all for the state of Utah. There are also some 'NAs' on Zip Codes that will have to be accounted for and we will have to align the years across all 29 datasets so that we have coverage of data across those years.

Common columns:

- Incident_id (except SLC)
- Case_number(except SLC)
- incident_datetime (except SLC)
- incident_type_primary (except SLC)
- incident_description (except SLC)
- clearance_type (except SLC)
- address_1 (except SLC)
- city (except SLC)
- state (except SLC)
- Zip (except SLC & Syracuse)
- Latitude (except SLC)
- Longitude (except SLC)
- created_at (except SLC)
- updated_at (except SLC)
- location (except SLC)
- hour_of_day (except SLC)
- day_of_week (except SLC)
- parent_incident_type (except SLC)

Columns to get rid of:

- Clearance Type (did not know what this was and is NA in 3-4 datasets)
- State (it is Utah for all the data)
- Zip code (since we have latitude and longitude and because some files do not have this column)

DATA MERGING

We will need to merge all the county data into one dataframe to use for analytics and visualizations

In [46]:
```
# import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import altair as alt
```

In [47]:
```
# 1.Read in data
st_g1 = pd.read_csv('./1.Saint_George_Police_Data_20231018.csv', dtype={
st_g1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99999 entries, 0 to 99998
Data columns (total 21 columns):
 #   Column                                              Non-Null
Count  Dtype
---  ------                                              --------
------  -----
 0   incident_id                                         99999 no
n-null  int64
 1   case_number                                         99999 no
n-null  object
 2   incident_datetime                                   99999 no
n-null  object
 3   incident_type_primary                               99999 no
n-null  object
 4   incident_description                                99975 no
n-null  object
 5   clearance_type                                      0 non-nu
ll      float64
 6   address_1                                           99978 no
n-null  object
 7   address_2                                           0 non-nu
ll      float64
 8   city                                                99999 no
n-null  object
 9   state                                               99999 no
n-null  object
 10  zip                                                 25454 no
n-null  object
 11  country                                             0 non-nu
ll      float64
 12  latitude                                            99998 no
n-null  float64
 13  longitude                                           99998 no
n-null  float64
 14  created_at                                          99994 no
n-null  object
 15  updated_at                                          99999 no
n-null  object
 16  location                                            99984 no
n-null  object
 17  hour_of_day                                         99999 no
n-null  int64
 18  day_of_week                                         99999 no
n-null  object
 19  parent_incident_type                                99999 no
n-null  object
 20  St George Police Department Districts Shapes - qdt2-uyjz  79156 no
n-null  float64
dtypes: float64(6), int64(2), object(13)
memory usage: 16.0+ MB
```

In [48]:
```
# Read in data
st_g2 = pd.read_csv('./2.Saint_George_Police_Data_20231018.csv')
st_g2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 90243 entries, 0 to 90242
Data columns (total 21 columns):
 #   Column                                                  Non-Null
Count  Dtype
---  ------                                                  --------
------  -----
 0   incident_id                                             90243 no
n-null  int64
 1   case_number                                             90243 no
n-null  object
 2   incident_datetime                                       90243 no
n-null  object
 3   incident_type_primary                                   90243 no
n-null  object
 4   incident_description                                    90144 no
n-null  object
 5   clearance_type                                          0 non-nu
ll      float64
 6   address_1                                               90228 no
n-null  object
 7   address_2                                               0 non-nu
ll      float64
 8   city                                                    90243 no
n-null  object
 9   state                                                   90243 no
n-null  object
 10  zip                                                     32649 no
n-null  object
 11  country                                                 0 non-nu
ll      float64
 12  latitude                                                90243 no
n-null  float64
 13  longitude                                               90243 no
n-null  float64
 14  created_at                                              90243 no
n-null  object
 15  updated_at                                              90243 no
n-null  object
 16  location                                                90240 no
n-null  object
 17  hour_of_day                                             90243 no
n-null  int64
 18  day_of_week                                             90243 no
n-null  object
 19  parent_incident_type                                    90243 no
n-null  object
 20  St George Police Department Districts Shapes - qdt2-uyjz  59944 no
n-null  float64
dtypes: float64(6), int64(2), object(13)
memory usage: 14.5+ MB
```

```
/var/folders/q4/rr159kcn4vb9yhnfzgzd85xh0000gn/T/ipykernel_3964/1193372
075.py:2: DtypeWarning: Columns (10) have mixed types. Specify dtype op
tion on import or set low_memory=False.
  st_g2 = pd.read_csv('./2.Saint_George_Police_Data_20231018.csv')
```

In [49]:

```
# Concatenate the 2 Saint George dataframes vertically
st_george = pd.concat([st_g1, st_g2], ignore_index=True)
#st_george
```

In [50]:

```
# Re-arrange columns
st_george = st_george[['incident_id', 'case_number', 'incident_datetime'
               'incident_description','address_1', 'city', 'latitude', ''
               'location', 'hour_of_day', 'day_of_week', 'created_at', ''
st_george.head()
```

Out[50]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incider
0	691160812	15P003395	2/11/2015 16:02	Theft	FRAUD	
1	712757140	15P008661	4/13/2015 13:04	Drugs	DRUGS	
2	715044076	15P010661	5/5/2015 22:05	Drugs	DRUGS	
3	825529671	17P027337	11/5/2017 5:11	Traffic	DUI	
4	736930480	15P027109	11/12/2015 23:11	Traffic	DUI	

In [51]:
```
# 2.Read in data
beaver = pd.read_csv('./Beaver_County_Police_Crime_Data_20231011.csv')
beaver.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2676 entries, 0 to 2675
Data columns (total 20 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Unnamed Column          2676 non-null   int64  
 1   address_1               2653 non-null   object 
 2   case_number             2676 non-null   int64  
 3   city                    2676 non-null   object 
 4   clearance_type          0 non-null      float64
 5   country                 1 non-null      object 
 6   created_at              2676 non-null   object 
 7   day_of_week             2676 non-null   object 
 8   hour_of_day             2676 non-null   int64  
 9   incident_datetime       2676 non-null   object 
 10  incident_description    2676 non-null   object 
 11  incident_id             2676 non-null   int64  
 12  incident_type_primary   2676 non-null   object 
 13  latitude                2676 non-null   float64
 14  location                2676 non-null   object 
 15  longitude               2676 non-null   float64
 16  parent_incident_type    2676 non-null   object 
 17  state                   2676 non-null   object 
 18  updated_at              2676 non-null   object 
 19  zip                     2514 non-null   float64
dtypes: float64(4), int64(4), object(12)
memory usage: 418.2+ KB
```

In [52]:
```
# Re-arrange columns
beaver = beaver[['incident_id', 'case_number', 'incident_datetime','parer
                'incident_description','address_1', 'city', 'latitude', '
                'location', 'hour_of_day', 'day_of_week', 'created_at', 'u
beaver.head()
```

Out[52]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incider
0	45510703	36871	8/31/2010 0:00	Theft	0	A
1	834901401	55101	2/10/2018 13:22	Traffic	PI Accident	
2	834901390	55112	2/12/2018 14:08	Traffic	Traffic Control	
3	834901372	55130	2/15/2018 10:54	Traffic	Livestock Probl	L
4	834901370	55132	2/15/2018 23:54	Community Policing	Suspicious	

In [53]:
```
# Look at the unique values of "clearance_type" to decide if we want the
# unique_values_clearance_type = pd.unique(beaver['clearance_type'])
# unique_values_clearance_type
```

In [54]:
```
# Look at the unique values of "parent_incident_type" to decide if we wai
unique_values_parent_incident_type = pd.unique(beaver['parent_incident_ty
unique_values_parent_incident_type
```

Out[54]:
```
array(['Theft', 'Traffic', 'Community Policing', 'Other',
       'Weapons Offense', 'Other Sexual Offense', 'Disorder', 'Drugs',
       'Alarm', 'Property Crime', 'Fire', 'Emergency', 'Family Offens
e',
       'Liquor', 'Breaking & Entering', 'Missing Person', 'Assault',
       'Theft of Vehicle', 'Robbery', 'Vehicle Recovery'], dtype=objec
t)
```

In [55]:
```
# 3.Read in data
Brigham = pd.read_csv('./Brigham_City_Police_Crime_Data_20231018.csv')
Brigham.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15000 entries, 0 to 14999
Data columns (total 19 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   column 1                15000 non-null  int64  
 1   address_1               15000 non-null  object 
 2   case_number             15000 non-null  object 
 3   city                    14999 non-null  object 
 4   clearance_type          0 non-null      float64
 5   created_at              15000 non-null  object 
 6   day_of_week             15000 non-null  object 
 7   hour_of_day             15000 non-null  int64  
 8   incident_datetime       15000 non-null  object 
 9   incident_description    15000 non-null  object 
 10  incident_id             15000 non-null  int64  
 11  incident_type_primary   15000 non-null  object 
 12  latitude                15000 non-null  float64
 13  location                15000 non-null  object 
 14  longitude               15000 non-null  float64
 15  parent_incident_type    15000 non-null  object 
 16  state                   15000 non-null  object 
 17  updated_at              15000 non-null  object 
 18  zip                     14957 non-null  float64
dtypes: float64(4), int64(3), object(12)
memory usage: 2.2+ MB
```

In [56]:
```
# 3.Re-arrange columns
Brigham=Brigham[['incident_id', 'case_number', 'incident_datetime','pare
               'incident_description','address_1', 'city', 'latitude', '
               'location', 'hour_of_day', 'day_of_week', 'created_at', '
Brigham.head()
```

Out[56]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incider
0	913039167	19-B02493	04/05/2019 11:41:06 AM	Weapons Offense	Weapon Offense	W
1	832915287	17-B09872	12/22/2017 08:33:10 AM	Other	Suspicious	
2	832915358	17-B09944	12/22/2017 11:00:00 AM	Traffic	PD Accident	
3	832915293	17-B09878	12/22/2017 12:16:37 PM	Other	Citizen Assist	
4	832915296	17-B09881	12/22/2017 02:53:57 PM	Other	911 Unknown	

In [57]:
```
# 4.Read in data
cache = pd.read_csv('./Cache_County_Sheriff_Police_Crime_Data_20231018.cs
cache.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15000 entries, 0 to 14999
Data columns (total 20 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   column 1                15000 non-null  int64  
 1   address_1               15000 non-null  object 
 2   case_number             15000 non-null  object 
 3   city                    15000 non-null  object 
 4   clearance_type          0 non-null      float64
 5   country                 0 non-null      float64
 6   created_at              15000 non-null  object 
 7   day_of_week             15000 non-null  object 
 8   hour_of_day             15000 non-null  int64  
 9   incident_datetime       15000 non-null  object 
 10  incident_description    15000 non-null  object 
 11  incident_id             15000 non-null  int64  
 12  incident_type_primary   15000 non-null  object 
 13  latitude                15000 non-null  float64
 14  location                15000 non-null  object 
 15  longitude               15000 non-null  float64
 16  parent_incident_type    15000 non-null  object 
 17  state                   15000 non-null  object 
 18  updated_at              15000 non-null  object 
 19  zip                     0 non-null      float64
dtypes: float64(5), int64(3), object(12)
memory usage: 2.3+ MB
```

In [58]:
```
# 4.Re-arrange columns
cache = cache[['incident_id', 'case_number', 'incident_datetime','parent_
                'incident_description','address_1', 'city', 'latitude', '
                'location', 'hour_of_day', 'day_of_week', 'created_at', '
cache.head()
```

Out[58]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incider
0	758696378	16-C3608	05/04/2016 06:31:41 PM	Drugs	[CCSO] C/S DRUGS	
1	756956187	16-C3027	04/16/2016 09:27:09 PM	Community Policing	[CCSO] SUSP INCIDENT	
2	757683093	16-C3198	04/22/2016 05:31:43 AM	Alarm	[CCSO] ALARM,INTRU	
3	757683094	16-C3199	04/22/2016 07:38:07 AM	Community Policing	[CCSO] ANIMAL PROBLEM	A
4	750003534	16-C1221	02/15/2016 01:03:09 PM	Community Policing	[CCSO] ANIMAL PROBLEM	A

In [59]:
```
# 5.Read in data
ephraim = pd.read_csv('./Ephraim_City_Police_Crime_Data_20231018.csv')
ephraim.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4061 entries, 0 to 4060
Data columns (total 20 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Unnamed Column         4061 non-null   int64
 1   address_1              4035 non-null   object
 2   case_number            4061 non-null   object
 3   city                   4061 non-null   object
 4   clearance_type         0 non-null      float64
 5   country                0 non-null      float64
 6   created_at             4061 non-null   object
 7   day_of_week            4061 non-null   object
 8   hour_of_day            4061 non-null   int64
 9   incident_datetime      4061 non-null   object
 10  incident_description   4061 non-null   object
 11  incident_id            4061 non-null   int64
 12  incident_type_primary  4061 non-null   object
 13  latitude               4035 non-null   float64
 14  location               4035 non-null   object
 15  longitude              4035 non-null   float64
 16  parent_incident_type   4061 non-null   object
 17  state                  4061 non-null   object
 18  updated_at             4061 non-null   object
 19  zip                    3987 non-null   float64
dtypes: float64(5), int64(3), object(12)
memory usage: 634.7+ KB
```

In [60]:
```
# 5.Re-arrange columns
ephraim = ephraim[['incident_id', 'case_number', 'incident_datetime','pa
               'incident_description','address_1', 'city', 'latitude', ''
               'location', 'hour_of_day', 'day_of_week', 'created_at', 'u
ephraim.head()
```

Out[60]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incider
0	758597485	15110222	11/06/2015 12:20:00 PM	Property Crime	TRESPASSING	TF T PI
1	758597159	16010069	01/03/2016 05:56:00 PM	Community Policing	SUSPICIOUS ACTIVITY	
2	758597083	16010500	01/15/2016 03:23:00 AM	Community Policing	SUSPICIOUS ACTIVITY	A(
3	758597081	16010515	01/15/2016 09:43:00 AM	Other	MISCELLANEOUS	MISC I
4	490848716	14070315	07/10/2014 06:54:00 PM	Community Policing	SUSPICIOUS ACTIVITY	AC

In [61]:
```
# 6.Read in data
iron = pd.read_csv('./Iron_County_Sheriffs_Office_Crime_Police_Data_2023
iron.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8513 entries, 0 to 8512
Data columns (total 20 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   Unnamed Column         8513 non-null   int64  
 1   address_1              8504 non-null   object 
 2   case_number            8513 non-null   object 
 3   city                   8495 non-null   object 
 4   clearance_type         0 non-null      float64
 5   country                0 non-null      float64
 6   created_at             8513 non-null   object 
 7   day_of_week            8513 non-null   object 
 8   hour_of_day            8513 non-null   int64  
 9   incident_datetime      8513 non-null   object 
 10  incident_description   8512 non-null   object 
 11  incident_id            8513 non-null   int64  
 12  incident_type_primary  8513 non-null   object 
 13  latitude               8509 non-null   float64
 14  location               8509 non-null   object 
 15  longitude              8509 non-null   float64
 16  parent_incident_type   8513 non-null   object 
 17  state                  8487 non-null   object 
 18  updated_at             8513 non-null   object 
 19  zip                    1794 non-null   float64
dtypes: float64(5), int64(3), object(12)
memory usage: 1.3+ MB
```

In [62]:
```
# 6.Re-arrange columns
iron = iron[['incident_id', 'case_number', 'incident_datetime','parent_i
               'incident_description','address_1', 'city', 'latitude', '
               'location', 'hour_of_day', 'day_of_week', 'created_at', '
iron.head()
```

Out[62]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incider
0	409580467	13-03269	12/09/2013 08:00:50 PM	Drugs	Drugs	
1	515955081	14-02014	08/05/2014 10:02:01 AM	Alarm	Alarm	
2	548497904	14-02349	09/08/2014 01:43:34 PM	Drugs	Drugs	
3	571312495	14-02613	10/03/2014 08:16:05 PM	Theft	Theft	
4	575721633	14-02657	10/08/2014 05:48:31 PM	Traffic	DUI	

