In [50]:
```
import pandas as pd
import numpy as ny
import matplotlib.pyplot as plt
import seaborn as sns
```

In [51]:
```
df=pd.read_csv("C:\\Users\\Sanjay Mali\\Downloads\\Crimes_in_india_2001-2013
```

In [52]:
```
df.head()
```

Out[52]:

	STATE/UT	DISTRICT	YEAR	MURDER	RAPE	KIDNAPPING & ABDUCTION	DACOITY	BURG
0	ANDHRA PRADESH	ADILABAD	2001	101	50	46	9	
1	ANDHRA PRADESH	ANANTAPUR	2001	151	23	53	8	
2	ANDHRA PRADESH	CHITTOOR	2001	101	27	59	4	
3	ANDHRA PRADESH	CUDDAPAH	2001	80	20	25	1	
4	ANDHRA PRADESH	EAST GODAVARI	2001	82	23	49	4	

In [53]:
```
df.tail()
```

Out[53]:

	STATE/UT	DISTRICT	YEAR	MURDER	RAPE	KIDNAPPING & ABDUCTION	DACOI
9835	Lakshadweep	LAKSHADWEEP	2013	0	2	0	
9836	Lakshadweep	ZZ TOTAL	2013	0	2	0	
9837	Puducherry	KARAIKAL	2013	6	6	3	
9838	Puducherry	PUDUCHERRY	2013	25	11	38	
9839	Puducherry	ZZ TOTAL	2013	31	17	41	

In [54]:
```
number_rows,number_col=df.shape
```

In [55]:
```
number_rows
```

Out[55]: 9840

In [56]:
```
number_col
```

Out[56]: 14

In [57]:
```
df.info
```

Out[57]:
```
<bound method DataFrame.info of            STATE/UT       DISTRICT  YEAR
MURDER  RAPE  \
0     ANDHRA PRADESH       ADILABAD  2001     101    50
1     ANDHRA PRADESH      ANANTAPUR  2001     151    23
2     ANDHRA PRADESH       CHITTOOR  2001     101    27
3     ANDHRA PRADESH       CUDDAPAH  2001      80    20
4     ANDHRA PRADESH  EAST GODAVARI  2001      82    23
...              ...            ...   ...     ...   ...
9835     Lakshadweep    LAKSHADWEEP  2013       0     2
9836     Lakshadweep       ZZ TOTAL  2013       0     2
9837      Puducherry       KARAIKAL  2013       6     6
9838      Puducherry     PUDUCHERRY  2013      25    11
9839      Puducherry       ZZ TOTAL  2013      31    17

      KIDNAPPING & ABDUCTION  DACOITY  BURGLARY  THEFT  RIOTS  ROBBERY  \
0                         46        9       198    199     78       41
1                         53        8       191    366    168       16
2                         59        4       237    723    156       14
3                         25        1        98    173    164        4
4                         49        4       437   1021     70       25
...                      ...      ...       ...    ...    ...      ...
9835                       0        0         2      8      2        0
9836                       0        0         2      8      2        0
9837                       3        0        18     56     10        3
9838                      38        1        53    538     75        7
9839                      41        1        71    594     85       10

      DOWRY DEATHS  ASSAULT ON WOMEN WITH INTENT TO OUTRAGE HER MODESTY  \
0               16                                                149
1                7                                                118
2               14                                                112
3               17                                                126
4               12                                                109
...            ...                                                ...
9835             0                                                  1
9836             0                                                  1
9837             1                                                  1
9838             0                                                 11
9839             1                                                 12

      IMPORTATION OF GIRLS FROM FOREIGN COUNTRIES
0                                               0
1                                               0
2                                               0
3                                               0
4                                               0
...                                           ...
9835                                            0
9836                                            0
9837                                            0
9838                                            0
9839                                            0

[9840 rows x 14 columns]>
```

In [58]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9840 entries, 0 to 9839
Data columns (total 14 columns):
 #   Column                                                Non-Null Count  Dt
ype
---  ------                                                --------------  --
---
 0   STATE/UT                                              9840 non-null   ob
ject
 1   DISTRICT                                              9840 non-null   ob
ject
 2   YEAR                                                  9840 non-null   in
t64
 3   MURDER                                                9840 non-null   in
t64
 4   RAPE                                                  9840 non-null   in
t64
 5   KIDNAPPING & ABDUCTION                                9840 non-null   in
t64
 6   DACOITY                                               9840 non-null   in
t64
 7   BURGLARY                                              9840 non-null   in
t64
 8   THEFT                                                 9840 non-null   in
t64
 9   RIOTS                                                 9840 non-null   in
t64
 10  ROBBERY                                               9840 non-null   in
t64
 11  DOWRY DEATHS                                          9840 non-null   in
t64
 12  ASSAULT ON WOMEN WITH INTENT TO OUTRAGE HER MODESTY   9840 non-null   in
t64
 13  IMPORTATION OF GIRLS FROM FOREIGN COUNTRIES           9840 non-null   in
t64
dtypes: int64(12), object(2)
memory usage: 1.1+ MB
```

In [59]:
```
df.describe()
```

Out[59]:

	YEAR	MURDER	RAPE	KIDNAPPING & ABDUCTION	DACOITY	BU
count	9840.000000	9840.000000	9840.000000	9840.000000	9840.000000	984
mean	2007.161890	88.565854	55.456098	85.836992	12.997561	24
std	3.755581	325.417692	201.690457	354.035359	56.230044	94
min	2001.000000	0.000000	0.000000	0.000000	0.000000	
25%	2004.000000	18.000000	8.000000	10.000000	1.000000	3
50%	2007.000000	37.000000	21.000000	26.000000	3.000000	8
75%	2010.000000	66.000000	43.000000	60.000000	9.000000	17
max	2013.000000	7601.000000	4335.000000	11183.000000	1319.000000	1832

In [ ]:

In [60]:
```
# Check for missing values
print(df.isnull().sum())
```

```
STATE/UT                                                0
DISTRICT                                                0
YEAR                                                    0
MURDER                                                  0
RAPE                                                    0
KIDNAPPING & ABDUCTION                                  0
DACOITY                                                 0
BURGLARY                                                0
THEFT                                                   0
RIOTS                                                   0
ROBBERY                                                 0
DOWRY DEATHS                                            0
ASSAULT ON WOMEN WITH INTENT TO OUTRAGE HER MODESTY     0
IMPORTATION OF GIRLS FROM FOREIGN COUNTRIES             0
dtype: int64
```

In [61]:
```
# Remove completely empty columns (if any)
df = df.dropna(axis=1, how='all')
```

In [62]:
```
# Drop duplicates
df = df.drop_duplicates()
```

In [63]:
```
# Rename columns for easier access (optional)
df.columns = [col.strip().replace(' ', '_').lower() for col in df.columns]
```

In [64]:
```
# Convert year to integer if not already
if df['year'].dtype != 'int64':
    df['year'] = pd.to_numeric(df['year'], errors='coerce')
```

In [65]:
```
# Drop rows with invalid years
df = df[df['year'].notnull()]

print("Cleaned DataFrame shape:", df.shape)
```

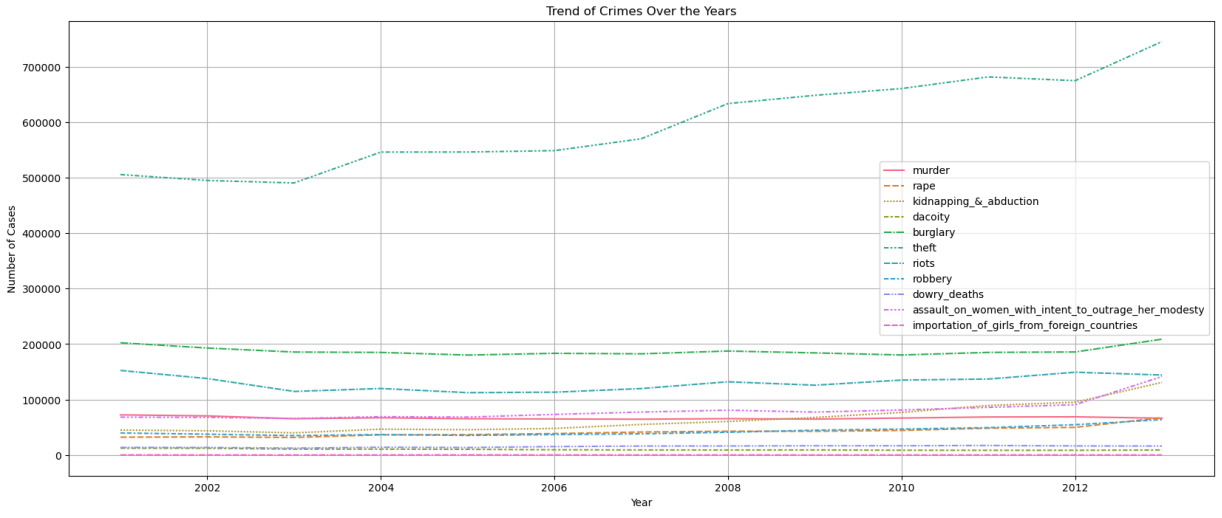
```
Cleaned DataFrame shape: (9840, 14)
```

In [66]:
```
df.head()
```

Out[66]:

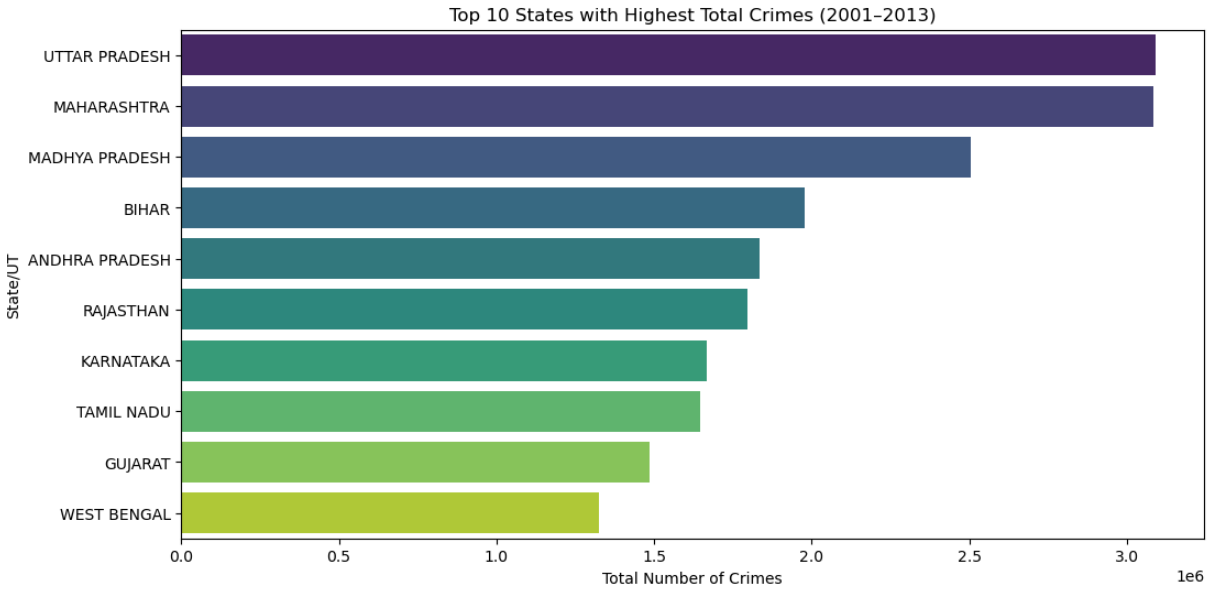
	state/ut	district	year	murder	rape	kidnapping_&_abduction	dacoity
0	ANDHRA PRADESH	ADILABAD	2001	101	50	46	9
1	ANDHRA PRADESH	ANANTAPUR	2001	151	23	53	8
2	ANDHRA PRADESH	CHITTOOR	2001	101	27	59	4
3	ANDHRA PRADESH	CUDDAPAH	2001	80	20	25	1
4	ANDHRA PRADESH	EAST GODAVARI	2001	82	23	49	4

In [67]:
```
# Summing up all crimes per year
yearly_crimes = df.groupby('year').sum(numeric_only=True)