In [63]:
```
# 7.Read in data
juab = pd.read_csv('./Juab_County_Sheriff_Police_Crime_Data_20231018.csv
juab.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1640 entries, 0 to 1639
Data columns (total 20 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Unnamed Column          1640 non-null   int64  
 1   address_1               1640 non-null   object 
 2   case_number             1640 non-null   object 
 3   city                    1640 non-null   object 
 4   clearance_type          0 non-null      float64
 5   country                 0 non-null      float64
 6   created_at              1640 non-null   object 
 7   day_of_week             1640 non-null   object 
 8   hour_of_day             1640 non-null   int64  
 9   incident_datetime       1640 non-null   object 
 10  incident_description    1640 non-null   object 
 11  incident_id             1640 non-null   int64  
 12  incident_type_primary   1640 non-null   object 
 13  latitude                1640 non-null   float64
 14  location                1640 non-null   object 
 15  longitude               1640 non-null   float64
 16  parent_incident_type    1640 non-null   object 
 17  state                   1640 non-null   object 
 18  updated_at              1640 non-null   object 
 19  zip                     0 non-null      float64
dtypes: float64(5), int64(3), object(12)
memory usage: 256.4+ KB
```

In [64]:
```
# 7.Re-arrange columns
juab = juab[['incident_id', 'case_number', 'incident_datetime','parent_i
                'incident_description','address_1', 'city', 'latitude', '
                'location', 'hour_of_day', 'day_of_week', 'created_at', '
juab.head()
```

Out[64]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incider
0	751460206	16JC0307	03/01/2016 11:03:02 AM	Community Policing	ANIMAL PROBLEM	Desc
1	842131619	18JC0358	03/27/2018 09:03:14 AM	Other	NON-CRIMINAL CIVIL COMPLAINT	Des
2	751760013	16JC0309	03/02/2016 08:03:13 AM	Community Policing	ANIMAL PROBLEM	Desc
3	752212713	16JC0314	03/03/2016 10:03:46 AM	Community Policing	CITIZEN ASSIST	Desc
4	752212707	16JC0321	03/04/2016 05:03:57 PM	Community Policing	CITIZEN ASSIST	Desc

In [65]:
```
# 8.Read in data
kaysville = pd.read_csv('./Kaysville_City_Police_Crime_Data_20231018.csv
kaysville.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15000 entries, 0 to 14999
Data columns (total 19 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   column 1                15000 non-null  int64  
 1   address_1               15000 non-null  object 
 2   case_number             15000 non-null  object 
 3   city                    15000 non-null  object 
 4   clearance_type          0 non-null      float64
 5   created_at              15000 non-null  object 
 6   day_of_week             15000 non-null  object 
 7   hour_of_day             15000 non-null  int64  
 8   incident_datetime       15000 non-null  object 
 9   incident_description    15000 non-null  object 
 10  incident_id             15000 non-null  int64  
 11  incident_type_primary   15000 non-null  object 
 12  latitude                15000 non-null  float64
 13  location                15000 non-null  object 
 14  longitude               15000 non-null  float64
 15  parent_incident_type    15000 non-null  object 
 16  state                   15000 non-null  object 
 17  updated_at              15000 non-null  object 
 18  zip                     13271 non-null  float64
dtypes: float64(4), int64(3), object(12)
memory usage: 2.2+ MB
```

In [66]:
```
# 8.Re-arrange columns
kaysville = kaysville[['incident_id', 'case_number', 'incident_datetime'
                'incident_description','address_1', 'city', 'latitude', '
                'location', 'hour_of_day', 'day_of_week', 'created_at', '
kaysville.head()
```

Out[66]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incider
0	833928479	C1858059	02/11/2018 07:23:38 AM	Vehicle Stop	CAD: Traffic Stop	
1	833928478	C1858068	02/11/2018 08:57:37 AM	Vehicle Stop	CAD: Traffic Stop	
2	833928477	C1858071	02/11/2018 09:06:37 AM	Vehicle Stop	CAD: Traffic Stop	
3	833928476	C1858073	02/11/2018 09:15:27 AM	Vehicle Stop	CAD: Traffic Stop	
4	833928475	C1858076	02/11/2018 09:21:26 AM	Vehicle Stop	CAD: Traffic Stop	

```
In [67]: # 9.Read in data
park_city = pd.read_csv('./Park_City_Police_Crime_Data_20231018.csv')
park_city.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15000 entries, 0 to 14999
Data columns (total 20 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Unnamed Column          15000 non-null  int64  
 1   address_1               14995 non-null  object 
 2   case_number             15000 non-null  object 
 3   city                    14969 non-null  object 
 4   clearance_type          0 non-null      float64
 5   country                 0 non-null      float64
 6   created_at              15000 non-null  object 
 7   day_of_week             15000 non-null  object 
 8   hour_of_day             15000 non-null  int64  
 9   incident_datetime       15000 non-null  object 
 10  incident_description    14996 non-null  object 
 11  incident_id             15000 non-null  int64  
 12  incident_type_primary   15000 non-null  object 
 13  latitude                15000 non-null  float64
 14  location                15000 non-null  object 
 15  longitude               15000 non-null  float64
 16  parent_incident_type    15000 non-null  object 
 17  state                   15000 non-null  object 
 18  updated_at              15000 non-null  object 
 19  zip                     393 non-null    float64
dtypes: float64(5), int64(3), object(12)
memory usage: 2.3+ MB
```

In [68]:
```
# 9.Re-arrange columns
park_city = park_city[['incident_id', 'case_number', 'incident_datetime'
                'incident_description','address_1', 'city', 'latitude', ''
                'location', 'hour_of_day', 'day_of_week', 'created_at', ''
park_city.head()
```

Out[68]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incider
0	828265191	17-20940	12/04/2017 01:12:09 PM	Theft	FRAUD	
1	95483627	A10-24288	10/26/2010 09:10:01 AM	Property Crime	CRIM MISCHIEF	
2	734697515	15-19052	10/21/2015 12:10:55 AM	Disorder	SUSPICIOUS	
3	760351917	16-09075	05/03/2016 12:05:00 AM	Property Crime	CRIM MISCHIEF	
4	760351928	16-09086	05/12/2016 10:05:41 AM	Disorder	HARASSMENT	

In [69]:
```
# 10.Read in data
perry = pd.read_csv('./Perry_City_Police_Crime_Data_20231018.csv')
perry.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 679 entries, 0 to 678
Data columns (total 19 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   column 1               679 non-null    int64  
 1   address_1              679 non-null    object 
 2   case_number            679 non-null    object 
 3   city                   679 non-null    object 
 4   clearance_type         0 non-null      float64
 5   created_at             679 non-null    object 
 6   day_of_week            679 non-null    object 
 7   hour_of_day            679 non-null    int64  
 8   incident_datetime      679 non-null    object 
 9   incident_description   679 non-null    object 
 10  incident_id            679 non-null    int64  
 11  incident_type_primary  679 non-null    object 
 12  latitude               679 non-null    float64
 13  location               679 non-null    object 
 14  longitude              679 non-null    float64
 15  parent_incident_type   679 non-null    object 
 16  state                  679 non-null    object 
 17  updated_at             679 non-null    object 
 18  zip                    661 non-null    float64
dtypes: float64(4), int64(3), object(12)
memory usage: 100.9+ KB
```

In [70]:
```
# 10.Re-arrange columns
perry = perry[['incident_id', 'case_number', 'incident_datetime','parent_
               'incident_description','address_1', 'city', 'latitude', ''
               'location', 'hour_of_day', 'day_of_week', 'created_at', 'u
perry.head()
```

Out[70]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incider
0	742623389	15-P01549	12/30/2015 12:48:02 PM	Property Crime	Vandalism	
1	742754753	16-P00005	01/01/2016 05:10:47 PM	Traffic	Hit & Run PD	
2	742754754	16-P00006	01/02/2016 04:44:18 AM	Property Crime	Property Damage	Prc
3	742923954	16-P00013	01/04/2016 04:13:13 AM	Property Crime	Property Damage	Prc
4	746843931	16-P00074	01/19/2016 09:11:04 AM	Theft	fraud	

In [71]:
```
# 11.Read in data: Pleasant_View_Police_Crime_Data_20231018
pleasant_view = pd.read_csv('./Pleasant_View_Police_Crime_Data_20231018.
pleasant_view.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15000 entries, 0 to 14999
Data columns (total 19 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   column 1                15000 non-null  int64  
 1   address_1               15000 non-null  object 
 2   case_number             15000 non-null  object 
 3   city                    14999 non-null  object 
 4   clearance_type          0 non-null      float64
 5   created_at              15000 non-null  object 
 6   day_of_week             15000 non-null  object 
 7   hour_of_day             15000 non-null  int64  
 8   incident_datetime       15000 non-null  object 
 9   incident_description    15000 non-null  object 
 10  incident_id             15000 non-null  int64  
 11  incident_type_primary   15000 non-null  object 
 12  latitude                15000 non-null  float64
 13  location                15000 non-null  object 
 14  longitude               15000 non-null  float64
 15  parent_incident_type    15000 non-null  object 
 16  state                   15000 non-null  object 
 17  updated_at              15000 non-null  object 
 18  zip                     14957 non-null  float64
dtypes: float64(4), int64(3), object(12)
memory usage: 2.2+ MB
```

In [72]:
```
# 11.Re-arrange columns
pleasant_view = pleasant_view[['incident_id', 'case_number', 'incident_da
               'incident_description','address_1', 'city', 'latitude', ''
               'location', 'hour_of_day', 'day_of_week', 'created_at', 'u
pleasant_view.head()
```

Out[72]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incider
0	913039167	19-B02493	04/05/2019 11:41:06 AM	Weapons Offense	Weapon Offense	W
1	832915287	17-B09872	12/22/2017 08:33:10 AM	Other	Suspicious	
2	832915358	17-B09944	12/22/2017 11:00:00 AM	Traffic	PD Accident	
3	832915293	17-B09878	12/22/2017 12:16:37 PM	Other	Citizen Assist	
4	832915296	17-B09881	12/22/2017 02:53:57 PM	Other	911 Unknown	

In [73]:
```
# 12.Read in data: Price_Police_Crime_Data_20231018
price = pd.read_csv('./Price_Police_Crime_Data_20231018.csv')
price.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8618 entries, 0 to 8617
Data columns (total 20 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Unnamed Column          8618 non-null   int64  
 1   address_1               8617 non-null   object 
 2   case_number             8618 non-null   int64  
 3   city                    8618 non-null   object 
 4   clearance_type          0 non-null      float64
 5   country                 0 non-null      float64
 6   created_at              8582 non-null   object 
 7   day_of_week             8618 non-null   object 
 8   hour_of_day             8618 non-null   int64  
 9   incident_datetime       8618 non-null   object 
 10  incident_description    8618 non-null   object 
 11  incident_id             8618 non-null   int64  
 12  incident_type_primary   8618 non-null   object 
 13  latitude                8617 non-null   float64
 14  location                8617 non-null   object 
 15  longitude               8617 non-null   float64
 16  parent_incident_type    8618 non-null   object 
 17  state                   8618 non-null   object 
 18  updated_at              8618 non-null   object 
 19  zip                     8432 non-null   object 
dtypes: float64(4), int64(4), object(12)
memory usage: 1.3+ MB
```

In [74]:
```
# 12.Re-arrange columns
price = price[['incident_id', 'case_number', 'incident_datetime','parent_
               'incident_description','address_1', 'city', 'latitude', '
               'location', 'hour_of_day', 'day_of_week', 'created_at', 'u
price.head()
```

Out[74]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incider
0	100181896	20120235	02/22/2012 06:00:00 PM	Disorder	[PCPD] Public Intoxication	PU Pub
1	100329494	20120233	02/22/2012 02:00:00 PM	Traffic	"[PCPD] Traffic Accident, Vehicle Damage"	ACC Accic
2	100329495	20120237	02/23/2012 12:45:00 PM	Theft	[PCPD] Theft Other	THEF
3	100329496	20120240	02/23/2012 04:08:00 PM	Drugs	[PCPD] Other Controlled Substances	(SUBS Contr
4	100329497	20120239	02/23/2012 04:08:00 PM	Traffic	"[PCPD] Traffic Accident, Vehicle Damage"	ACC Accic

In [75]:
```
# 13.Read in data: Roy_City_Police_Crime_Data_20231018
roy = pd.read_csv('./Roy_City_Police_Crime_Data_20231018.csv')
roy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15000 entries, 0 to 14999
Data columns (total 20 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   column 1                15000 non-null  int64  
 1   address_1               14983 non-null  object 
 2   case_number             15000 non-null  object 
 3   city                    15000 non-null  object 
 4   clearance_type          0 non-null      float64
 5   country                 0 non-null      float64
 6   created_at              15000 non-null  object 
 7   day_of_week             15000 non-null  object 
 8   hour_of_day             15000 non-null  int64  
 9   incident_datetime       15000 non-null  object 
 10  incident_description    15000 non-null  object 
 11  incident_id             15000 non-null  int64  
 12  incident_type_primary   15000 non-null  object 
 13  latitude                14983 non-null  float64
 14  location                14983 non-null  object 
 15  longitude               14983 non-null  float64
 16  parent_incident_type    15000 non-null  object 
 17  state                   15000 non-null  object 
 18  updated_at              15000 non-null  object 
 19  zip                     14826 non-null  float64
dtypes: float64(5), int64(3), object(12)
memory usage: 2.3+ MB
```

In [76]:
```
# 13.Re-arrange columns
roy = roy[['incident_id', 'case_number', 'incident_datetime','parent_inc
               'incident_description','address_1', 'city', 'latitude', '
               'location', 'hour_of_day', 'day_of_week', 'created_at', '
roy.head()
```

Out[76]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incider
0	751766860	16-5647	03/04/2016 11:13:53 AM	Other	Alarm	ALAI
1	751029830	16-5048	02/27/2016 01:54:08 PM	Traffic	Traffic	TRAF
2	751029833	16-5053	02/27/2016 02:16:06 PM	Other	Other	KEE
3	751029835	16-5061	02/27/2016 04:44:29 PM	Traffic	Traffic	TRAF
4	751029838	16-5066	02/27/2016 05:31:02 PM	Traffic	Traffic	TRAF

In [77]:
```
# 14.Read in data: Salt_Lake_County_Crime_Data_2013_20231018
slc = pd.read_csv('./Salt_Lake_County_Crime_Data_2013_20231018.csv')
slc.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14 entries, 0 to 13
Data columns (total 12 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   Agency               14 non-null     object 
 1   Population           14 non-null     int64  
 2   Homicide             13 non-null     float64
 3   Rape                 13 non-null     float64
 4   Robbery              13 non-null     float64
 5   Aggravated Assault   13 non-null     float64
 6   Burglary             13 non-null     float64
 7   Larceny              13 non-null     float64
 8   Motor Vehicle Theft  13 non-null     float64
 9   Arson                13 non-null     float64
 10  Total Crime Index    13 non-null     float64
 11  Crime Rate per 1,000  11 non-null     float64
dtypes: float64(10), int64(1), object(1)
memory usage: 1.4+ KB
```

In [78]:
```
# 15.Read in data: Smithfield_Police_Crime_Data_20231018
smithfield = pd.read_csv('./Smithfield_Police_Crime_Data_20231018.csv')
smithfield.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6211 entries, 0 to 6210
Data columns (total 20 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   column 1               6211 non-null   int64  
 1   address_1              6211 non-null   object 
 2   case_number            6211 non-null   int64  
 3   city                   6211 non-null   object 
 4   clearance_type         0 non-null      float64
 5   country                0 non-null      float64
 6   created_at             6211 non-null   object 
 7   day_of_week            6211 non-null   object 
 8   hour_of_day            6211 non-null   int64  
 9   incident_datetime      6211 non-null   object 
 10  incident_description   6211 non-null   object 
 11  incident_id            6211 non-null   int64  
 12  incident_type_primary  6211 non-null   object 
 13  latitude               6211 non-null   float64
 14  location               6211 non-null   object 
 15  longitude              6211 non-null   float64
 16  parent_incident_type   6211 non-null   object 
 17  state                  6211 non-null   object 
 18  updated_at             6211 non-null   object 
 19  zip                    6080 non-null   float64
dtypes: float64(5), int64(4), object(11)
memory usage: 970.6+ KB
```

In [79]:
```
# 15.Re-arrange columns
smithfield = smithfield[['incident_id', 'case_number', 'incident_datetim
               'incident_description','address_1', 'city', 'latitude', '
               'location', 'hour_of_day', 'day_of_week', 'created_at', '
smithfield.head()
```