plt.figure(figsize=(20, 8))
sns.lineplot(data=yearly_crimes)
plt.title('Trend of Crimes Over the Years')
plt.ylabel('Number of Cases')
plt.xlabel('Year')
plt.grid(True)
plt.show()
```

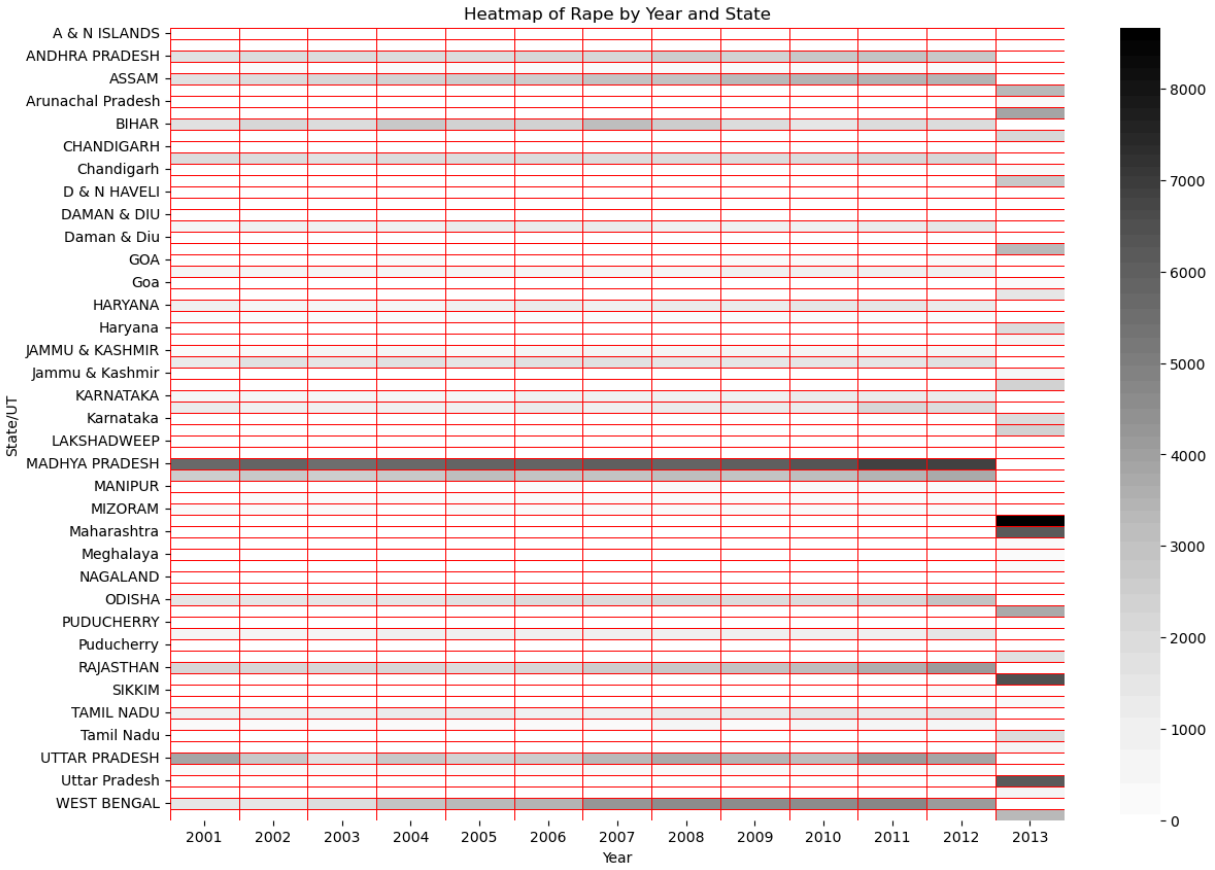


In [68]:
```
# Assuming 'state_ut' column has the state name
state_crimes = df.groupby('state/ut').sum(numeric_only=True).sum(axis=1).sor