Out[79]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incide
0	290521772	132895	07/16/2013 09:47:00 AM	Theft	-FRAUD	SCAN
1	717774978	151248	05/21/2015 03:17:00 PM	Alarm	-ALARM	ALARN
2	715047696	151062	05/05/2015 10:06:00 AM	Family Offense	-DOMESTIC/FAMILY INCIDENT	DON
3	715139075	151069	05/05/2015 02:52:00 PM	Traffic	-TRAFFIC CRASH	TR/ TRAF VE
4	715139076	151071	05/05/2015 08:45:00 PM	Drugs	-CONTROLLED SUBSTANCE	

In [80]:
```
# 16.Read in data: South_Ogden_Police_Crime_Data_20231018
s_ogden = pd.read_csv('./South_Ogden_Police_Crime_Data_20231018.csv')
s_ogden.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15000 entries, 0 to 14999
Data columns (total 20 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   column 1                15000 non-null  int64  
 1   address_1               14987 non-null  object 
 2   case_number             15000 non-null  object 
 3   city                    15000 non-null  object 
 4   clearance_type          0 non-null      float64
 5   country                 0 non-null      float64
 6   created_at              15000 non-null  object 
 7   day_of_week             15000 non-null  object 
 8   hour_of_day             15000 non-null  int64  
 9   incident_datetime       15000 non-null  object 
 10  incident_description    15000 non-null  object 
 11  incident_id             15000 non-null  int64  
 12  incident_type_primary   15000 non-null  object 
 13  latitude                15000 non-null  float64
 14  location                14987 non-null  object 
 15  longitude               15000 non-null  float64
 16  parent_incident_type    15000 non-null  object 
 17  state                   15000 non-null  object 
 18  updated_at              15000 non-null  object 
 19  zip                     4032 non-null   float64
dtypes: float64(5), int64(3), object(12)
memory usage: 2.3+ MB
```

In [81]:
```
# 16.Re-arrange columns
s_ogden = s_ogden[['incident_id', 'case_number', 'incident_datetime','pa
               'incident_description','address_1', 'city', 'latitude', ''
               'location', 'hour_of_day', 'day_of_week', 'created_at', 'u
s_ogden.head()
```

Out[81]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incider
0	748390201	16-1345	01/30/2016 09:11:18 PM	Traffic	Traffic	TRAF
1	640185082	14-17947	12/19/2014 12:13:20 AM	Other	Alarm	ALAI
2	650733862	15-16	01/01/2015 02:10:20 PM	Other	Other	KEE
3	660620611	15-456	01/12/2015 01:00:30 PM	Theft	Theft	
4	667748129	15-712	01/19/2015 07:46:02 AM	Other	Alarm	ALAI

In [82]:
```
# 17.Read in data: Sunset_Police_Crime_Data_20231018
sunset = pd.read_csv('./Sunset_Police_Crime_Data_20231018.csv')
sunset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15000 entries, 0 to 14999
Data columns (total 20 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Unnamed Column          15000 non-null  int64  
 1   address_1               14982 non-null  object 
 2   case_number             15000 non-null  int64  
 3   city                    14992 non-null  object 
 4   clearance_type          0 non-null      float64
 5   country                 0 non-null      float64
 6   created_at              15000 non-null  object 
 7   day_of_week             15000 non-null  object 
 8   hour_of_day             15000 non-null  int64  
 9   incident_datetime       15000 non-null  object 
 10  incident_description    15000 non-null  object 
 11  incident_id             15000 non-null  int64  
 12  incident_type_primary   15000 non-null  object 
 13  latitude                14983 non-null  float64
 14  location                14982 non-null  object 
 15  longitude               14983 non-null  float64
 16  parent_incident_type    15000 non-null  object 
 17  state                   15000 non-null  object 
 18  updated_at              15000 non-null  object 
 19  zip                     14331 non-null  object 
dtypes: float64(4), int64(4), object(12)
memory usage: 2.3+ MB
```

In [83]:
```
# 17.Re-arrange columns
sunset = sunset[['incident_id', 'case_number', 'incident_datetime','pare
                 'incident_description','address_1', 'city', 'latitude', '
                 'location', 'hour_of_day', 'day_of_week', 'created_at', '
sunset.head()
```

Out[83]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incider
0	745069660	1602372	01/15/2016 07:32:55 AM	Community Policing	[CFS] SCHOOL ZONE ENFORCEMENT	S EI
1	746646114	1603443	01/22/2016 05:58:00 AM	Other	[CFS] 1050 PD	
2	746646113	1603448	01/22/2016 07:40:10 AM	Community Policing	[CFS] CITIZEN REQUESTING INFORMATION	I
3	788938238	1702064	01/12/2017 03:09:02 AM	Other	[CFS] 1050 PD	
4	788938237	1702067	01/12/2017 03:45:25 AM	Other	[CFS] 1075	

In [84]:
```
# 18.Read in data: Syracuse_Police_Crime_Data_20231018
syracuse = pd.read_csv('./Syracuse_Police_Crime_Data_20231018.csv')
syracuse.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15000 entries, 0 to 14999
Data columns (total 18 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   column 1               15000 non-null  int64  
 1   address_1              15000 non-null  object 
 2   case_number            15000 non-null  object 
 3   city                   15000 non-null  object 
 4   clearance_type         0 non-null      float64
 5   created_at             15000 non-null  object 
 6   day_of_week            15000 non-null  object 
 7   hour_of_day            15000 non-null  int64  
 8   incident_datetime      15000 non-null  object 
 9   incident_description   15000 non-null  object 
 10  incident_id            15000 non-null  int64  
 11  incident_type_primary  15000 non-null  object 
 12  latitude               15000 non-null  float64
 13  location               15000 non-null  object 
 14  longitude              15000 non-null  float64
 15  parent_incident_type   15000 non-null  object 
 16  state                  15000 non-null  object 
 17  updated_at             15000 non-null  object 
dtypes: float64(3), int64(3), object(12)
memory usage: 2.1+ MB
```

In [85]:
```
# 18.Re-arrange columns
syracuse = syracuse[['incident_id', 'case_number', 'incident_datetime','
                'incident_description','address_1', 'city', 'latitude', '
                'location', 'hour_of_day', 'day_of_week', 'created_at', '
syracuse.head()
```

Out[85]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incider
0	833897173	C1857588	02/10/2018 12:47:04 AM	Vehicle Stop	CAD: Traffic Stop	
1	833897172	C1857600	02/10/2018 01:22:28 AM	Vehicle Stop	CAD: Traffic Stop	
2	833897171	C1857604	02/10/2018 01:33:07 AM	Vehicle Stop	CAD: Traffic Stop	
3	833897170	C1857606	02/10/2018 01:39:09 AM	Other	CAD: Warrant Service	W
4	833897169	C1857608	02/10/2018 01:40:24 AM	Community Policing	CAD: Susp Circumstance	Susp

```
In [86]: # 19.Read in data: Utah_County_Sheriff_Crime_Incident_Data_20231018
         utah = pd.read_csv('./Utah_County_Sheriff_Crime_Incident_Data_20231018.c
         utah.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 72400 entries, 0 to 72399
Data columns (total 20 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   incident_id            72400 non-null  int64  
 1   case_number            72400 non-null  object 
 2   incident_datetime      72400 non-null  object 
 3   incident_type_primary  72400 non-null  object 
 4   incident_description   72400 non-null  object 
 5   clearance_type         0 non-null      float64
 6   address_1              72381 non-null  object 
 7   address_2              0 non-null      float64
 8   city                   72399 non-null  object 
 9   state                  72400 non-null  object 
 10  zip                    3266 non-null   float64
 11  country                22156 non-null  object 
 12  latitude               72400 non-null  float64
 13  longitude              72400 non-null  float64
 14  created_at             72400 non-null  object 
 15  updated_at             72400 non-null  object 
 16  location               72398 non-null  object 
 17  hour_of_day            72400 non-null  int64  
 18  day_of_week            72400 non-null  object 
 19  parent_incident_type   72400 non-null  object 
dtypes: float64(5), int64(2), object(13)
memory usage: 11.0+ MB

/var/folders/q4/rr159kcn4vb9yhnfzgzd85xh0000gn/T/ipykernel_3964/3430271
747.py:2: DtypeWarning: Columns (11) have mixed types. Specify dtype op
tion on import or set low_memory=False.
  utah = pd.read_csv('./Utah_County_Sheriff_Crime_Incident_Data_2023101
8.csv')
```

In [87]:
```
# 19.Re-arrange columns
utah = utah[['incident_id', 'case_number', 'incident_datetime','parent_i
               'incident_description','address_1', 'city', 'latitude', '
               'location', 'hour_of_day', 'day_of_week', 'created_at', '
utah.head()
```

Out[87]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incider
0	4155486	08UC11086	10/05/2008 12:00:00 AM	Other	Weapons Offense	
1	781823043	16UC11433	11/19/2016 12:11:51 AM	Liquor	ALCOHOL OFFENSE	Descr
2	781161909	16UC08720	09/04/2016 12:09:09 PM	Community Policing	ANIMAL PROBLEM	Desc
3	752214707	16UC02132	03/06/2016 06:03:29 PM	Alarm	ALARM	Des
4	771139313	16UC08685	09/03/2016 01:09:26 AM	Traffic	DRIVING UNDER INFLUENCE	Desc U

In [88]:
```
# 20.Read in data: Woods_Cross_Police_Crime_Data_20231018
woods_cross = pd.read_csv('./Woods_Cross_Police_Crime_Data_20231018.csv'
woods_cross.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 49624 entries, 0 to 49623
Data columns (total 20 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   column 1                49624 non-null  int64  
 1   address_1               49598 non-null  object 
 2   case_number             49624 non-null  int64  
 3   city                    49600 non-null  object 
 4   clearance_type          0 non-null      float64
 5   country                 0 non-null      float64
 6   created_at              49624 non-null  object 
 7   day_of_week             49624 non-null  object 
 8   hour_of_day             49624 non-null  int64  
 9   incident_datetime       49624 non-null  object 
 10  incident_description    49624 non-null  object 
 11  incident_id             49624 non-null  int64  
 12  incident_type_primary   49624 non-null  object 
 13  latitude                49602 non-null  float64
 14  location                49587 non-null  object 
 15  longitude               49602 non-null  float64
 16  parent_incident_type    49624 non-null  object 
 17  state                   49624 non-null  object 
 18  updated_at              49624 non-null  object 
 19  zip                     45007 non-null  object 
dtypes: float64(4), int64(4), object(12)
memory usage: 7.6+ MB
```

In [89]:
```
# 20.Re-arrange columns
woods_cross = woods_cross[['incident_id', 'case_number', 'incident_datet
               'incident_description','address_1', 'city', 'latitude', '
               'location', 'hour_of_day', 'day_of_week', 'created_at', '
woods_cross.head()
```

Out[89]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incider
0	745069660	1602372	01/15/2016 07:32:55 AM	Community Policing	[CFS] SCHOOL ZONE ENFORCEMENT	S EN
1	746646114	1603443	01/22/2016 05:58:00 AM	Other	[CFS] 1050 PD	
2	746646113	1603448	01/22/2016 07:40:10 AM	Community Policing	[CFS] CITIZEN REQUESTING INFORMATION	I
3	788938238	1702064	01/12/2017 03:09:02 AM	Other	[CFS] 1050 PD	
4	788938237	1702067	01/12/2017 03:45:25 AM	Other	[CFS] 1075	

In [90]:
```
# Concatenate all the dataframes vertically
df = pd.concat([st_george, beaver, Brigham, cache, ephraim, iron, juab, l
               perry, pleasant_view, price, roy, smithfield, s_ogden, su
               utah, woods_cross], ignore_index=True)
df
```

Out[90]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	i
0	691160812	15P003395	2/11/2015 16:02	Theft	FRAUD	
1	712757140	15P008661	4/13/2015 13:04	Drugs	DRUGS	
2	715044076	15P010661	5/5/2015 22:05	Drugs	DRUGS	
3	825529671	17P027337	11/5/2017 5:11	Traffic	DUI	
4	736930480	15P027109	11/12/2015 23:11	Traffic	DUI	
...	...	...	...	...	...	
479659	808705608	1729112	06/09/2017 08:46:12 AM	Other	[CFS] 1090 COM	
479660	808705606	1729120	06/09/2017 09:26:42 AM	Community Policing	[CFS] OUTSIDE ASSIST	
479661	808705604	1729143	06/09/2017 11:40:39 AM	Community Policing	[CFS] VACATION WATCH	
479662	808705600	1729197	06/09/2017 04:05:21 PM	Other	[CFS] C/S VIOLATIONS	
479663	808705594	1729279	06/10/2017 01:57:04 AM	Disorder	[CFS] NOISE DISTURBANCE	

479664 rows × 15 columns

FEATURE SELECTION

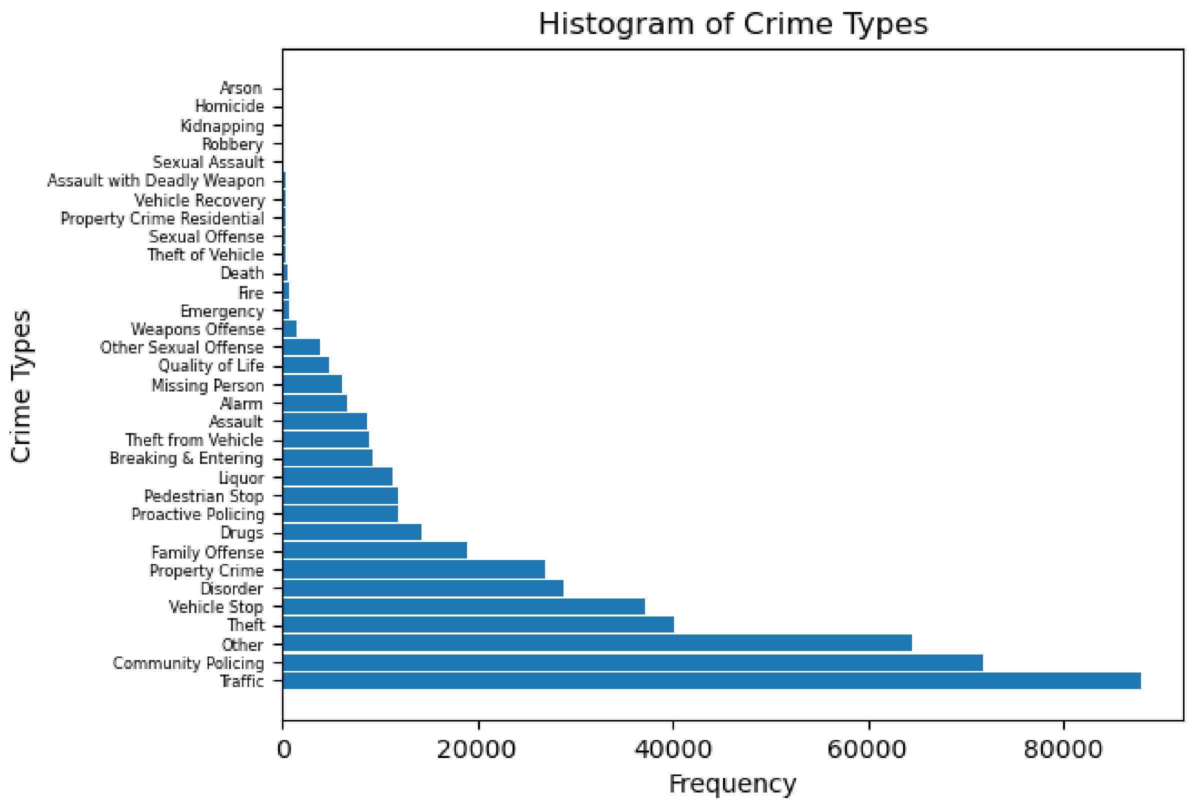
We will need to extract the most relevant features to get a more accurate dataset for our visualization. Also our primary focus will be on the most common types of crimes so we will determine the top 5-10 crimes and place the rest in an 'Misc' category

In [91]:
```
# # Look at the unique values of "parent_incident_type" to decide if we w
unique_values = pd.unique(df['parent_incident_type'])
unique_values
```