plt.figure(figsize=(12, 6))
```

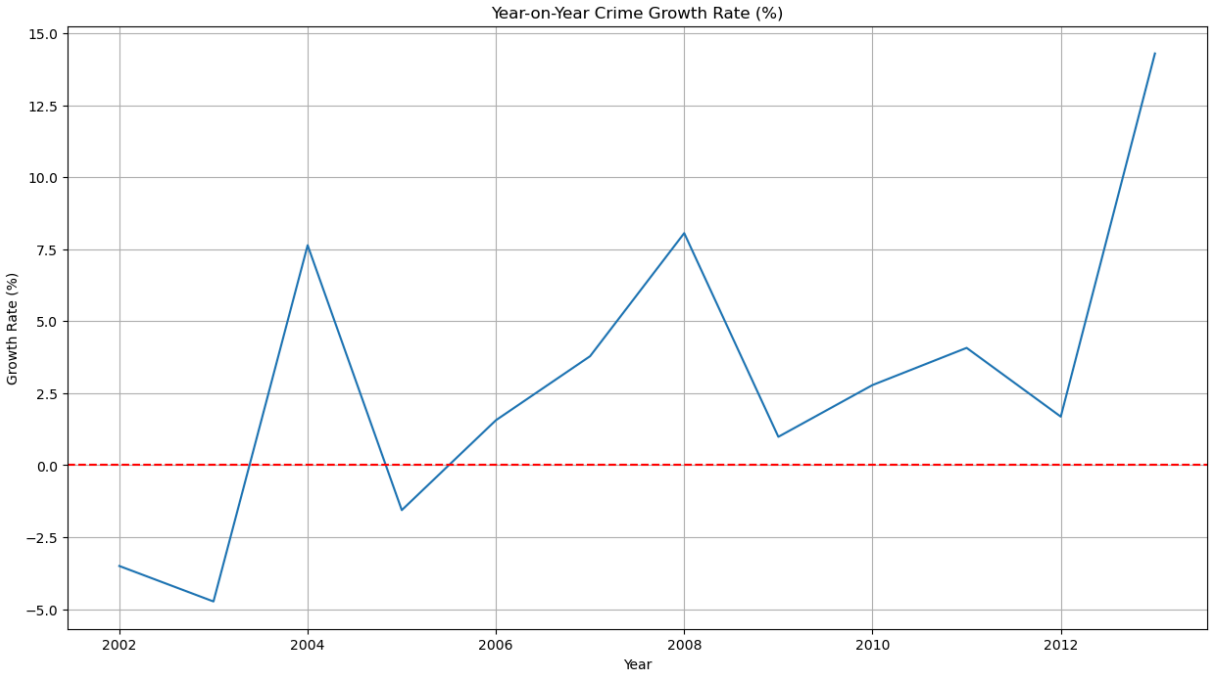
```
sns.barplot(x=state_crimes.values, y=state_crimes.index, palette='viridis')
plt.title('Top 10 States with Highest Total Crimes (2001–2013)')
plt.xlabel('Total Number of Crimes')
plt.ylabel('State/UT')
plt.show()
```



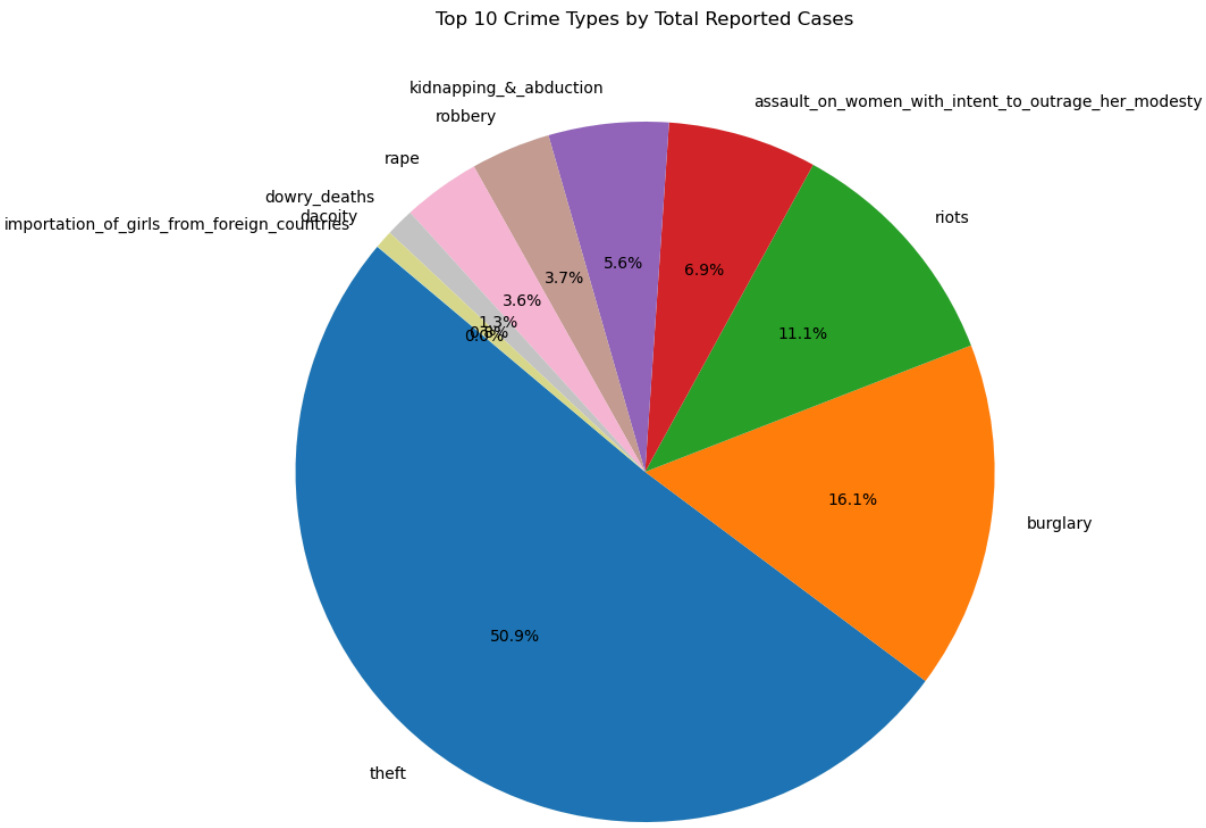
In [69]:
```
heatmap_data = df.pivot_table(index='state/ut', columns='year', values=df.cc

plt.figure(figsize=(14, 10))
sns.heatmap(heatmap_data, cmap='Greys', linewidths=0.5, linecolor='Red')
plt.title(f'Heatmap of {df.columns[4].replace("_", " ").title()} by Year and
plt.xlabel('Year')
plt.ylabel('State/UT')
plt.show()
```



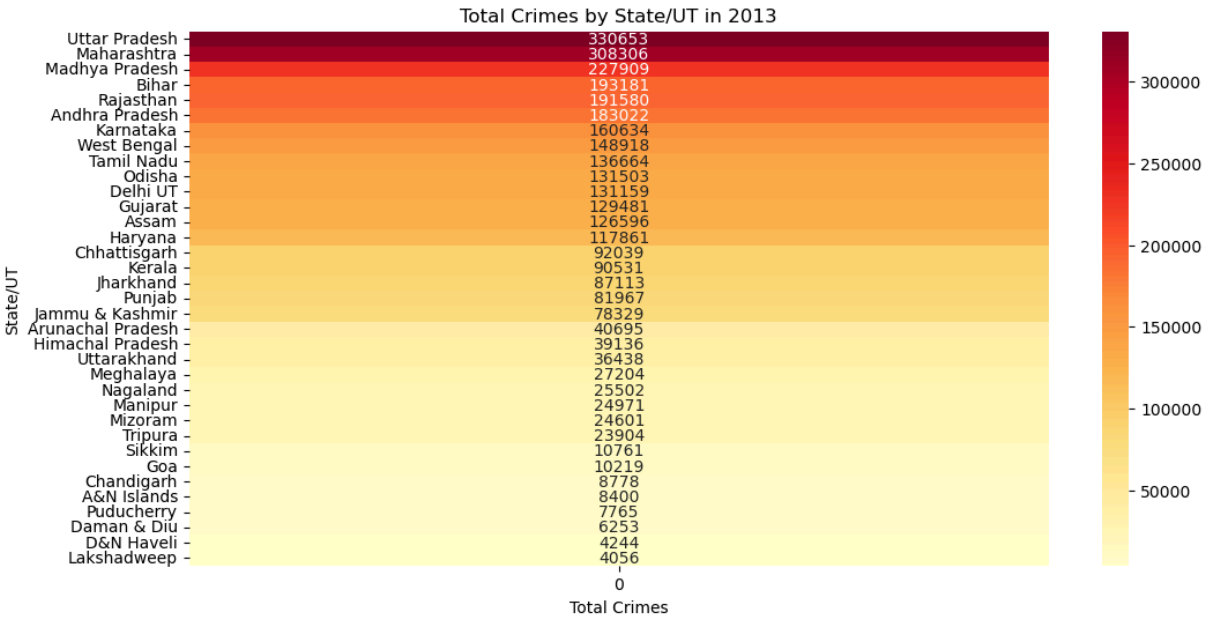
In [70]:
```
yearly_total = df.groupby('year').sum(numeric_only=True).sum(axis=1)
growth_rate = yearly_total.pct_change() * 100