Out[91]:
```
array(['Theft', 'Drugs', 'Traffic', 'Disorder', 'Pedestrian Stop',
       'Assault', 'Missing Person', 'Theft from Vehicle',
       'Property Crime', 'Liquor', 'Family Offense', 'Community Policin
g',
       'Vehicle Stop', 'Breaking & Entering', 'Other',
       'Other Sexual Offense', 'Weapons Offense', 'Vehicle Recovery',
       'Kidnapping', 'Assault with Deadly Weapon', 'Alarm', 'Fire',
       'Emergency', 'Theft of Vehicle', 'Robbery', 'Death', 'Arson',
       'Sexual Assault', 'Proactive Policing', 'Quality of Life',
       'Homicide', 'Sexual Offense', 'Property Crime Residential'],
      dtype=object)
```

In [92]:
```
# Do we want to use the 'parent_incident_type' column
value_counts = df['parent_incident_type'].value_counts()
```

In [93]:
```python
import matplotlib.pyplot as plt

# Assuming 'crime_type_column' is the name of the column in your DataFra
crime_type_counts = df['parent_incident_type'].value_counts()

# Extract unique values and their counts
unique_values = crime_type_counts.index
value_counts = crime_type_counts.values

# Set the positions and labels for y-ticks
y_tick_positions = range(len(unique_values))
y_ticks = unique_values

# Create the horizontal bar chart
plt.barh(y_tick_positions, value_counts, height=0.9, align='center')

# Set labels and title
plt.xlabel('Frequency')
plt.ylabel('Crime Types')
plt.title('Histogram of Crime Types')

# Set the y-ticks with custom positions, labels, and font size
plt.yticks(y_tick_positions, y_ticks, fontsize=6)  # Adjust the font siz

# Show the plot
plt.show()
```



In [94]:
```
# List of categories to keep
categories_to_keep = ['Traffic', 'Community Policing', 'Other', 'Theft',

# Function to categorize as 'Misc' if not in the list
def categorize_as_other(category):
    return category if category in categories_to_keep else 'Misc'

# Apply the function to the 'Category' column
df['Crime_Category'] = df['parent_incident_type'].apply(categorize_as_ot

# Display the updated DataFrame
df.head()
```

Out[94]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incider
0	691160812	15P003395	2/11/2015 16:02	Theft	FRAUD	
1	712757140	15P008661	4/13/2015 13:04	Drugs	DRUGS	
2	715044076	15P010661	5/5/2015 22:05	Drugs	DRUGS	
3	825529671	17P027337	11/5/2017 5:11	Traffic	DUI	
4	736930480	15P027109	11/12/2015 23:11	Traffic	DUI	

OUTLIERS

We will also need to determine any outliers in the data and determine if we should remove them or replace them with Median values.

In [95]:
```python
from scipy import stats

# Assuming 'numeric_column' is the name of the column with numerical dat
z_scores = stats.zscore(df['hour_of_day'])
threshold = 3  # You can adjust this threshold as needed

# Find indices of potential outliers
outlier_indices = np.where(np.abs(z_scores) > threshold)

# List the actual data points that are potential outliers
outliers = df.iloc[outlier_indices]
outliers
```

Out[95]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_type_primary	incident

In [96]:
```python
# Detect missing values
df.isna().sum()
# Drop the 88 rows with missing longitude and latitude
# Drop location column
# Drop all N/A
```

Out[96]:
```
incident_id                0
case_number                0
incident_datetime          0
parent_incident_type       0
incident_type_primary      0
incident_description     128
address_1                193
city                      84
latitude                  88
longitude                 88
location                 136
hour_of_day                0
day_of_week                0
created_at                41
updated_at                 0
Crime_Category             0
dtype: int64
```

DATA CLEANUP

Some of the cities within the files were not cities from UTAH, so they needed to be cleanup/removed

In [97]:
```python
# Drop the 'location' and 'incident_type_primary' columns
df = df.drop(['location', 'incident_type_primary'], axis=1)
```

In [98]:
```python
# Drop rows with missing values
df = df.dropna()
df
```

Out[98]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_description	ad
0	691160812	15P003395	2/11/2015 16:02	Theft	FRAUD	
1	712757140	15P008661	4/13/2015 13:04	Drugs	DRUGS	1
2	715044076	15P010661	5/5/2015 22:05	Drugs	DRUGS	I-1
3	825529671	17P027337	11/5/2017 5:11	Traffic	DUI	RI
4	736930480	15P027109	11/12/2015 23:11	Traffic	DUI	
...	...	...	...	...	...	
479659	808705608	1729112	06/09/2017 08:46:12 AM	Other	1090 COM	
479660	808705606	1729120	06/09/2017 09:26:42 AM	Community Policing	OUTSIDE ASSIST	6(
479661	808705604	1729143	06/09/2017 11:40:39 AM	Community Policing	VACATION WATCH	70
479662	808705600	1729197	06/09/2017 04:05:21 PM	Other	C/S VIOLATIONS	9(
479663	808705594	1729279	06/10/2017 01:57:04 AM	Disorder	NOISE DISTURBANCE	

479221 rows × 14 columns

In [99]:
```
# Check to see if rows with missing values were deleted
df.isna().sum()
```

Out[99]:
```
incident_id             0
case_number             0
incident_datetime       0
parent_incident_type    0
incident_description    0
address_1               0
city                    0
latitude                0
longitude               0
hour_of_day             0
day_of_week             0
created_at              0
updated_at              0
Crime_Category          0
dtype: int64
```

In [100]:
```
# Convert the 'city' column to uppercase
df.loc[:, 'city'] = df['city'].str.upper()
```

In [101]:
```
# Prints unique values for cities
cities = df['city'].unique()
cities
```

Out[101]:
```
array(['ST GEORGE', 'WASHINGTON', 'HURRICANE', 'LAVERKIN', 'SANTA CLAR
A',
       'LEEDS', 'WASH CO OTHER', 'IVINS', 'WASHCO STGEORGE', 'COVINGTO
N',
       'APPLE VALLEY', 'WASHCO WASHINGT', 'NEW HARMONY',
       'WINCHESTER HILL', 'PINTURA', 'MOHAVE COUNTY', 'ZION NAT PARK',
       'DIAMOND VALLEY', 'SPRINGDALE', 'BROWSE', 'GUNLOCK', 'VIRGIN',
       'TOQUERVILLE', 'HILDALE', 'CENTRAL', 'CLARK COUNTY', 'PINTO',
       'DAMMERON VALLEY', 'LINCOLN COUNTY', 'PINE VALLEY', 'ROCKVILLE',
       'BROOKSIDE', 'ENTERPRISE', 'VEYO', 'DIXIE COLLEGE', 'IRON COUNT
Y',
       'ENTERPRISE WCSO', 'WASHCO TOQUERVI', 'WASHCO HURRICAN',
       'WASHCO LEEDS', 'WASHCO NEWHARMO', 'WASHCO PINE VLY',
       'WASHCO VEYO', 'WASHCO IVINS', 'SHIVWITS RESERV',
       'WASHCO ENTERPRI', 'MILFORD', 'BEAVER', 'BRIGHAM CITY', 'MANTU
A',
       'GARLAND', 'PERRY', 'BOX ELDER CO', 'WILLARD', 'RIVERDALE',
       'WELLSVILLE', 'FAR WEST', 'TREMONTON', 'HONEYVILLE', 'CORINNE',
       'BRIGHAM', 'THATCHER', 'LOGAN', 'MILLVILLE', 'CACHE COUNTY',
       'HYRUM', 'NORTH LOGAN', 'NIBLEY', 'RICHMOND', 'PROVIDENCE',
       'AMALGA', 'SMITHFIELD', 'PARADISE', 'MENDON', 'LEWISTON',
       'TRENTON', 'NEWTON', 'RIVER HEIGHTS', 'CORNISH', 'HYDE PARK',
       'CLARKSTON', 'PRESTON', 'BENSON', 'LOGAN CANYON', 'EPHRAIM',
       'MANTI', 'MOUNT PLEASANT', 'SANPETE COUNTY', 'FAIRVIEW', 'CHESTE
R',
       'SPRING CITY', 'MORONI', 'CENTERFIELD', 'WALES', 'STERLING',
       'FOUNTAIN GREEN', 'GUNNISON', 'PAROWAN', 'IRON CO CC GRID',
       'CEDAR CITY', 'IRON CO BERYLG', 'ENOCH', 'IRON CO NC G',
       'KANARRAVILLE', 'PARAGONAH', 'IRON CO PAR GRI', 'IRON CO PARGG',
       'IRON CO KANG', 'IRON CO SUMMIT', 'WASH CO NEW HAR', 'BRIAN HEA
D',
       'IRON CO BHG', 'IRON CO MODG', 'DELTA', 'BERYL', 'SUMMIT',
       'SAINT GEORGE', 'NEWCASTLE', 'IRON CO ENTERG', 'MILLARD CO DEL
T',
       'NEW CASTLE', 'BRIANHEAD', 'HAMILTON FORT', 'MODENA', 'LUND',
       'BEAVER CO BG', 'HAMBLIN VALLEY', 'WASH CO ENTERPR', 'KANARAVILL
E',
       'MILLARD CO FILL', 'KANE COUNTY DUC', 'MONA', 'JUAB NEPHI',
       'EUREKA', 'LEVAN', 'NEPHI', 'JUAB CO EUR GR', 'JUAB CO NEP GR',
       'LITTLE SAHARA', 'JUAB CO LEV GR', 'JUAB CO MON GR', 'JUAB MON
A',
       'JUAB COUNTY', 'JUAB CO WES GR', 'JUAB WEST DESER', 'JUAB LEVA
N',
       'GOSHEN', 'UTAH COUNTY', 'ROCKY RIDGE', 'LEHI', 'JUAB EUREKA',
       'PROVO', 'KAYSVILLE', 'PARK CITY', 'COALVILLE', 'HOYTSVILLE',
       'ROCKPORT', 'FRANCIS', 'SUMMIT COUNTY', 'HEBER CITY', 'KAMAS',
       'SAMAK', 'WEBER CANYON', 'MARION', 'OAKLEY', 'MURRAY',
       'SALT LAKE CITY', 'WANSHIP', 'MCKINNON', 'MIDWAY', 'PEOA',
       'HENEFER', 'PRICE', 'HELPER', 'PRICE, UTAH (CARBON)', 'WELLINGTO
N',
       'CARBONVILLE', 'EMERY COUNTY', 'OTHER', 'EAST CARBON',
       'SPRING GLEN', 'TEMP', 'KENILWORTH', 'ROY', 'SOUTH OGDEN',
       'WOODS CROSS', 'BOUNTIFUL', 'NORTH SALT LAKE', 'WEST BOUNTIFUL',
       'DAVIS COUNTY', 'LAYTON', 'FARMINGTON', 'CENTERVILLE', 'SYRACUS
E',
       'CLEARFIELD', 'SOUTH WEBER', 'COUNTY NW', 'GENOLA', 'VINEYARD',
       'WOODLAND HILLS', 'EAGLE MOUNTAIN', 'ELK RIDGE', 'CEDAR FORT',
       'SPANISH FORK', 'AMERICAN FORK', 'OREM', 'FAIRFIELD', 'LINDON',
       'SANTAQUIN', 'SPRINGVILLE', 'PAYSON', 'MAPLETON', 'CEDAR HILLS',
       'HIGHLAND', 'PLEASANT GROVE', 'SALEM', 'ALPINE', 'SARATOGA SPRIN
G',
       'DRAPER', 'TOOELE COUNTY', 'SPRING LAKE', 'BLUFFDALE',
       'THISTLE/BIRDSEY', 'UTAH VALLEY U', 'PALMYRA', 'CARBON COUNTY',
       'SUNDANCE', 'ELBERTA', 'SALT LAKE CNTY', 'LAKE SHORE',
       'COVERED BRIDGE', 'LELAND', 'BENJAMIN', 'WASATCH COUNTY',
       'BYU CAMPUS', 'UINTAH', 'DUCHESNE', 'MOAB', 'SALT LAKE', 'EMER
Y',
       'KANAB', 'NEVADA', 'SALT LAKE COUNT', 'SANTA ANA', 'HEBER',
       'FILLMORE', 'OGDEN', 'IDAHO FALLS', 'RICHFIELD',
       'SARATOGA SPRINGS', 'WEBER COUNTY', 'TOOELE', 'SAN FRANCISCO',
       'POCATELLO', 'PALM SPRINGS', 'FLORENCE', 'LOS ANGELES',
       'CEDAR VALLEY', 'PROVOST', 'YINEYARD', 'CLINTON', 'FRUIT HEIGHT
S'],
      dtype=object)
```

In [102]:
```python
unique_city_count = df['city'].nunique()
print("Number of unique cities:", unique_city_count)
```

```
Number of unique cities: 262
```

In [103]:
```python
cities_to_exclude = ['SANTA ANA', 'NEVADA', 'IDAHO FALLS','SAN FRANCISCO
                     'PALM SPRINGS', 'FLORENCE','POCATELLO', 'LOS ANGELE

df = df[~df['city'].isin(cities_to_exclude)]
```

In [104]:
```
# Prints unique values for cities
cities = df['city'].unique()
cities
```

Out[104]:
```
array(['ST GEORGE', 'WASHINGTON', 'HURRICANE', 'LAVERKIN', 'SANTA CLAR
A',
       'LEEDS', 'WASH CO OTHER', 'IVINS', 'WASHCO STGEORGE', 'COVINGTO
N',
       'APPLE VALLEY', 'WASHCO WASHINGT', 'NEW HARMONY',
       'WINCHESTER HILL', 'PINTURA', 'MOHAVE COUNTY', 'ZION NAT PARK',
       'DIAMOND VALLEY', 'SPRINGDALE', 'BROWSE', 'GUNLOCK', 'VIRGIN',
       'TOQUERVILLE', 'HILDALE', 'CENTRAL', 'CLARK COUNTY', 'PINTO',
       'DAMMERON VALLEY', 'LINCOLN COUNTY', 'PINE VALLEY', 'ROCKVILLE',
       'BROOKSIDE', 'ENTERPRISE', 'VEYO', 'DIXIE COLLEGE', 'IRON COUNT
Y',
       'ENTERPRISE WCSO', 'WASHCO TOQUERVI', 'WASHCO HURRICAN',
       'WASHCO LEEDS', 'WASHCO NEWHARMO', 'WASHCO PINE VLY',
       'WASHCO VEYO', 'WASHCO IVINS', 'SHIVWITS RESERV',
       'WASHCO ENTERPRI', 'MILFORD', 'BEAVER', 'BRIGHAM CITY', 'MANTU
A',
       'GARLAND', 'PERRY', 'BOX ELDER CO', 'WILLARD', 'RIVERDALE',
       'WELLSVILLE', 'FAR WEST', 'TREMONTON', 'HONEYVILLE', 'CORINNE',
       'BRIGHAM', 'THATCHER', 'LOGAN', 'MILLVILLE', 'CACHE COUNTY',
       'HYRUM', 'NORTH LOGAN', 'NIBLEY', 'RICHMOND', 'PROVIDENCE',
       'AMALGA', 'SMITHFIELD', 'PARADISE', 'MENDON', 'LEWISTON',
       'TRENTON', 'NEWTON', 'RIVER HEIGHTS', 'CORNISH', 'HYDE PARK',
       'CLARKSTON', 'PRESTON', 'BENSON', 'LOGAN CANYON', 'EPHRAIM',
       'MANTI', 'MOUNT PLEASANT', 'SANPETE COUNTY', 'FAIRVIEW', 'CHESTE
R',
       'SPRING CITY', 'MORONI', 'CENTERFIELD', 'WALES', 'STERLING',
       'FOUNTAIN GREEN', 'GUNNISON', 'PAROWAN', 'IRON CO CC GRID',
       'CEDAR CITY', 'IRON CO BERYLG', 'ENOCH', 'IRON CO NC G',
       'KANARRAVILLE', 'PARAGONAH', 'IRON CO PAR GRI', 'IRON CO PARGG',
       'IRON CO KANG', 'IRON CO SUMMIT', 'WASH CO NEW HAR', 'BRIAN HEA
D',
       'IRON CO BHG', 'IRON CO MODG', 'DELTA', 'BERYL', 'SUMMIT',
       'SAINT GEORGE', 'NEWCASTLE', 'IRON CO ENTERG', 'MILLARD CO DEL
T',
       'NEW CASTLE', 'BRIANHEAD', 'HAMILTON FORT', 'MODENA', 'LUND',
       'BEAVER CO BG', 'HAMBLIN VALLEY', 'WASH CO ENTERPR', 'KANARAVILL
E',
       'MILLARD CO FILL', 'KANE COUNTY DUC', 'MONA', 'JUAB NEPHI',
       'EUREKA', 'LEVAN', 'NEPHI', 'JUAB CO EUR GR', 'JUAB CO NEP GR',
       'LITTLE SAHARA', 'JUAB CO LEV GR', 'JUAB CO MON GR', 'JUAB MON
A',
       'JUAB COUNTY', 'JUAB CO WES GR', 'JUAB WEST DESER', 'JUAB LEVA
N',
       'GOSHEN', 'UTAH COUNTY', 'ROCKY RIDGE', 'LEHI', 'JUAB EUREKA',
       'PROVO', 'KAYSVILLE', 'PARK CITY', 'COALVILLE', 'HOYTSVILLE',
       'ROCKPORT', 'FRANCIS', 'SUMMIT COUNTY', 'HEBER CITY', 'KAMAS',
       'SAMAK', 'WEBER CANYON', 'MARION', 'OAKLEY', 'MURRAY',
       'SALT LAKE CITY', 'WANSHIP', 'MCKINNON', 'MIDWAY', 'PEOA',
       'HENEFER', 'PRICE', 'HELPER', 'PRICE, UTAH (CARBON)', 'WELLINGTO
N',
       'CARBONVILLE', 'EMERY COUNTY', 'OTHER', 'EAST CARBON',
       'SPRING GLEN', 'TEMP', 'KENILWORTH', 'ROY', 'SOUTH OGDEN',
       'WOODS CROSS', 'BOUNTIFUL', 'NORTH SALT LAKE', 'WEST BOUNTIFUL',
       'DAVIS COUNTY', 'LAYTON', 'FARMINGTON', 'CENTERVILLE', 'SYRACUS
E',
       'CLEARFIELD', 'SOUTH WEBER', 'COUNTY NW', 'GENOLA', 'VINEYARD',
       'WOODLAND HILLS', 'EAGLE MOUNTAIN', 'ELK RIDGE', 'CEDAR FORT',
       'SPANISH FORK', 'AMERICAN FORK', 'OREM', 'FAIRFIELD', 'LINDON',
       'SANTAQUIN', 'SPRINGVILLE', 'PAYSON', 'MAPLETON', 'CEDAR HILLS',
       'HIGHLAND', 'PLEASANT GROVE', 'SALEM', 'ALPINE', 'SARATOGA SPRIN
G',
       'DRAPER', 'TOOELE COUNTY', 'SPRING LAKE', 'BLUFFDALE',
       'THISTLE/BIRDSEY', 'UTAH VALLEY U', 'PALMYRA', 'CARBON COUNTY',
       'SUNDANCE', 'ELBERTA', 'SALT LAKE CNTY', 'LAKE SHORE',
       'COVERED BRIDGE', 'LELAND', 'BENJAMIN', 'WASATCH COUNTY',
       'BYU CAMPUS', 'UINTAH', 'DUCHESNE', 'MOAB', 'SALT LAKE', 'EMER
Y',
       'KANAB', 'SALT LAKE COUNT', 'HEBER', 'FILLMORE', 'OGDEN',
       'RICHFIELD', 'SARATOGA SPRINGS', 'WEBER COUNTY', 'TOOELE',
       'CEDAR VALLEY', 'PROVOST', 'YINEYARD', 'CLINTON', 'FRUIT HEIGHT
S'],
      dtype=object)
```

In [105]:
```
cities_to_exclude2 = ['WASH CO OTHER', 'WASHCO STGEORGE', 'COVINGTON', 'W
                      'BROWSE','CLARK COUNTY', 'LINCOLN COUNTY', 'ENTERPRI
                      'WASHCO LEEDS', 'WASHCO NEWHARMO', 'WASHCO PINE VLY
                      'SHIVWITS RESERV','WASHCO ENTERPRI','BOX ELDER CO',
                      'IRON CO NC G','IRON CO PAR GRI','IRON CO PARGG','IR
                      'IRON CO BHG', 'IRON CO MODG', 'IRON CO ENTERG', 'MI

df = df[~df['city'].isin(cities_to_exclude2)]
```

In [106]:
```
# Find rows where 'city' column is equal to "New Castle"
new_castle_rows = df[df['city'] == 'NEW CASTLE']
new_castle_rows
```

Out[106]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_description	ad
232315	761991986	16-01490	06/11/2016 12:00:00 AM	Other	Domestic Prob	20
234120	820251554	17-02744	09/25/2017 12:00:00 AM	Emergency	Medical	

In [107]:
```
# Replace 'NEW CASTLE' with 'NEWCASTLE' in the 'city' column
df['city'] = df['city'].str.replace('NEW CASTLE', 'NEWCASTLE')
```

In [108]:
```
# Replace 'NEW CASTLE' with 'NEWCASTLE' in the 'city' column
df['city'] = df['city'].str.replace('BRIANHEAD', 'BRIAN HEAD')
```

In [109]:
```
# Prints unique values for cities
cities = df['city'].unique()
cities
```

Out[109]:
```
array(['ST GEORGE', 'WASHINGTON', 'HURRICANE', 'LAVERKIN', 'SANTA CLAR
A',
       'LEEDS', 'IVINS', 'APPLE VALLEY', 'NEW HARMONY', 'WINCHESTER HIL
L',
       'PINTURA', 'ZION NAT PARK', 'DIAMOND VALLEY', 'SPRINGDALE',
       'GUNLOCK', 'VIRGIN', 'TOQUERVILLE', 'HILDALE', 'CENTRAL', 'PINT
O',
       'DAMMERON VALLEY', 'PINE VALLEY', 'ROCKVILLE', 'BROOKSIDE',
       'ENTERPRISE', 'VEYO', 'DIXIE COLLEGE', 'IRON COUNTY', 'MILFORD',
       'BEAVER', 'BRIGHAM CITY', 'MANTUA', 'GARLAND', 'PERRY', 'WILLAR
D',
       'RIVERDALE', 'WELLSVILLE', 'FAR WEST', 'TREMONTON', 'HONEYVILL
E',
       'CORINNE', 'BRIGHAM', 'THATCHER', 'LOGAN', 'MILLVILLE', 'HYRUM',
       'NORTH LOGAN', 'NIBLEY', 'RICHMOND', 'PROVIDENCE', 'AMALGA',
       'SMITHFIELD', 'PARADISE', 'MENDON', 'LEWISTON', 'TRENTON',
       'NEWTON', 'RIVER HEIGHTS', 'CORNISH', 'HYDE PARK', 'CLARKSTON',
       'PRESTON', 'BENSON', 'LOGAN CANYON', 'EPHRAIM', 'MANTI',
       'MOUNT PLEASANT', 'SANPETE COUNTY', 'FAIRVIEW', 'CHESTER',
       'SPRING CITY', 'MORONI', 'CENTERFIELD', 'WALES', 'STERLING',
       'FOUNTAIN GREEN', 'GUNNISON', 'PAROWAN', 'CEDAR CITY', 'ENOCH',
       'KANARRAVILLE', 'PARAGONAH', 'BRIAN HEAD', 'DELTA', 'BERYL',
       'SUMMIT', 'SAINT GEORGE', 'NEWCASTLE', 'HAMILTON FORT', 'MODEN
A',
       'LUND', 'BEAVER CO BG', 'HAMBLIN VALLEY', 'WASH CO ENTERPR',
       'KANARAVILLE', 'MILLARD CO FILL', 'KANE COUNTY DUC', 'MONA',
       'JUAB NEPHI', 'EUREKA', 'LEVAN', 'NEPHI', 'JUAB CO EUR GR',
       'JUAB CO NEP GR', 'LITTLE SAHARA', 'JUAB CO LEV GR',
       'JUAB CO MON GR', 'JUAB MONA', 'JUAB COUNTY', 'JUAB CO WES GR',
       'JUAB WEST DESER', 'JUAB LEVAN', 'GOSHEN', 'UTAH COUNTY',
       'ROCKY RIDGE', 'LEHI', 'JUAB EUREKA', 'PROVO', 'KAYSVILLE',
       'PARK CITY', 'COALVILLE', 'HOYTSVILLE', 'ROCKPORT', 'FRANCIS',
       'SUMMIT COUNTY', 'HEBER CITY', 'KAMAS', 'SAMAK', 'WEBER CANYON',
       'MARION', 'OAKLEY', 'MURRAY', 'SALT LAKE CITY', 'WANSHIP',
       'MCKINNON', 'MIDWAY', 'PEOA', 'HENEFER', 'PRICE', 'HELPER',
       'PRICE, UTAH (CARBON)', 'WELLINGTON', 'CARBONVILLE',
       'EMERY COUNTY', 'OTHER', 'EAST CARBON', 'SPRING GLEN', 'TEMP',
       'KENILWORTH', 'ROY', 'SOUTH OGDEN', 'WOODS CROSS', 'BOUNTIFUL',
       'NORTH SALT LAKE', 'WEST BOUNTIFUL', 'DAVIS COUNTY', 'LAYTON',
       'FARMINGTON', 'CENTERVILLE', 'SYRACUSE', 'CLEARFIELD',
       'SOUTH WEBER', 'COUNTY NW', 'GENOLA', 'VINEYARD', 'WOODLAND HILL
S',
       'EAGLE MOUNTAIN', 'ELK RIDGE', 'CEDAR FORT', 'SPANISH FORK',
       'AMERICAN FORK', 'OREM', 'FAIRFIELD', 'LINDON', 'SANTAQUIN',
       'SPRINGVILLE', 'PAYSON', 'MAPLETON', 'CEDAR HILLS', 'HIGHLAND',
       'PLEASANT GROVE', 'SALEM', 'ALPINE', 'SARATOGA SPRING', 'DRAPE
R',
       'TOOELE COUNTY', 'SPRING LAKE', 'BLUFFDALE', 'THISTLE/BIRDSEY',
       'UTAH VALLEY U', 'PALMYRA', 'CARBON COUNTY', 'SUNDANCE', 'ELBERT
A',
       'SALT LAKE CNTY', 'LAKE SHORE', 'COVERED BRIDGE', 'LELAND',
       'BENJAMIN', 'WASATCH COUNTY', 'BYU CAMPUS', 'UINTAH', 'DUCHESN
E',
       'MOAB', 'SALT LAKE', 'EMERY', 'KANAB', 'SALT LAKE COUNT', 'HEBE
R',
       'FILLMORE', 'OGDEN', 'RICHFIELD', 'SARATOGA SPRINGS',
       'WEBER COUNTY', 'TOOELE', 'CEDAR VALLEY', 'PROVOST', 'YINEYARD',
       'CLINTON', 'FRUIT HEIGHTS'], dtype=object)
```

In [110]:
```
cities_to_exclude3 = ['BEAVER CO BG','WASH CO ENTERPR','WASH CO ENTERPR'
                      'JUAB NEPHI','JUAB CO EUR GR','JUAB CO NEP GR', 'JU
                      'JUAB COUNTY', 'JUAB CO WES GR','JUAB WEST DESER',

df = df[~df['city'].isin(cities_to_exclude3)]
```

In [111]:
```
# Prints unique values for cities
cities = df['city'].unique()
cities
```

Out[111]:
```
array(['ST GEORGE', 'WASHINGTON', 'HURRICANE', 'LAVERKIN', 'SANTA CLAR
A',
       'LEEDS', 'IVINS', 'APPLE VALLEY', 'NEW HARMONY', 'WINCHESTER HIL
L',
       'PINTURA', 'ZION NAT PARK', 'DIAMOND VALLEY', 'SPRINGDALE',
       'GUNLOCK', 'VIRGIN', 'TOQUERVILLE', 'HILDALE', 'CENTRAL', 'PINT
O',
       'DAMMERON VALLEY', 'PINE VALLEY', 'ROCKVILLE', 'BROOKSIDE',
       'ENTERPRISE', 'VEYO', 'DIXIE COLLEGE', 'IRON COUNTY', 'MILFORD',
       'BEAVER', 'BRIGHAM CITY', 'MANTUA', 'GARLAND', 'PERRY', 'WILLAR
D',
       'RIVERDALE', 'WELLSVILLE', 'FAR WEST', 'TREMONTON', 'HONEYVILL
E',
       'CORINNE', 'BRIGHAM', 'THATCHER', 'LOGAN', 'MILLVILLE', 'HYRUM',
       'NORTH LOGAN', 'NIBLEY', 'RICHMOND', 'PROVIDENCE', 'AMALGA',
       'SMITHFIELD', 'PARADISE', 'MENDON', 'LEWISTON', 'TRENTON',
       'NEWTON', 'RIVER HEIGHTS', 'CORNISH', 'HYDE PARK', 'CLARKSTON',
       'PRESTON', 'BENSON', 'LOGAN CANYON', 'EPHRAIM', 'MANTI',
       'MOUNT PLEASANT', 'SANPETE COUNTY', 'FAIRVIEW', 'CHESTER',
       'SPRING CITY', 'MORONI', 'CENTERFIELD', 'WALES', 'STERLING',
       'FOUNTAIN GREEN', 'GUNNISON', 'PAROWAN', 'CEDAR CITY', 'ENOCH',
       'KANARRAVILLE', 'PARAGONAH', 'BRIAN HEAD', 'DELTA', 'BERYL',
       'SUMMIT', 'SAINT GEORGE', 'NEWCASTLE', 'HAMILTON FORT', 'MODEN
A',
       'LUND', 'HAMBLIN VALLEY', 'KANARAVILLE', 'MONA', 'EUREKA', 'LEVA
N',
       'NEPHI', 'LITTLE SAHARA', 'GOSHEN', 'UTAH COUNTY', 'ROCKY RIDG
E',
       'LEHI', 'JUAB EUREKA', 'PROVO', 'KAYSVILLE', 'PARK CITY',
       'COALVILLE', 'HOYTSVILLE', 'ROCKPORT', 'FRANCIS', 'SUMMIT COUNT
Y',
       'HEBER CITY', 'KAMAS', 'SAMAK', 'WEBER CANYON', 'MARION', 'OAKLE
Y',
       'MURRAY', 'SALT LAKE CITY', 'WANSHIP', 'MCKINNON', 'MIDWAY',
       'PEOA', 'HENEFER', 'PRICE', 'HELPER', 'PRICE, UTAH (CARBON)',
       'WELLINGTON', 'CARBONVILLE', 'EMERY COUNTY', 'OTHER',
       'EAST CARBON', 'SPRING GLEN', 'TEMP', 'KENILWORTH', 'ROY',
       'SOUTH OGDEN', 'WOODS CROSS', 'BOUNTIFUL', 'NORTH SALT LAKE',
       'WEST BOUNTIFUL', 'DAVIS COUNTY', 'LAYTON', 'FARMINGTON',
       'CENTERVILLE', 'SYRACUSE', 'CLEARFIELD', 'SOUTH WEBER',
       'COUNTY NW', 'GENOLA', 'VINEYARD', 'WOODLAND HILLS',
       'EAGLE MOUNTAIN', 'ELK RIDGE', 'CEDAR FORT', 'SPANISH FORK',
       'AMERICAN FORK', 'OREM', 'FAIRFIELD', 'LINDON', 'SANTAQUIN',
       'SPRINGVILLE', 'PAYSON', 'MAPLETON', 'CEDAR HILLS', 'HIGHLAND',
       'PLEASANT GROVE', 'SALEM', 'ALPINE', 'SARATOGA SPRING', 'DRAPE
R',
       'TOOELE COUNTY', 'SPRING LAKE', 'BLUFFDALE', 'THISTLE/BIRDSEY',
       'UTAH VALLEY U', 'PALMYRA', 'CARBON COUNTY', 'SUNDANCE', 'ELBERT
A',
       'SALT LAKE CNTY', 'LAKE SHORE', 'COVERED BRIDGE', 'LELAND',
       'BENJAMIN', 'WASATCH COUNTY', 'BYU CAMPUS', 'UINTAH', 'DUCHESN
E',
       'MOAB', 'SALT LAKE', 'EMERY', 'KANAB', 'SALT LAKE COUNT', 'HEBE
R',
       'FILLMORE', 'OGDEN', 'RICHFIELD', 'SARATOGA SPRINGS',
       'WEBER COUNTY', 'TOOELE', 'CEDAR VALLEY', 'PROVOST', 'YINEYARD',
       'CLINTON', 'FRUIT HEIGHTS'], dtype=object)
```