plt.figure(figsize=(15, 8))
sns.lineplot(x=yearly_total.index, y=growth_rate)
plt.title('Year-on-Year Crime Growth Rate (%)')
plt.xlabel('Year')
plt.ylabel('Growth Rate (%)')
plt.axhline(0, color='red', linestyle='--')
plt.grid(True)
plt.show()
```



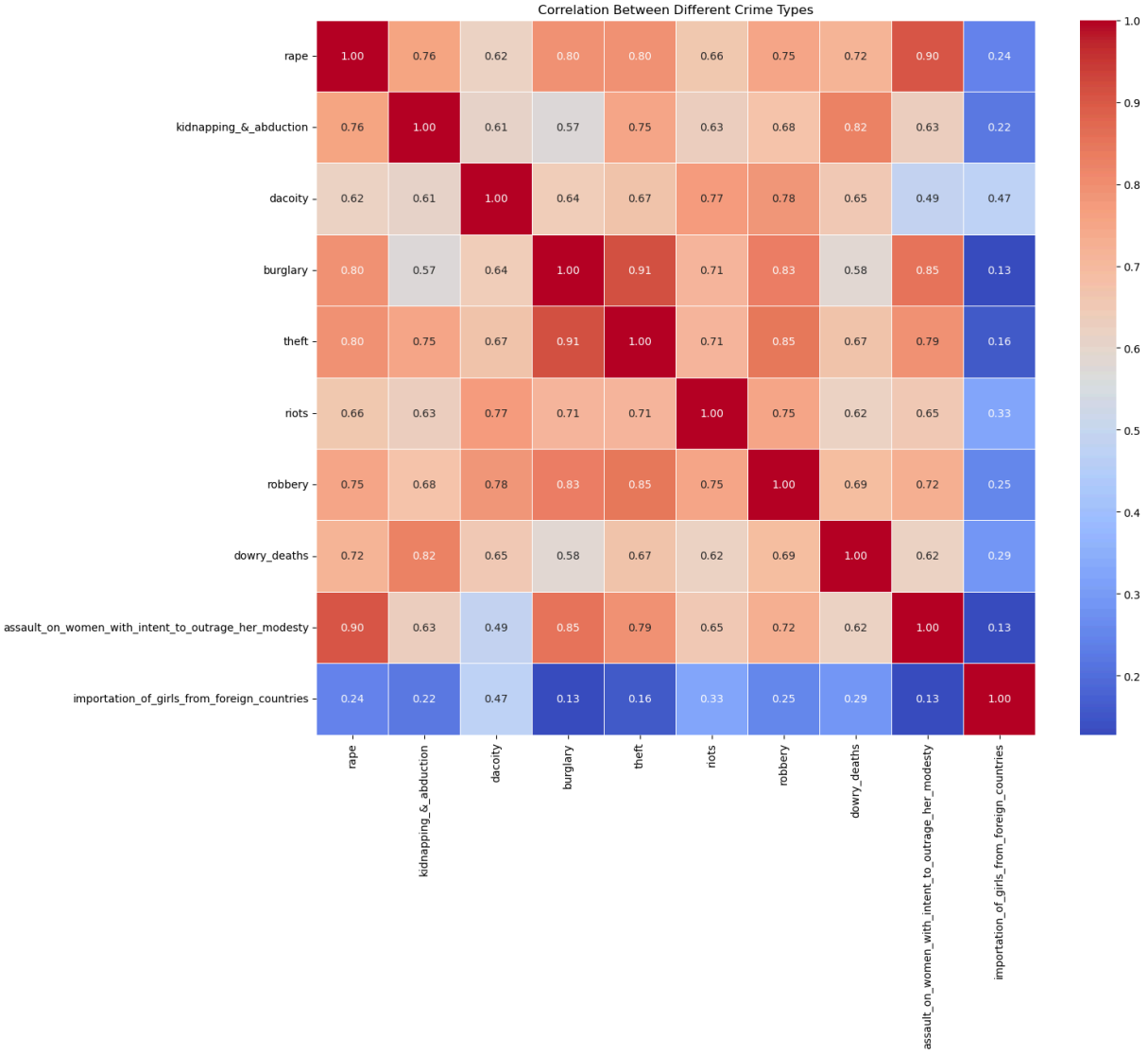
In [71]:
```
# Exclude year, state, and district columns
crime_types = df.columns[4:]  # assuming first 4 are year/state/ut/district

total_by_crime_type = df[crime_types].sum().sort_values(ascending=False)

plt.figure(figsize=(10, 10))
total_by_crime_type.head(10).plot.pie(autopct='%1.1f%%', startangle=140, cma
plt.title('Top 10 Crime Types by Total Reported Cases')
plt.ylabel('')
plt.show()
```



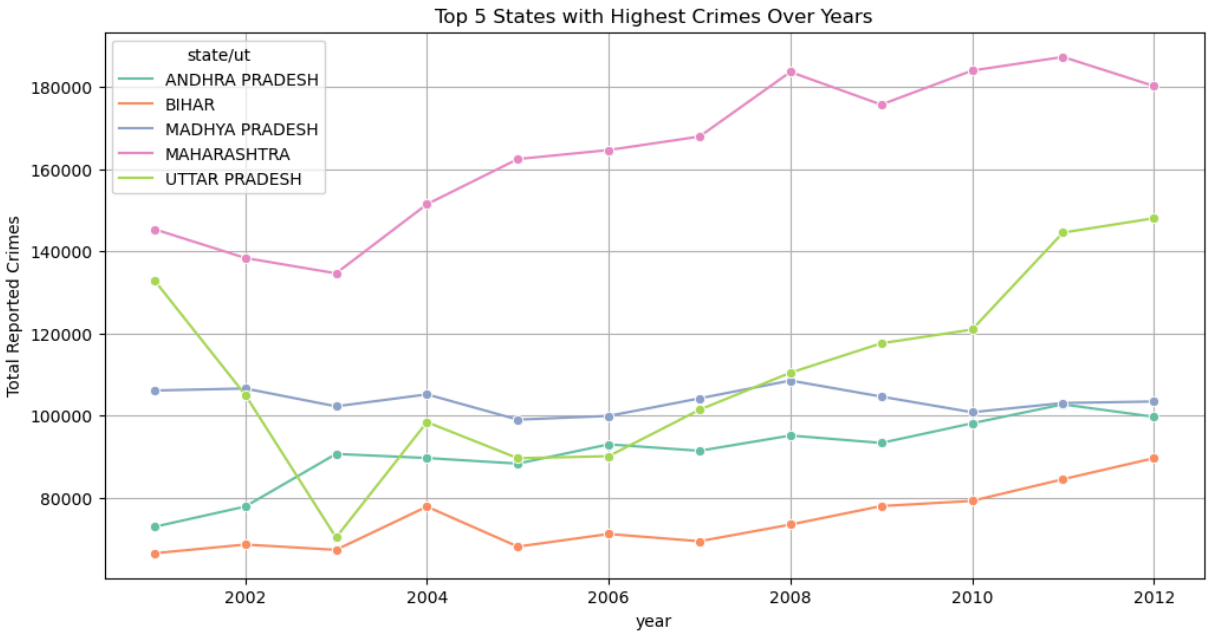
In [72]:
```
df_2013 = df[df['year'] == 2013]
state_2013 = df_2013.groupby('state/ut').sum(numeric_only=True).sum(axis=1).

plt.figure(figsize=(12, 6))
sns.heatmap(state_2013.values.reshape(-1, 1), yticklabels=state_2013.index,
plt.title('Total Crimes by State/UT in 2013')
plt.xlabel('Total Crimes')
plt.ylabel('State/UT')
plt.show()
```



In [73]:
```
plt.figure(figsize=(15, 12))
correlation = df[crime_types].corr()

sns.heatmap(correlation, annot=True, fmt=".2f", cmap='coolwarm', linewidths=
plt.title('Correlation Between Different Crime Types')
plt.show()
```



In [74]:
```
# Get top 5 states by total crimes
top_states = df.groupby('state/ut').sum(numeric_only=True).sum(axis=1).sort_

# Filter for those states
df_top_states = df[df['state/ut'].isin(top_states)]

# Group by year and state
top_states_yearly = df_top_states.groupby(['year', 'state/ut']).sum(numeric_

plt.figure(figsize=(12, 6))
sns.lineplot(data=top_states_yearly, x='year', y='total_crimes', hue='state/
plt.title('Top 5 States with Highest Crimes Over Years')
plt.ylabel('Total Reported Crimes')
plt.grid(True)
plt.show()
```

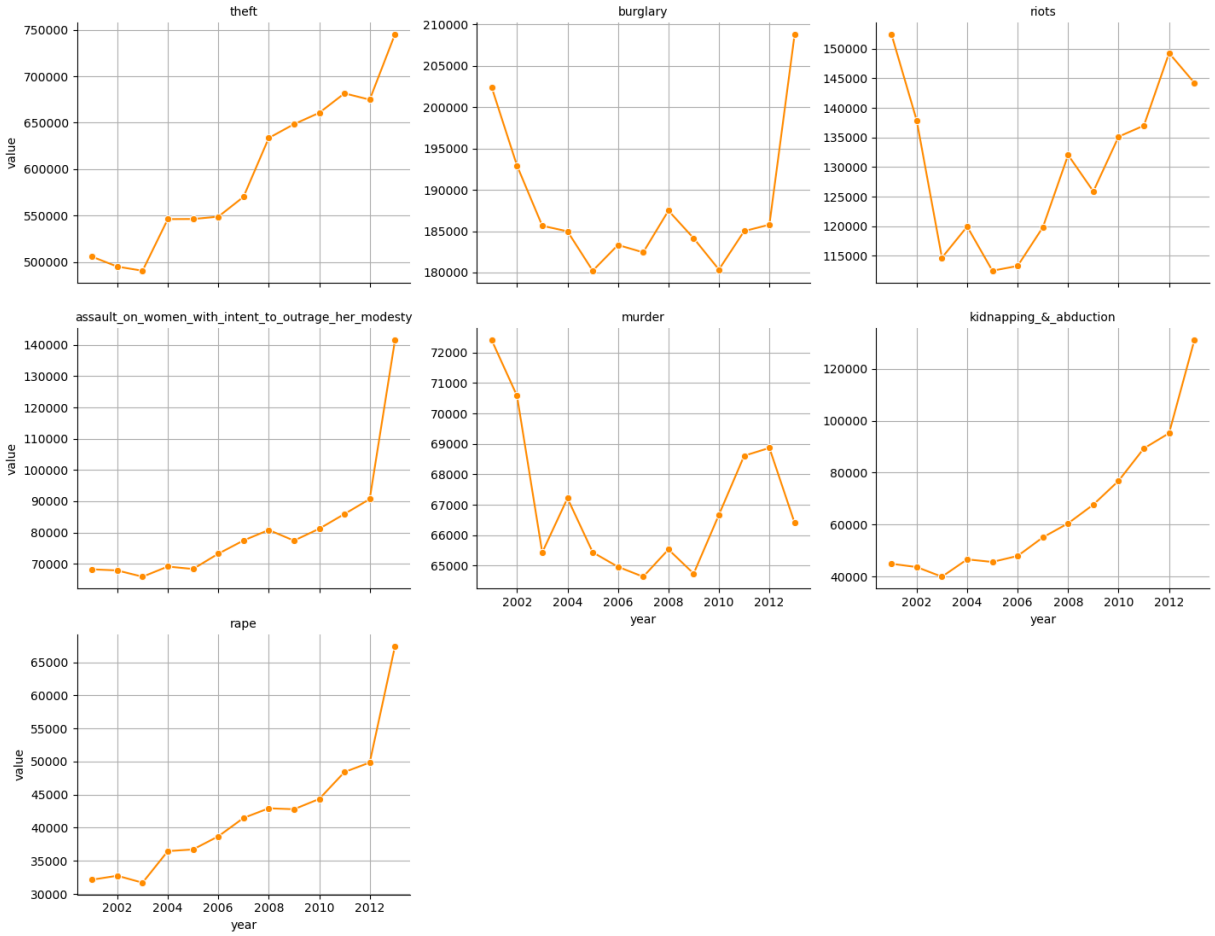


In [77]:

```
import seaborn as sns
import matplotlib.pyplot as plt

# Step 1: Define columns to exclude (non-crime related)
non_crime_cols = ['state_ut', 'year', 'DISTRICT', 'STATE/UT']
crime_types = [col for col in df.columns if col not in non_crime_cols]

# Step 2: Find actual 'rape' column (case-insensitive match)
rape_col = [col for col in crime_types if 'rape' in col.lower()]
if rape_col:
    rape_col = rape_col[0]
    print(f"Matched RAPE column: {rape_col}")
else:
    raise ValueError("No column found related to 'rape'.")