In [112]:
```
# Replace 'PRICE, UTAH (CARBON)' with 'PRICE' in the 'city' column
df['city'] = df['city'].str.replace('PRICE, UTAH (CARBON)', 'PRICE')
```

In [113]:
```
cities_to_exclude4 = ['JUAB EUREKA','OTHER','DAVIS COUNTY','SUMMIT COUNTY
                      'EMERY COUNTY','COUNTY NW','CARBON COUNTY', 'TOOELE
                     'UTAH COUNTY','WASATCH COUNTY']  # List of cities to

df = df[~df['city'].isin(cities_to_exclude4)]
```

In [114]:
```
# Prints unique values for cities
cities = df['city'].unique()
cities
```

Out[114]:
```
array(['ST GEORGE', 'WASHINGTON', 'HURRICANE', 'LAVERKIN', 'SANTA CLAR
A',
       'LEEDS', 'IVINS', 'APPLE VALLEY', 'NEW HARMONY', 'WINCHESTER HIL
L',
       'PINTURA', 'ZION NAT PARK', 'DIAMOND VALLEY', 'SPRINGDALE',
       'GUNLOCK', 'VIRGIN', 'TOQUERVILLE', 'HILDALE', 'CENTRAL', 'PINT
O',
       'DAMMERON VALLEY', 'PINE VALLEY', 'ROCKVILLE', 'BROOKSIDE',
       'ENTERPRISE', 'VEYO', 'DIXIE COLLEGE', 'IRON COUNTY', 'MILFORD',
       'BEAVER', 'BRIGHAM CITY', 'MANTUA', 'GARLAND', 'PERRY', 'WILLAR
D',
       'RIVERDALE', 'WELLSVILLE', 'FAR WEST', 'TREMONTON', 'HONEYVILL
E',
       'CORINNE', 'BRIGHAM', 'THATCHER', 'LOGAN', 'MILLVILLE', 'HYRUM',
       'NORTH LOGAN', 'NIBLEY', 'RICHMOND', 'PROVIDENCE', 'AMALGA',
       'SMITHFIELD', 'PARADISE', 'MENDON', 'LEWISTON', 'TRENTON',
       'NEWTON', 'RIVER HEIGHTS', 'CORNISH', 'HYDE PARK', 'CLARKSTON',
       'PRESTON', 'BENSON', 'LOGAN CANYON', 'EPHRAIM', 'MANTI',
       'MOUNT PLEASANT', 'FAIRVIEW', 'CHESTER', 'SPRING CITY', 'MORON
I',
       'CENTERFIELD', 'WALES', 'STERLING', 'FOUNTAIN GREEN', 'GUNNISO
N',
       'PAROWAN', 'CEDAR CITY', 'ENOCH', 'KANARRAVILLE', 'PARAGONAH',
       'BRIAN HEAD', 'DELTA', 'BERYL', 'SUMMIT', 'SAINT GEORGE',
       'NEWCASTLE', 'HAMILTON FORT', 'MODENA', 'LUND', 'HAMBLIN VALLE
Y',
       'KANARAVILLE', 'MONA', 'EUREKA', 'LEVAN', 'NEPHI', 'LITTLE SAHAR
A',
       'GOSHEN', 'ROCKY RIDGE', 'LEHI', 'PROVO', 'KAYSVILLE', 'PARK CIT
Y',
       'COALVILLE', 'HOYTSVILLE', 'ROCKPORT', 'FRANCIS', 'HEBER CITY',
       'KAMAS', 'SAMAK', 'WEBER CANYON', 'MARION', 'OAKLEY', 'MURRAY',
       'SALT LAKE CITY', 'WANSHIP', 'MCKINNON', 'MIDWAY', 'PEOA',
       'HENEFER', 'PRICE', 'HELPER', 'WELLINGTON', 'CARBONVILLE',
       'EAST CARBON', 'SPRING GLEN', 'KENILWORTH', 'ROY', 'SOUTH OGDE
N',
       'WOODS CROSS', 'BOUNTIFUL', 'NORTH SALT LAKE', 'WEST BOUNTIFUL',
       'LAYTON', 'FARMINGTON', 'CENTERVILLE', 'SYRACUSE', 'CLEARFIELD',
       'SOUTH WEBER', 'GENOLA', 'VINEYARD', 'WOODLAND HILLS',
       'EAGLE MOUNTAIN', 'ELK RIDGE', 'CEDAR FORT', 'SPANISH FORK',
       'AMERICAN FORK', 'OREM', 'FAIRFIELD', 'LINDON', 'SANTAQUIN',
       'SPRINGVILLE', 'PAYSON', 'MAPLETON', 'CEDAR HILLS', 'HIGHLAND',
       'PLEASANT GROVE', 'SALEM', 'ALPINE', 'SARATOGA SPRING', 'DRAPE
R',
       'SPRING LAKE', 'BLUFFDALE', 'THISTLE/BIRDSEY', 'UTAH VALLEY U',
       'PALMYRA', 'SUNDANCE', 'ELBERTA', 'LAKE SHORE', 'COVERED BRIDG
E',
       'LELAND', 'BENJAMIN', 'BYU CAMPUS', 'UINTAH', 'DUCHESNE', 'MOA
B',
       'SALT LAKE', 'EMERY', 'KANAB', 'HEBER', 'FILLMORE', 'OGDEN',
       'RICHFIELD', 'SARATOGA SPRINGS', 'TOOELE', 'CEDAR VALLEY',
       'PROVOST', 'YINEYARD', 'CLINTON', 'FRUIT HEIGHTS'], dtype=objec
t)
```

In [115]:
```
# Find rows where 'city' column is equal to "SALT LAKE"
SALT_LAKE_rows = df[df['city'] == 'SALT LAKE']
SALT_LAKE_rows
```

Out[115]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_description	ad
382709	2850990	08UC07934	07/21/2008 12:00:00 AM	Proactive Policing	Description: WARRANT	
382929	2973079	08UC08343	07/30/2008 12:00:00 AM	Proactive Policing	Description: WARRANT	
383539	3110001	08UC08828	08/11/2008 12:00:00 AM	Proactive Policing	Description: WARRANT	Bl
384479	3235791	08UC09102	08/18/2008 12:00:00 AM	Proactive Policing	Description: WARRANT	Bl

In [116]:
```
# Drop rows where 'city' column is equal to "SALT LAKE" from the origina
df = df.drop(SALT_LAKE_rows.index)
```

In [117]:
```
# Prints unique values for cities
cities = df['city'].unique()
cities
```

Out[117]:
```
array(['ST GEORGE', 'WASHINGTON', 'HURRICANE', 'LAVERKIN', 'SANTA CLA
RA',
       'LEEDS', 'IVINS', 'APPLE VALLEY', 'NEW HARMONY', 'WINCHESTER H
ILL',
       'PINTURA', 'ZION NAT PARK', 'DIAMOND VALLEY', 'SPRINGDALE',
       'GUNLOCK', 'VIRGIN', 'TOQUERVILLE', 'HILDALE', 'CENTRAL', 'PIN
TO',
       'DAMMERON VALLEY', 'PINE VALLEY', 'ROCKVILLE', 'BROOKSIDE',
       'ENTERPRISE', 'VEYO', 'DIXIE COLLEGE', 'IRON COUNTY', 'MILFOR
D',
       'BEAVER', 'BRIGHAM CITY', 'MANTUA', 'GARLAND', 'PERRY', 'WILLA
RD',
       'RIVERDALE', 'WELLSVILLE', 'FAR WEST', 'TREMONTON', 'HONEYVILL
E',
       'CORINNE', 'BRIGHAM', 'THATCHER', 'LOGAN', 'MILLVILLE', 'HYRU
M',
       'NORTH LOGAN', 'NIBLEY', 'RICHMOND', 'PROVIDENCE', 'AMALGA',
       'SMITHFIELD', 'PARADISE', 'MENDON', 'LEWISTON', 'TRENTON',
       'NEWTON', 'RIVER HEIGHTS', 'CORNISH', 'HYDE PARK', 'CLARKSTO
```

In [118]:
```
# Find duplicate rows
duplicate_rows = df[df.duplicated()]
# Print the resulting DataFrame containing duplicate rows
print("Duplicate Rows:")
duplicate_rows
```

```
Duplicate Rows:
```

Out[118]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_description	ad
99999	544003266	22978	12/1/1997 20:12	Pedestrian Stop	SUSP OTHER	
267811	913039167	19-B02493	04/05/2019 11:41:06 AM	Weapons Offense	Weapon Offense	10 N
267812	832915287	17-B09872	12/22/2017 08:33:10 AM	Other	Suspicious	90 S
267813	832915358	17-B09944	12/22/2017 11:00:00 AM	Traffic	PD Accident	80 W
267814	832915293	17-B09878	12/22/2017 12:16:37 PM	Other	Citizen Assist	M
...	...	...	...	...	...	
445050	378861433	1403980	01/26/2014 05:38:58 PM	Other	HARASSMENT	40
445051	378861434	1403979	01/26/2014 05:33:38 PM	Community Policing	WARRANT SERVICE	H
445052	378861435	1403976	01/26/2014 05:06:35 PM	Community Policing	WARRANT SERVICE	10 S
445053	378861438	1403958	01/26/2014 12:28:34 PM	Community Policing	CIVIL STANDBY	92
445054	378861439	1403954	01/26/2014 11:26:17 AM	Other	1050 PD	50 S O

29905 rows × 14 columns

```
In [119]: # Identify rows that appear more than once
          duplicated_rows = df[df.duplicated(keep=False)]

          # Sort the resulting DataFrame
          duplicated_rows = duplicated_rows.sort_values(by=list(df.columns))

          # Print the resulting sorted DataFrame containing rows that appear more
          print("Rows that Appear More than Once (Sorted):")
          duplicated_rows.head(20)
```

```
Rows that Appear More than Once (Sorted):
```

Out[119]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_description	ad
327731	83579655	1005671	02/15/2010 02:33:45 AM	Other	1047 PER	4(
430131	83579655	1005671	02/15/2010 02:33:45 AM	Other	1047 PER	4(
327948	83581148	1013984	04/20/2010 04:14:27 PM	Other	1090 RES	
430349	83581148	1013984	04/20/2010 04:14:27 PM	Other	1090 RES	
328143	83582141	1020691	06/11/2010 12:45:18 AM	Other	1075	4(
430544	83582141	1020691	06/11/2010 12:45:18 AM	Other	1075	4(
328545	83583929	1034040	09/17/2010 12:34:26 PM	Other	1075	
430945	83583929	1034040	09/17/2010 12:34:26 PM	Other	1075	
328690	83584765	1039807	11/01/2010 09:23:20 PM	Theft	THEFT	
431090	83584765	1039807	11/01/2010 09:23:20 PM	Theft	THEFT	
328135	83587220	1110337	03/19/2011 07:23:44 PM	Community Policing	OUTSIDE ASSIST	5(
430536	83587220	1110337	03/19/2011 07:23:44 PM	Community Policing	OUTSIDE ASSIST	5(
328136	83587221	1110444	03/20/2011 03:48:26 PM	Community Policing	SUSPICIOUS CIRCUMSTANCE	
430537	83587221	1110444	03/20/2011 03:48:26 PM	Community Policing	SUSPICIOUS CIRCUMSTANCE	
328137	83587222	1109999	03/17/2011 11:28:57 AM	Other	911 TRACE	

	incident_id	case_number	incident_datetime	parent_incident_type	incident_description	ad
430538	83587222	1109999	03/17/2011 11:28:57 AM	Other	911 TRACE	
328138	83587223	1110821	03/23/2011 01:27:29 PM	Other	VIN INSPECTION	
430539	83587223	1110821	03/23/2011 01:27:29 PM	Other	VIN INSPECTION	
328139	83587224	1111401	03/28/2011 05:58:33 AM	Other	1090 COM	9
430540	83587224	1111401	03/28/2011 05:58:33 AM	Other	1090 COM	9

In [120]:
```
# Convert 'case_number' column to a consistent type (e.g., string) and t
df['case_number'] = df['case_number'].astype(str)
duplicates = df[df.duplicated(subset='case_number', keep=False)].sort_va

# Print or further process the sorted duplicates
duplicates
```

Out[120]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_description	ad
430131	83579655	1005671	02/15/2010 02:33:45 AM	Other	1047 PER	4
327731	83579655	1005671	02/15/2010 02:33:45 AM	Other	1047 PER	4
327948	83581148	1013984	04/20/2010 04:14:27 PM	Other	1090 RES	
430349	83581148	1013984	04/20/2010 04:14:27 PM	Other	1090 RES	
328143	83582141	1020691	06/11/2010 12:45:18 AM	Other	1075	4
...	...	...	...	...	...	
192907	913011626	57840	4/5/2019 14:57	Traffic	Traffic Hazard	6
130591	544164324	57844	4/8/1999 14:04	Theft	THEFT-RETAIL	
192910	913011622	57844	4/5/2019 20:19	Other	Lockout	9
192914	913082732	57849	4/6/2019 18:56	Traffic	Livestock Probl	8
130306	544164219	57849	4/8/1999 15:04	Traffic	ABANDONED VEHIC	S

77935 rows × 14 columns

In [121]:
```
# Drop duplicates and keep only one instance
df = df.drop_duplicates(keep='first')

# Print the resulting DataFrame without duplicates
print("DataFrame without Duplicates:")
df
```

```
DataFrame without Duplicates:
```

Out[121]:

	incident_id	case_number	incident_datetime	parent_incident_type	incident_description	ad
0	691160812	15P003395	2/11/2015 16:02	Theft	FRAUD	
1	712757140	15P008661	4/13/2015 13:04	Drugs	DRUGS	1
2	715044076	15P010661	5/5/2015 22:05	Drugs	DRUGS	I-1
3	825529671	17P027337	11/5/2017 5:11	Traffic	DUI	RI
4	736930480	15P027109	11/12/2015 23:11	Traffic	DUI	
...	...	...	...	...	...	
479659	808705608	1729112	06/09/2017 08:46:12 AM	Other	1090 COM	
479660	808705606	1729120	06/09/2017 09:26:42 AM	Community Policing	OUTSIDE ASSIST	6(
479661	808705604	1729143	06/09/2017 11:40:39 AM	Community Policing	VACATION WATCH	70
479662	808705600	1729197	06/09/2017 04:05:21 PM	Other	C/S VIOLATIONS	9(
479663	808705594	1729279	06/10/2017 01:57:04 AM	Disorder	NOISE DISTURBANCE	

418575 rows × 14 columns

In [122]:
```
# Prints unique values for case_number column
unique_case_numbers = df['case_number'].nunique()
print("Number of unique case_numbers:", unique_case_numbers)
```

```
Number of unique case_numbers: 409264
```

In [123]:
```
# Prints unique values for incident_id column
unique_incident_ids = df['incident_id'].nunique()
print("Number of unique incident_ids:", unique_incident_ids)
```

```
Number of unique incident_ids: 418575
```

In [124]:
```
# Drop 'case_number' column since it contains duplicate values using "in
# Drop the 'case_number' column
df = df.drop('case_number', axis=1)
df
```

Out[124]:

	incident_id	incident_datetime	parent_incident_type	incident_description	address_1	
0	691160812	2/11/2015 16:02	Theft	FRAUD	BY 21	GE
1	712757140	4/13/2015 13:04	Drugs	DRUGS	1 Block N 200 E	GE
2	715044076	5/5/2015 22:05	Drugs	DRUGS	I-15 SB X8 ONR	GE
3	825529671	11/5/2017 5:11	Traffic	DUI	4800 Block S RIVER RD	GE
4	736930480	11/12/2015 23:11	Traffic	DUI	I-15 NB MM 8	GE
...	...	...	...	...	...	
479659	808705608	06/09/2017 08:46:12 AM	Other	1090 COM	1400 Block S 1800 WEST	W C
479660	808705606	06/09/2017 09:26:42 AM	Community Policing	OUTSIDE ASSIST	600 Block S 700 WEST	W C
479661	808705604	06/09/2017 11:40:39 AM	Community Policing	VACATION WATCH	2000 Block S 700 WEST	W C
479662	808705600	06/09/2017 04:05:21 PM	Other	C/S VIOLATIONS	900 Block W 500 SOUTH	BOUN
479663	808705594	06/10/2017 01:57:04 AM	Disorder	NOISE DISTURBANCE	1800 Block S 1200 WEST	W C

418575 rows × 13 columns

VALIDITY OF DATA VALUES

We will also need to validate the locations of the files, on initial analysis, it was noticed the the latitude and longitudes did not all reside in Utah. As shown in the image/code below. We will likely have to filter latitude and longitude ranges by: 35 thru 42, and -114 thru -108

In [125]:
```
#first we need to select our latitude and longitude range
#selecting our latitude range
loc_df_1 = df[(df["latitude"] >= 35) & (df["latitude"] <= 42)]

#selecting our longitude range
loc_df_2 = loc_df_1[(loc_df_1["longitude"] >= -114) & (loc_df_1["longitu


loc_df_2.head()
```

Out[125]:

	incident_id	incident_datetime	parent_incident_type	incident_description	address_1	city	l
0	691160812	2/11/2015 16:02	Theft	FRAUD	BY 21	ST GEORGE	
6	112983061	4/2/2012 17:04	Disorder	CIVIL	BY 21	ST GEORGE	
7	112983974	4/10/2012 13:04	Theft	THEFT-MISDEMEAN	BY 21	ST GEORGE	
8	112984779	4/15/2012 0:04	Traffic	HIT AND RUN	BY 21	ST GEORGE	
9	112989665	5/21/2012 10:05	Traffic	PARKING PROBLEM	BY 21	ST GEORGE	

DATA SIZE

After merging all the datasets together, we ended up with about 4millions rows of data. This was a lot of data to process, so we needed to train/test split this data to get a smaller amount without losing the variety of the dataset.

In [145]:
```
from sklearn.model_selection import train_test_split
# Assuming 'target_column' is your target variable
X = loc_df_2.drop('incident_id', axis=1)  # Features
y = loc_df_2['Crime_Category']  # Target variable

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1,

X_test['Total'] = X_test.groupby('Crime_Category')['Crime_Category'].tra

print(len(X_test))
print(X_test.head())
```

```
37918
             incident_datetime parent_incident_type  \
323604  06/07/2013 04:57:57 PM  Breaking & Entering
475748  01/02/2012 08:48:33 PM         Vehicle Stop
27969          10/27/2011 0:10                Theft
104652          9/4/1998 17:09       Property Crime
284811  03/26/2013 10:45:00 AM                Drugs

                                 incident_description  \
323604                                 BURGLARY (10-91)
475748                                 T / TRAFFIC STOP
27969                                             FRAUD
104652                                    CRIM MISCHIEF
284811  "CONTROLLED SUBSTANCE / Amphetamine, Sell"

                         address_1         city   latitude   longitude
\
323604           800 Block 5450 S  SOUTH OGDEN  41.164021 -111.960445
475748          1700 Block W 500 S  WOODS CROSS  40.884085 -111.928060
27969   700 Block S INDIAN HILLS DR    ST GEORGE  37.096300 -113.611000
104652     1400 Block N DIXIE DOWNS    ST GEORGE  37.132946 -113.622726
284811         1 Block W MAIN STREET        PRICE  39.599536 -110.811287

        hour_of_day day_of_week              created_at  \
323604           16      Friday  06/08/2013 09:23:37 AM
475748           20      Monday  01/03/2012 09:21:11 AM
27969             0    Thursday         5/23/2012 23:28
104652           17      Friday          9/4/2014 20:44
284811           10     Tuesday  04/16/2013 07:08:49 AM

                    updated_at       Crime_Category  Total
323604  06/21/2013 09:23:33 AM  Breaking & Entering    727
475748  01/09/2012 09:21:12 AM         Vehicle Stop   3381
27969            9/5/2014 3:22                Theft   3330
104652           9/5/2014 1:00       Property Crime   2205
284811  09/21/2013 07:08:01 AM                Drugs   1168
```

Analysis Questions

Primary questions: Our primary focus will be on the most common types of crimes, that we discovered in our initial data analysis

1) Are there crime trends in different cities? 2) Are there crime trends with different years? 3) What different types of crime occurs, and which one is most prelavent per city? 4) Does the day in the week effect the amount of crime and what type of crime occurs?

We would like to learn how to display and visualize this data in an unbiased and straightforward fashion. In the long-run, if a visualization worked it could be used to determine which areas in Utah to live. However, this could arise another issue with how a visualization can be harmful than helpful.

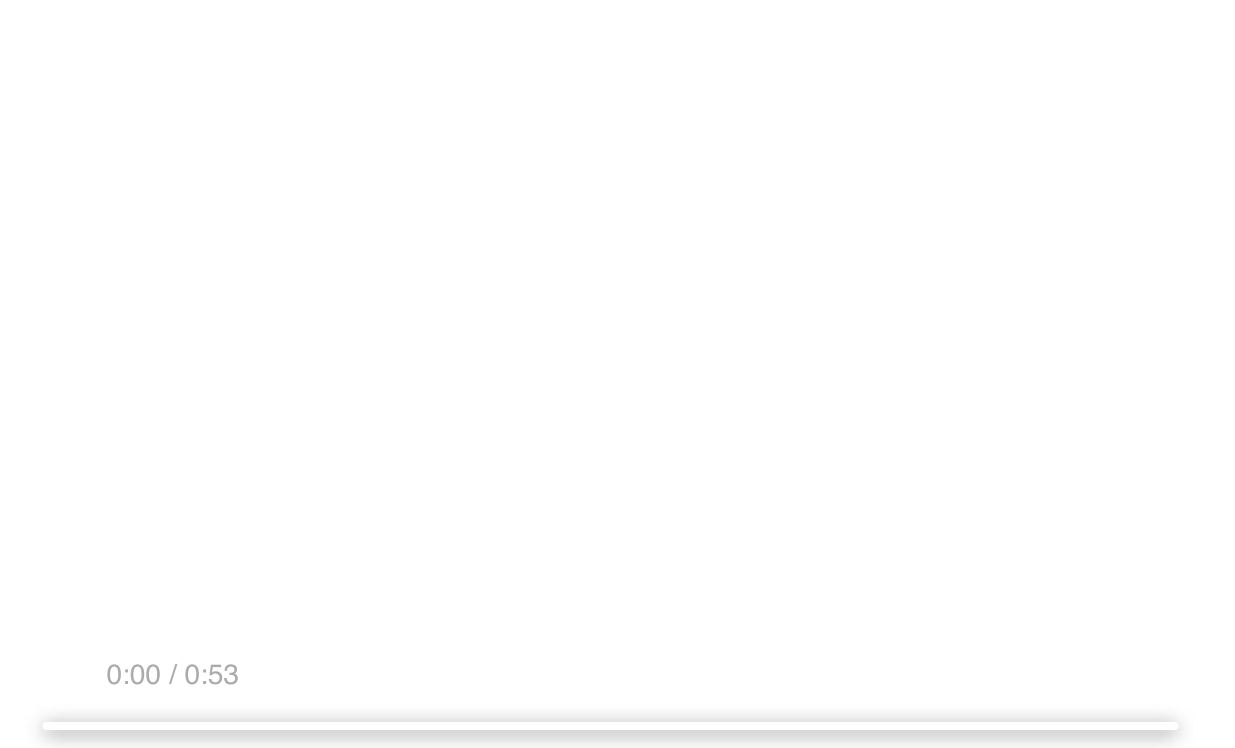
DESIGN IDEAS

MAP of Crime Data

We started with a map of the crime data, which is helpful in it's own way, but we will need to think about how to make this filterable and easier to read. Right now we can kind of see a trend that crime is occuring along the freeway, but you really have to zoom in to see what types of crimes are occuring and where.

In [182]:
```
from IPython.display import HTML

HTML("""
    <video alt="test" controls style="width: 600px; height: 400px;">
        <source src="map.mp4" type="video/mp4">
    </video>
""")
```

Out[182]:



Time Series and Location

The plan for this visualization is to be able to see at where the crime occured at a specific year and at the specific location. The goal is to be able to add a corresponding bar chart to be able to break down the type of crime that also occurred at that location.

Just get the Years of our Dates so we can do a Time Series visualization

In [147]:
```
# Convert the datetime column to datetime format
X_test['datetime_column'] = pd.to_datetime(X_test['created_at'], format=

# Format the datetime column as desired
X_test['formatted_column'] = X_test['datetime_column'].dt.strftime('%m/%

X_test['final_date_column'] = pd.to_datetime(X_test['formatted_column'])

# Extract the year and create a new column
X_test['year_column'] = X_test['final_date_column'].dt.year

print(X_test['year_column'])
```

```
323604    2013
475748    2012
27969     2012
104652    2014
284811    2013
          ...
298521    2012
445849    2014
320939    2013
138519    2014
106708    2014
Name: year_column, Length: 37918, dtype: int32
```

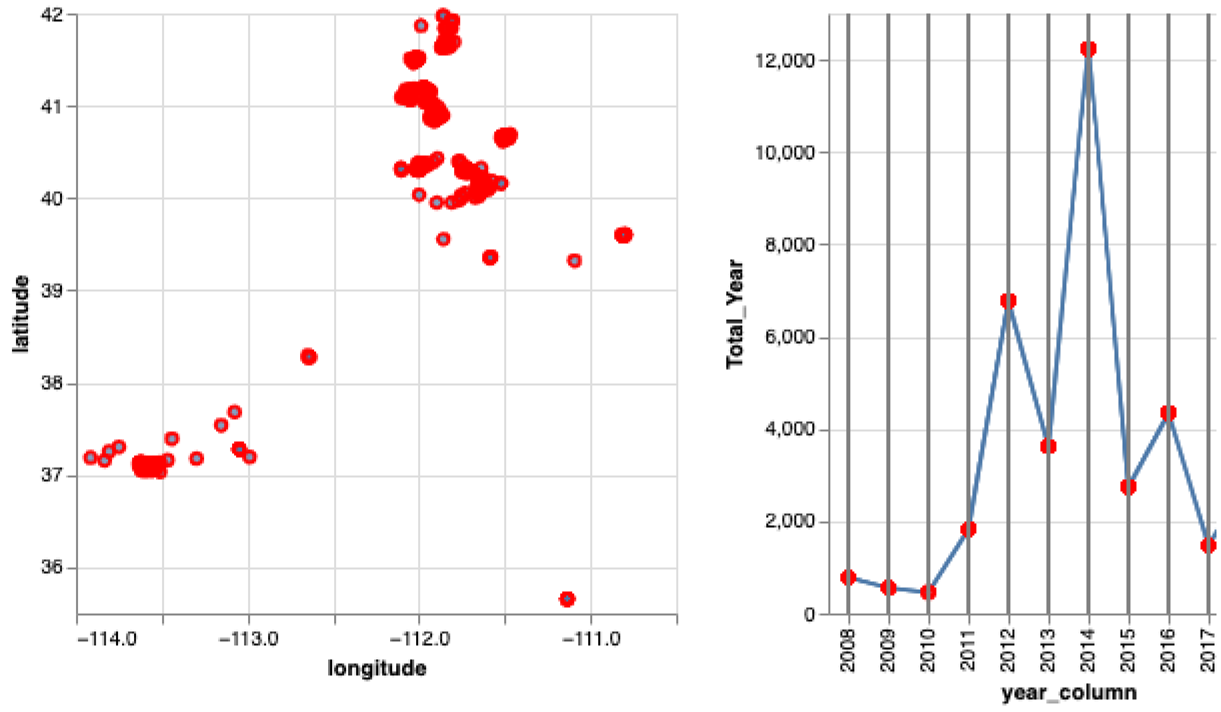
In [148]:
```
# Group by year and count rows
X_test['Total_Year'] = X_test['year_column'].groupby(X_test['year_column
```

In [149]:
```
skip = 50

line = alt.Chart(X_test.iloc[::skip, :]).mark_line(tooltip=True).encode(
    x=alt.X('year_column:O'),
    y=alt.Y('Total_Year:Q', scale=alt.Scale(zero=False))
)

# Create a selection that chooses the nearest point & selects based on x-
nearest = alt.selection(type='single', nearest=True, on='mouseover',
                        fields=['year_column'])

# Transparent selectors across the chart. This is what tells us
# the x-value of the cursor
selectors = alt.Chart(X_test.iloc[::skip, :]).mark_point().encode(
    x='year_column:O',
    opacity=alt.value(0),
).add_params(
    nearest
)

# Draw points on the line, and highlight based on selection
points = line.mark_point(color='red').encode(
    opacity=alt.condition(nearest, alt.value(1), alt.value(0))
)

# Draw a rule at the location of the selection
rules = alt.Chart(X_test.iloc[::skip, :]).mark_rule(color='gray').encode
    x='year_column:O',
).transform_filter(
    nearest
)

#line.mark_line() + selectors + points + rules
```

```
/Users/roannarague/opt/anaconda3/lib/python3.8/site-packages/altair/uti
ls/deprecation.py:65: AltairDeprecationWarning: 'selection' is deprecat
ed.
   Use 'selection_point()' or 'selection_interval()' instead; these fun
ctions also include more helpful docstrings.
  warnings.warn(message, AltairDeprecationWarning, stacklevel=1)
/Users/roannarague/opt/anaconda3/lib/python3.8/site-packages/altair/veg
alite/v5/api.py:450: AltairDeprecationWarning: The types 'single' and
'multi' are now
        combined and should be specified using "selection_point()".
  warnings.warn(
```

In [150]:
```
map = alt.Chart(X_test.iloc[::skip, :]).mark_circle().encode(
    x=alt.X('longitude:Q', scale=alt.Scale(zero=False)),
    y=alt.Y('latitude:Q', scale=alt.Scale(zero=False)),
    order='year_column'
).project('albersUsa')

map_point = map.mark_point(color='red').encode(
    opacity=alt.condition(nearest, alt.value(1), alt.value(0))
)

map + map_point | line + line.mark_line() + selectors + points + rules
```

Out[150]:

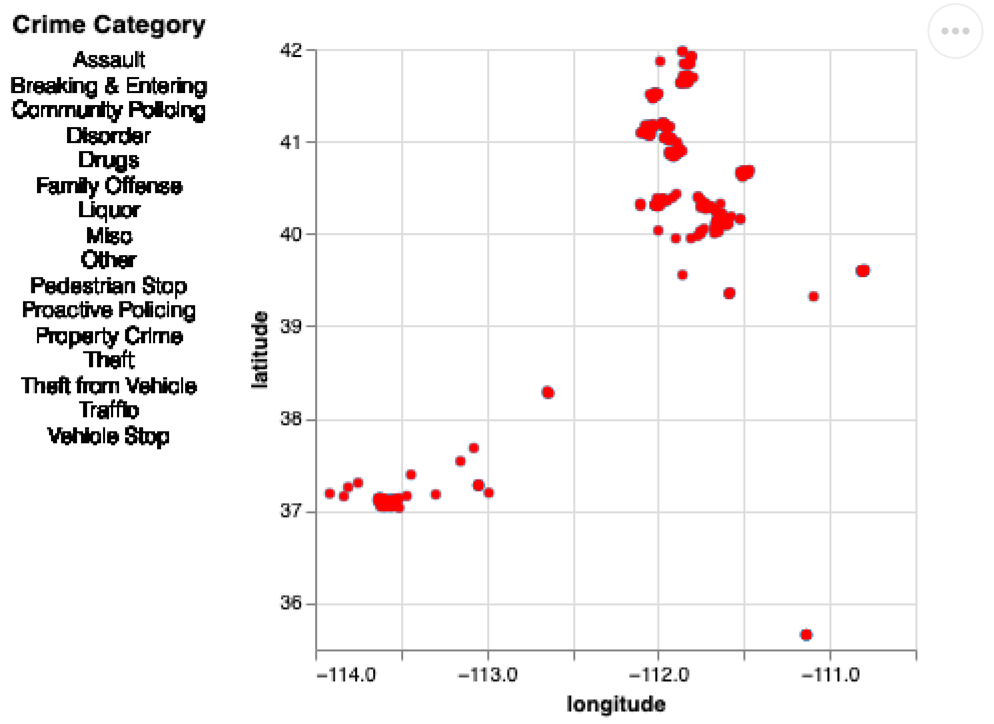


The second graph combines the Type of Crime as a table and then highlights where this crime occured on the map when you click on it. The idea is to somehow include this as a bar chart instead.

In [179]:
```
crime_cat = alt.selection(type='single', nearest=True, on='mouseover',
                          fields=['Crime_Category'])

splits = alt.Chart(X_test.iloc[::skip, :]).mark_text(fontWeight='lighter
    y=alt.Y('Crime_Category:O', axis=None),
    text='Crime_Category:N',
    opacity=alt.value(1)
).properties(
    height=200
).add_params(
    crime_cat
)

map = alt.Chart(X_test.iloc[::skip, :]).mark_circle().encode(
    x=alt.X('longitude:Q', scale=alt.Scale(zero=False)),
    y=alt.Y('latitude:Q', scale=alt.Scale(zero=False)),
    order='created_at'
).project('albersUsa')


map_point = map.mark_circle(color='red', size=20).encode(
    opacity=alt.condition(crime_cat, alt.value(1), alt.value(0))
)

(splits.properties(title="Crime Category") | (map+map_point)).configure_
    strokeWidth=0,
)
```

```
/Users/roannarague/opt/anaconda3/lib/python3.8/site-packages/altair/uti
ls/deprecation.py:65: AltairDeprecationWarning:

'selection' is deprecated.
   Use 'selection_point()' or 'selection_interval()' instead; these fun
ctions also include more helpful docstrings.

/Users/roannarague/opt/anaconda3/lib/python3.8/site-packages/altair/veg
alite/v5/api.py:450: AltairDeprecationWarning:

The types 'single' and 'multi' are now
        combined and should be specified using "selection_point()".
```

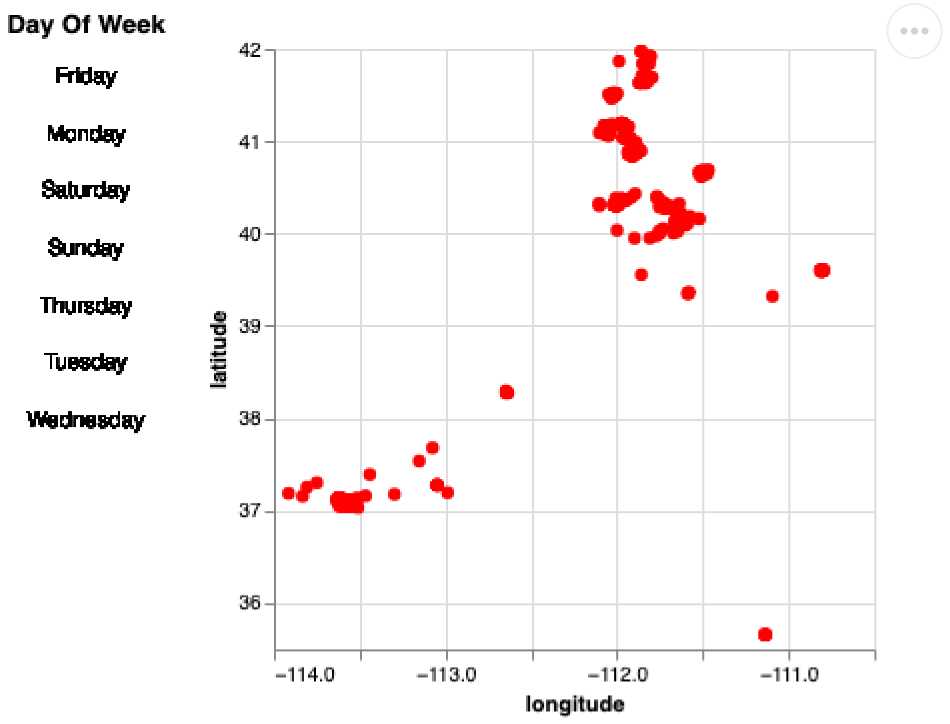
Out[179]:


Crime Category
Assault
Breaking & Entering
Community Policing
Disorder
Drugs
Family Offense
Liquor
Misc
Other
Pedestrian Stop
Proactive Policing
Property Crime
Theft
Theft from Vehicle
Traffic
Vehicle Stop
latitude
42
41
40
39
38
37
36
longitude
−114.0
−113.0
−112.0
−111.0

In [178]:
```
crime_cat = alt.selection(type='single', nearest=True, on='mouseover',
                          fields=['day_of_week'])

splits = alt.Chart(X_test.iloc[::skip, :]).mark_text(fontWeight='lighter
    y=alt.Y('day_of_week:O', axis=None),
    text='day_of_week:N',
    opacity=alt.value(1)
).properties(
    height=200
).add_params(
    crime_cat
)

map = alt.Chart(X_test.iloc[::skip, :]).mark_circle().encode(
    x=alt.X('longitude:Q', scale=alt.Scale(zero=False)),
    y=alt.Y('latitude:Q', scale=alt.Scale(zero=False)),
    order='created_at'
).project('albersUsa')


map_point = map.mark_circle(color='red', size=40).encode(
    opacity=alt.condition(crime_cat, alt.value(1), alt.value(0))
)

(splits.properties(title="Day Of Week") | (map+map_point)).configure_vie
    strokeWidth=0,
)
```

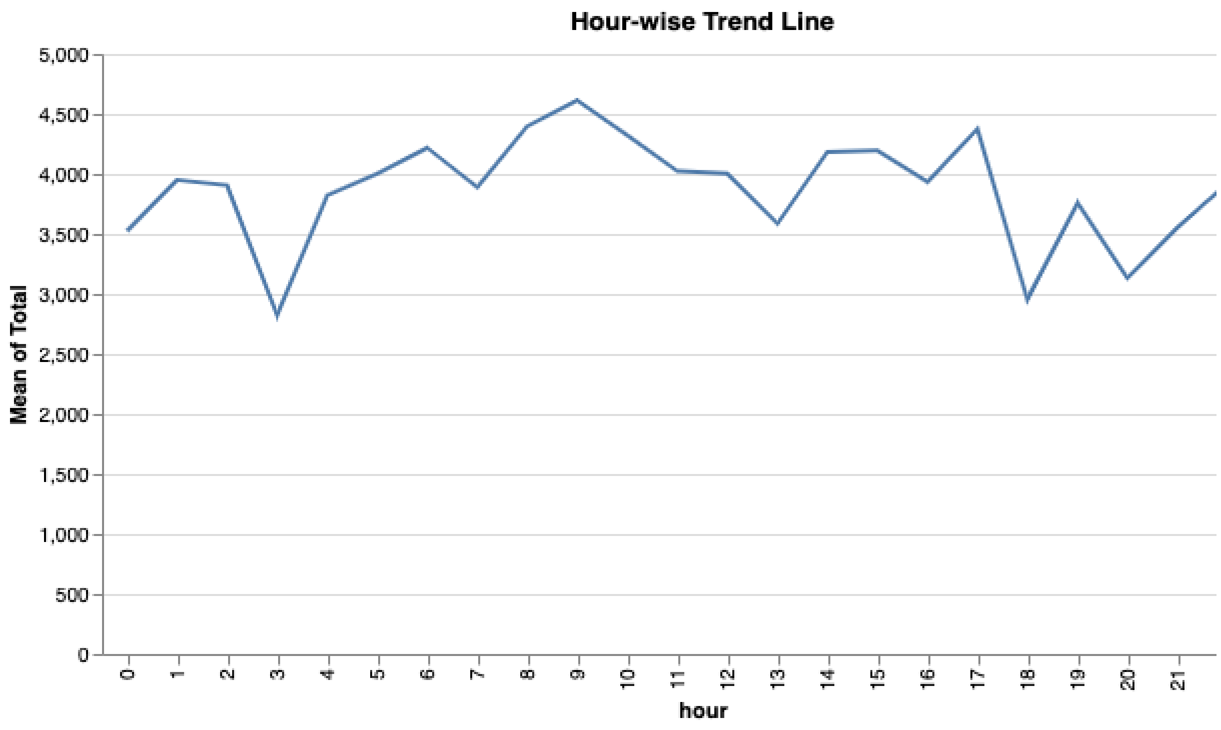
```
/Users/roannarague/opt/anaconda3/lib/python3.8/site-packages/altair/uti
ls/deprecation.py:65: AltairDeprecationWarning:

'selection' is deprecated.
   Use 'selection_point()' or 'selection_interval()' instead; these fun
ctions also include more helpful docstrings.

/Users/roannarague/opt/anaconda3/lib/python3.8/site-packages/altair/veg
alite/v5/api.py:450: AltairDeprecationWarning:

The types 'single' and 'multi' are now
        combined and should be specified using "selection_point()".
```

Out[178]:


Day Of Week
Friday
Monday
Saturday
Sunday
Thursday
Tuesday
Wednesday
latitude
42
41
40
39
38
37
36
longitude
−114.0
−113.0
−112.0
−111.0

In [152]:
```
alt.data_transformers.disable_max_rows()

# Extract hour from timestamp
X_test['hour'] = X_test['final_date_column'].dt.hour

# Create an Altair chart
chart = alt.Chart(X_test).mark_line().encode(
    x='hour:O',
    y='mean(Total):Q',
    tooltip=['hour:O', 'mean(Total):Q']
).properties(
    title='Hour-wise Trend Line',
    width=600
)

# Show the chart
chart
```

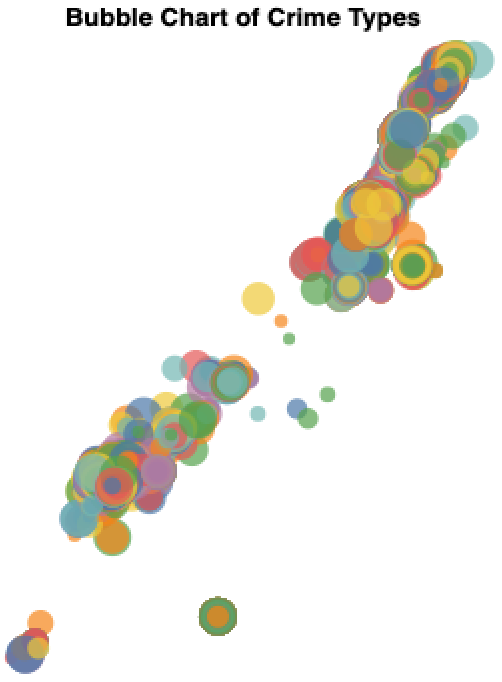
Out[152]:



Bubble Chart plotted on the Map of where the Crime Occured

In [175]:
```
# Create an Altair bubble chart
chart = alt.Chart(X_test).mark_circle().encode(
    longitude='longitude:Q',
    latitude='latitude:Q',
    size='Total:Q',
    color='day_of_week:N',
    tooltip=['Crime_Category:N', 'Total:Q']
).properties(
    title='Bubble Chart of Crime Types',
    width=600
)

# Show the chart
chart
```

Out[175]:



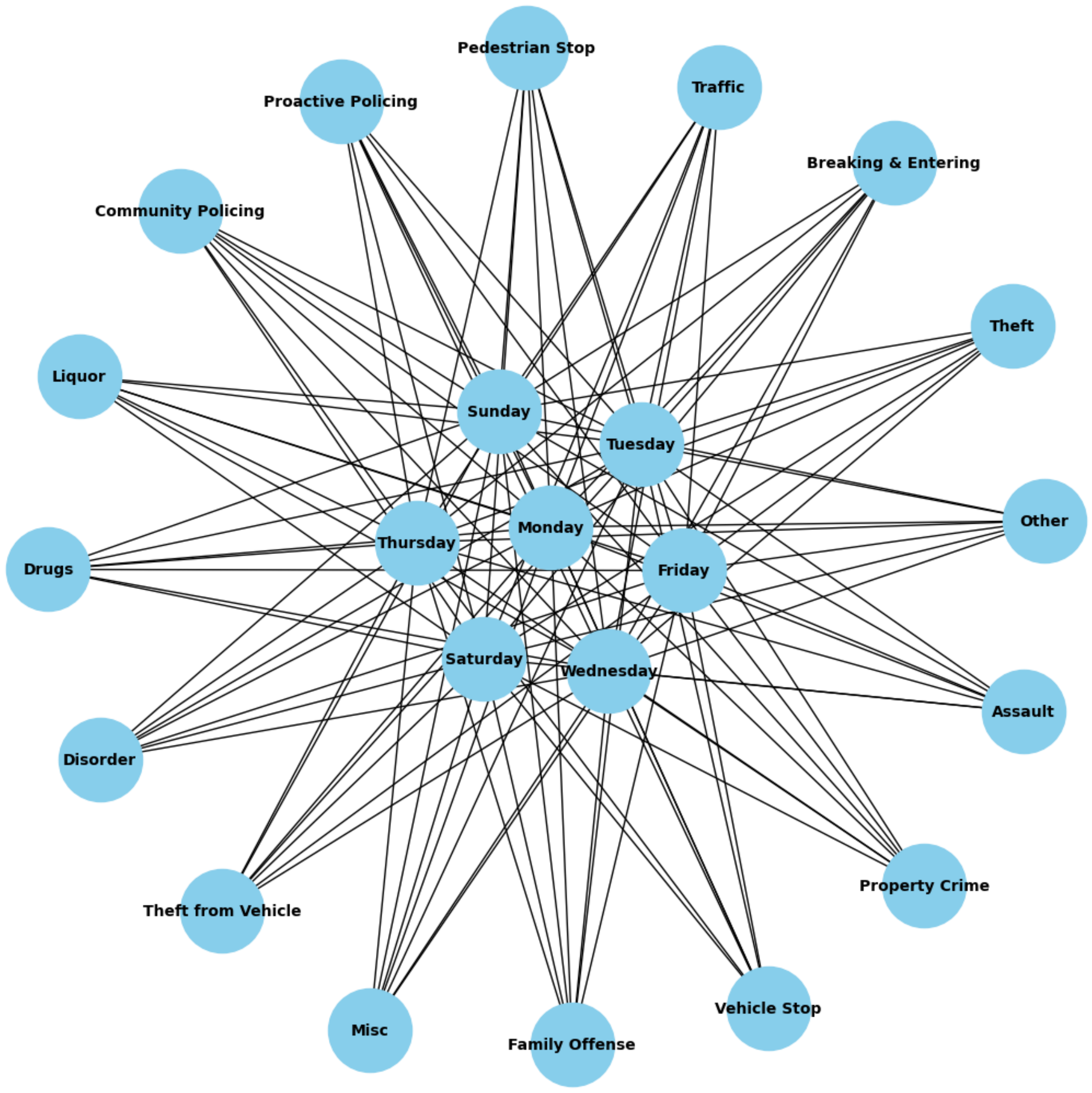
In [174]:
```
import pandas as pd
import plotly.express as px

# Calculate the total count of each crime category
crime_category_counts = X_test['Crime_Category'].value_counts().reset_in
crime_category_counts.columns = ['Crime_Category', 'Total']

# Create a packed bubble chart
fig = px.scatter(crime_category_counts, x='Total', y='Total', size='Tota
                 hover_name='Crime_Category', title='Packed Bubble Chart
                 labels={'Count': 'Total Count'})

# Show the chart
fig.show()
```

NETWORK GRAPH

This graph is to visualize any relation between the type of crime and the day of the week.

In [180]:
```
import networkx as nx
```

In [181]:
```
# Build your graph
G = nx.from_pandas_edgelist(X_test, 'day_of_week', 'Crime_Category')

# Set the figure size
plt.figure(figsize=(10, 10))

# Plot it
nx.draw(G, with_labels=True, font_size=10, node_size=3000, node_color='sl

plt.show()
```



Final Result Plan

The goal is to combine the above graphs to create a filterable dashboard to be able to filter and check to see if there are trends with the data that answer the following questions:

1) Are there crime trends in different cities? 2) Are there crime trends with different years? 3) What different types of crime occurs, and which one is most prelavent per city? 4) Does the day in the week effect the amount of crime and what type of crime occurs? 5) Does the time of day impact where a crime occurs and what type?

Type *Markdown* and LaTeX: α^2

```
In [ ]:
```