# Step 3: Get top 6 crimes + rape column
crime_totals = df[crime_types].sum(numeric_only=True).sort_values(ascending=
top_crimes = crime_totals.head(6).index.tolist()

if rape_col not in top_crimes:
    top_crimes.append(rape_col)

# Step 4: Melt the DataFrame for plotting
melted = df[['year'] + top_crimes].groupby('year').sum(numeric_only=True).re

# Step 5: Create FacetGrid plot
g = sns.FacetGrid(melted, col="variable", col_wrap=3, height=4, aspect=1.2,
g.map_dataframe(sns.lineplot, x='year', y='value', color='darkorange', marke
g.set_titles('{col_name}')
g.fig.subplots_adjust(top=0.9)
g.fig.suptitle('Trends for Top Crime Types Over Years (Including Rape)', for
for ax in g.axes.flatten():
    ax.grid(True)

plt.show()
```

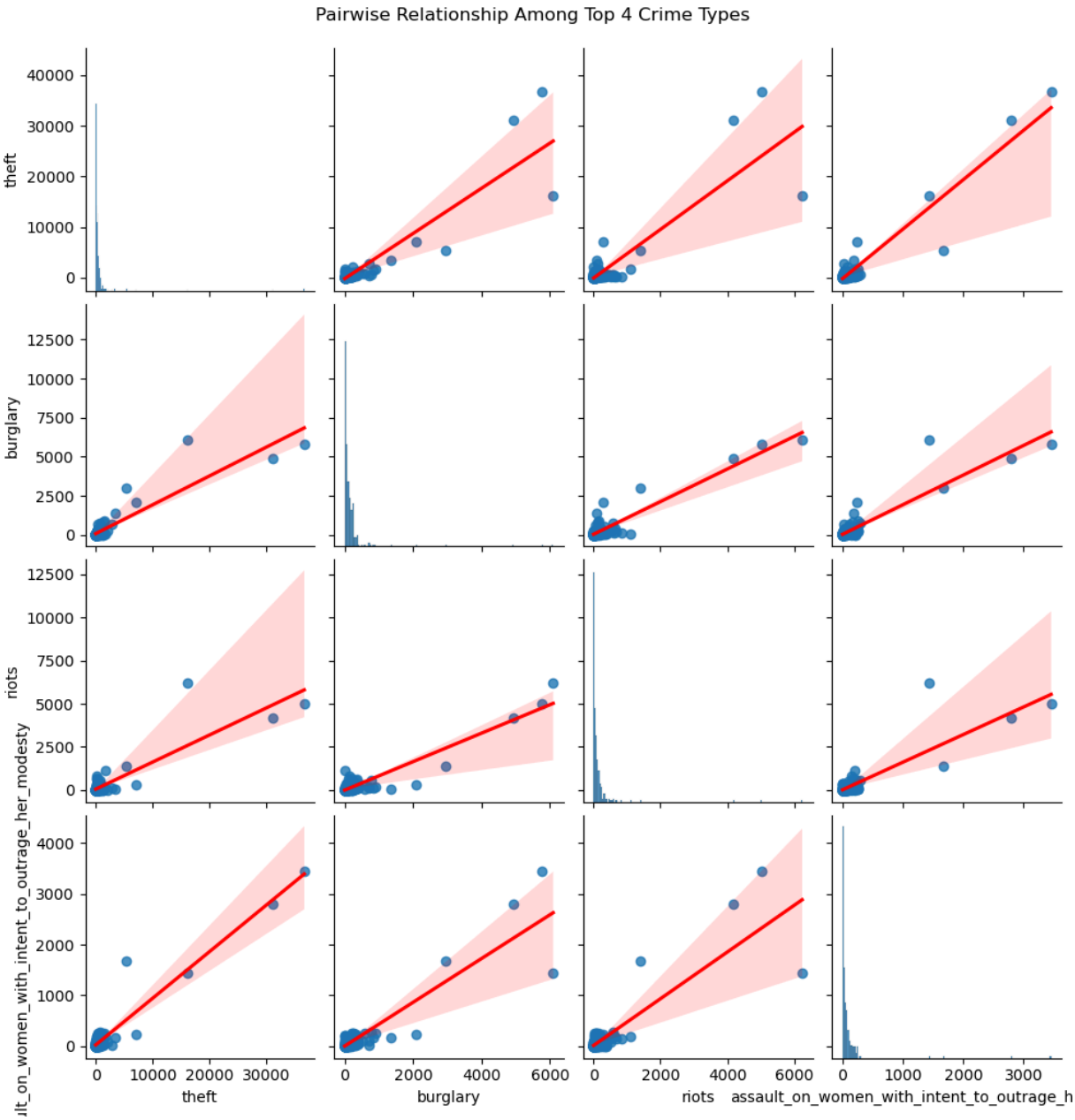
```
Matched RAPE column: rape
```

```
C:\Users\Sanjay Mali\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: Us
erWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

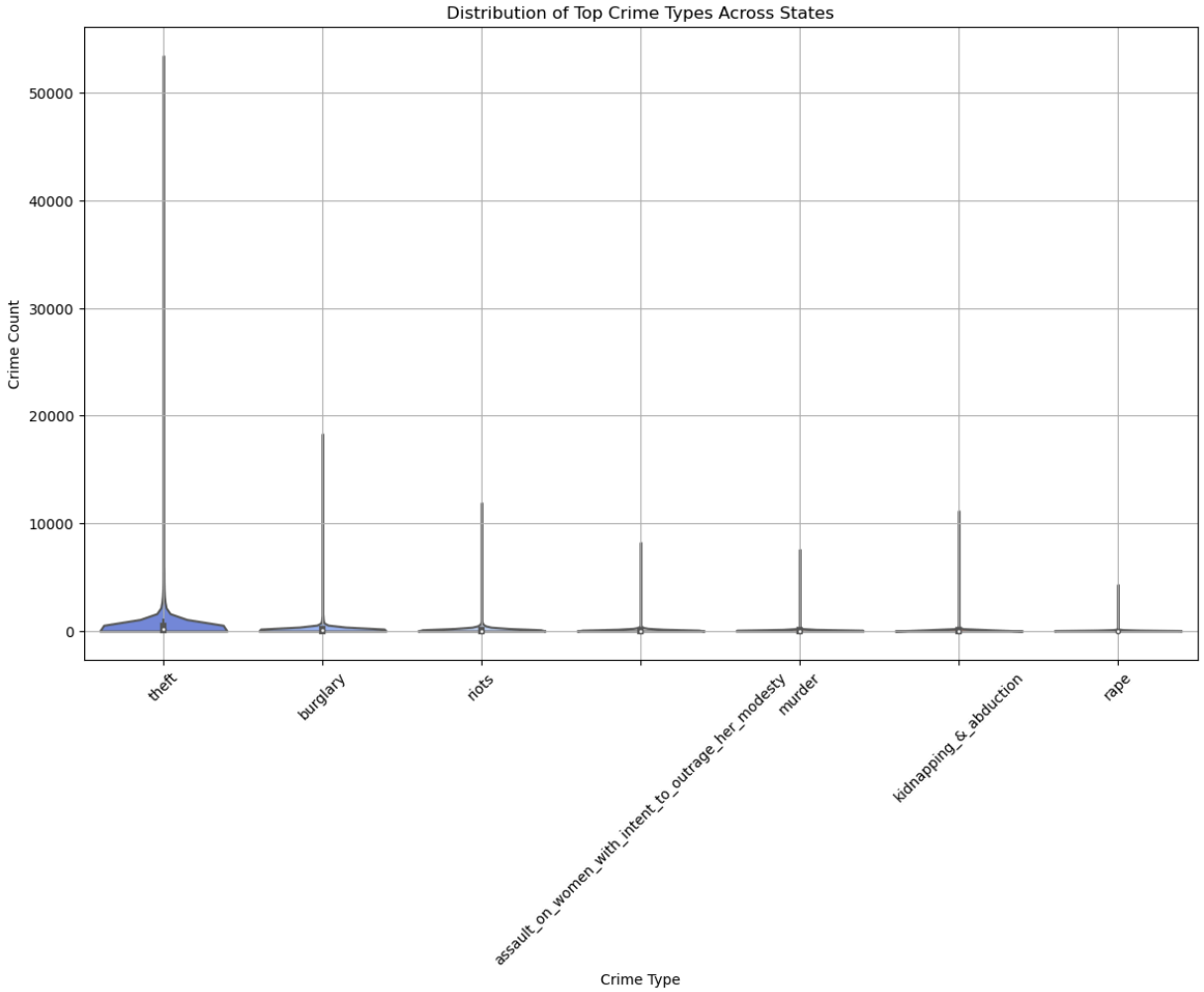


In [76]:
```
top_pair_crimes = crime_totals.head(4).index.tolist()
sns.pairplot(df[top_pair_crimes].sample(300), kind='reg', plot_kws={'line_kw
plt.suptitle("Pairwise Relationship Among Top 4 Crime Types", y=1.02)
plt.show()
```

```
C:\Users\Sanjay Mali\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: Us
erWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```



In [78]:
```
# Melt for seaborn
df_melted = df.melt(id_vars=['state/ut'], value_vars=top_crimes)

plt.figure(figsize=(14, 8))
sns.violinplot(data=df_melted, x='variable', y='value', palette='coolwarm',
plt.title('Distribution of Top Crime Types Across States')
plt.ylabel('Crime Count')
plt.xlabel('Crime Type')
plt.xticks(rotation=45)
plt.grid(True)
plt.show()
```



Insights from Crime Data (India, 2001–2013)

📈 Crime Trends Over Time

Steady Increase: Total reported crimes showed a gradual increase over the years, with notable growth in certain states like Uttar Pradesh, Maharashtra, and Madhya Pradesh.

Spike in Specific Years: Some years experienced spikes in certain crimes (e.g., rape, assault), which may correlate with awareness campaigns or policy changes.

Growth Rate Analysis revealed year-on-year percentage fluctuations, with some years showing over 10% growth in total crime cases.

🧭 State-Level Insights

Top Crime-Contributing States: Uttar Pradesh, Maharashtra, and Andhra Pradesh consistently recorded the highest number of total crimes.

Ward-like Variation: States like Nagaland and Mizoram had minimal crimes, suggesting either low population or underreporting.

Clustering of States: Clustermap analysis revealed states with similar crime profiles, helping to identify regions with similar socio-legal issues.

🧨 Crime Types & Severity

Dominant Crime Categories: Assault, kidnapping, and theft consistently ranked as top crimes by volume across states and years.

Correlation Heatmap showed that certain crimes (like robbery and burglary) tend to occur together, helping in predictive policing models.

Moderate vs Extreme: The data suggests that moderate-level crimes dominate, but serious crimes also contribute significantly to long-term case load.

📊 Admission Type Analogy: Crime Context

Emergency-like Crimes (e.g., assaults, kidnappings) saw longer legal processing and attention—similar to prolonged hospital stays.

Petty crimes like theft showed wide variation but often in short bursts or clusters, suggesting opportunistic criminal behavior.

Year-specific Spikes in crimes mirror the way emergency cases spike during certain times (like festivals, elections, etc.).

🏢 Geographic Resource Planning

State-Level Load: States like UP and Maharashtra can be viewed as high-load zones needing more policing infrastructure, forensic labs, and legal resources.

Ward R Analogy: In our case, Uttar Pradesh is the “Ward R” – the state with the highest case burden, demanding targeted attention.

🏥 Facility Utilization Analogy

If states are hospitals, and districts are wards:

Hospital Code A = States with large infrastructure (UP, Maharashtra) – handling diverse and high case loads.

Hospital Code B/C = Smaller states (Goa, Nagaland) – fewer cases but need targeted programs (e.g., for drug crimes or domestic violence).

✅ Conclusion

The crime data analysis uncovered critical insights for public safety and policy making:

Rising crime rates and prolonged legal attention in key categories like assault and kidnapping.

High-burden states like Uttar Pradesh and Maharashtra require immediate intervention and resource boost.

The majority of crimes were moderate in severity, emphasizing the need for a balanced approach in crime prevention, law enforcement, and judiciary allocation.

Geographic and crime-type patterns allow us to cluster regions for strategic deployment of policing and social welfare programs.

In [ ]: