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

In [ ]:
```

```

In [ ]:
```

```

In [ ]:
```

```

In [2]:
```
df = pd.read_csv('/home/zec/Downloads/archive1/onlinefraud.csv')
```

In [3]:
```
df.head()
```

Out[3]:

	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	old
0	1	PAYMENT	9839.64	C1231006815	170136.0	160296.36	M1979787155	
1	1	PAYMENT	1864.28	C1666544295	21249.0	19384.72	M2044282225	
2	1	TRANSFER	181.00	C1305486145	181.0	0.00	C553264065	
3	1	CASH_OUT	181.00	C840083671	181.0	0.00	C38997010	
4	1	PAYMENT	11668.14	C2048537720	41554.0	29885.86	M1230701703	

In [4]:
```
df.tail()
```

Out[4]:

	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameD
6362615	743	CASH_OUT	339682.13	C786484425	339682.13	0.0	C776919
6362616	743	TRANSFER	6311409.28	C1529008245	6311409.28	0.0	C1881841
6362617	743	CASH_OUT	6311409.28	C1162922333	6311409.28	0.0	C1365125
6362618	743	TRANSFER	850002.52	C1685995037	850002.52	0.0	C2080388
6362619	743	CASH_OUT	850002.52	C1280323807	850002.52	0.0	C873221

In [6]:
```
df.shape
```

Out[6]:
```
(6362620, 11)
```

In [7]:
```
df.columns
```

Out[7]:
```
Index(['step', 'type', 'amount', 'nameOrig', 'oldbalanceOrg', 'newbalanceOr
ig',
       'nameDest', 'oldbalanceDest', 'newbalanceDest', 'isFraud',
       'isFlaggedFraud'],
      dtype='object')
```

In [9]:
```
df.duplicated().sum()
```

Out[9]:
```
0
```

In [10]:
```
df.isnull().sum()
```

Out[10]:
```
step              0
type              0
amount            0
nameOrig          0
oldbalanceOrg     0
newbalanceOrig    0
nameDest          0
oldbalanceDest    0
newbalanceDest    0
isFraud           0
isFlaggedFraud    0
dtype: int64
```

In [11]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6362620 entries, 0 to 6362619
Data columns (total 11 columns):
 #   Column          Dtype
---  ------          -----
 0   step            int64
 1   type            object
 2   amount          float64
 3   nameOrig        object
 4   oldbalanceOrg   float64
 5   newbalanceOrig  float64
 6   nameDest        object
 7   oldbalanceDest  float64
 8   newbalanceDest  float64
 9   isFraud         int64
 10  isFlaggedFraud  int64
dtypes: float64(5), int64(3), object(3)
memory usage: 534.0+ MB
```

In [12]:
```
df.describe()
```

Out[12]:

	step	amount	oldbalanceOrg	newbalanceOrig	oldbalanceDest	newbalance
count	6.362620e+06	6.362620e+06	6.362620e+06	6.362620e+06	6.362620e+06	6.362620
mean	2.433972e+02	1.798619e+05	8.338831e+05	8.551137e+05	1.100702e+06	1.224996
std	1.423320e+02	6.038582e+05	2.888243e+06	2.924049e+06	3.399180e+06	3.674129
min	1.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000
25%	1.560000e+02	1.338957e+04	0.000000e+00	0.000000e+00	0.000000e+00	0.000000
50%	2.390000e+02	7.487194e+04	1.420800e+04	0.000000e+00	1.327057e+05	2.146614
75%	3.350000e+02	2.087215e+05	1.073152e+05	1.442584e+05	9.430367e+05	1.111909
max	7.430000e+02	9.244552e+07	5.958504e+07	4.958504e+07	3.560159e+08	3.561793

In [13]:
```
df.nunique()
```

Out[13]:
```
step                 743
type                   5
amount           5316900
nameOrig         6353307
oldbalanceOrg    1845844
newbalanceOrig   2682586
nameDest         2722362
oldbalanceDest   3614697
newbalanceDest   3555499
isFraud                2
isFlaggedFraud         2
dtype: int64
```

In [17]:
```
object_columns=df.select_dtypes(include=['object']).columns
print(object_columns)
print("object type columns:")
```

```
Index(['type', 'nameOrig', 'nameDest'], dtype='object')
object type columns:
```

In [27]:
```
numerical_columns=df.select_dtypes(include=['int64','float64']).columns
print("\nNumerical type columns:")
print('\nnumerical_columns')
```

```
Numerical type columns:

numerical_columns
```

In [29]:
```
import matplotlib.pyplot as plt
import seaborn as sbn
```

In [31]:
```
import numpy as np
```

In [33]:
```
import warnings
warnings.filterwarnings('ignore')
```

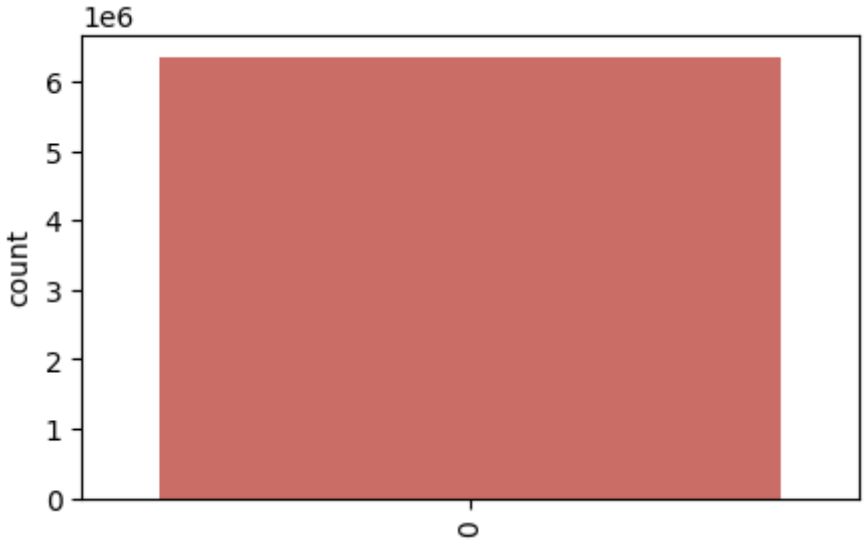
In [34]:
```
df['step'].unique()
```

Out[34]:
```
array([  1,   2,   3,   4,   5,   6,   7,   8,   9,  10,  11,  12,  13,
        14,  15,  16,  17,  18,  19,  20,  21,  22,  23,  24,  25,  26,
        27,  28,  29,  30,  31,  32,  33,  34,  35,  36,  37,  38,  39,
        40,  41,  42,  43,  44,  45,  46,  47,  48,  49,  50,  51,  52,
        53,  54,  55,  56,  57,  58,  59,  60,  61,  62,  63,  64,  65,
        66,  67,  68,  69,  70,  71,  72,  73,  74,  75,  76,  77,  78,
        79,  80,  81,  82,  83,  84,  85,  86,  87,  88,  89,  90,  91,
        92,  93,  94,  95,  96,  97,  98,  99, 100, 101, 102, 103, 104,
       105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117,
       118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130,
       131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143,
       144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156,
       157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169,
       170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182,
       183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195,
       196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208,
       209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221,
       222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234,
       235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247,
       248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259, 260,
       261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273,
       274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285, 286,
       287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299,
       300, 301, 302, 303, 304, 305, 306, 307, 308, 309, 310, 311, 312,
       313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324, 325,
       326, 327, 328, 329, 330, 331, 332, 333, 334, 335, 336, 337, 338,
       339, 340, 341, 342, 343, 344, 345, 346, 347, 348, 349, 350, 351,
       352, 353, 354, 355, 356, 357, 358, 359, 360, 361, 362, 363, 364,
       365, 366, 367, 368, 369, 370, 371, 372, 373, 374, 375, 376, 377,
       378, 379, 380, 381, 382, 383, 384, 385, 386, 387, 388, 389, 390,
       391, 392, 393, 394, 395, 396, 397, 398, 399, 400, 401, 402, 403,
       404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416,
       417, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428, 429,
       430, 431, 432, 433, 434, 435, 436, 437, 438, 439, 440, 441, 442,
       443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454, 455,
       456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468,
       469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480, 481,
       482, 483, 484, 485, 486, 487, 488, 489, 490, 491, 492, 493, 494,
       495, 496, 497, 498, 499, 500, 501, 502, 503, 504, 505, 506, 507,
       508, 509, 510, 511, 512, 513, 514, 515, 516, 517, 518, 519, 520,
       521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531, 532, 533,
       534, 535, 536, 537, 538, 539, 540, 541, 542, 543, 544, 545, 546,
       547, 548, 549, 550, 551, 552, 553, 554, 555, 556, 557, 558, 559,
       560, 561, 562, 563, 564, 565, 566, 567, 568, 569, 570, 571, 572,
       573, 574, 575, 576, 577, 578, 579, 580, 581, 582, 583, 584, 585,
       586, 587, 588, 589, 590, 591, 592, 593, 594, 595, 596, 597, 598,
       599, 600, 601, 602, 603, 604, 605, 606, 607, 608, 609, 610, 611,
       612, 613, 614, 615, 616, 617, 618, 619, 620, 621, 622, 623, 624,
       625, 626, 627, 628, 629, 630, 631, 632, 633, 634, 635, 636, 637,
       638, 639, 640, 641, 642, 643, 644, 645, 646, 647, 648, 649, 650,
       651, 652, 653, 654, 655, 656, 657, 658, 659, 660, 661, 662, 663,
       664, 665, 666, 667, 668, 669, 670, 671, 672, 673, 674, 675, 676,
       677, 678, 679, 680, 681, 682, 683, 684, 685, 686, 687, 688, 689,
       690, 691, 692, 693, 694, 695, 696, 697, 698, 699, 700, 701, 702,
       703, 704, 705, 706, 707, 708, 709, 710, 711, 712, 713, 714, 715,
       716, 717, 718, 719, 720, 721, 722, 723, 724, 725, 726, 727, 728,
       729, 730, 731, 732, 733, 734, 735, 736, 737, 738, 739, 740, 741,
       742, 743])
```

In [37]:
```
df['step'].value_counts()
```

Out[37]:
```
19     51352
18     49579
187    49083
235    47491
307    46968
       ...
432        4
706        4
693        4
112        2
662        2
Name: step, Length: 743, dtype: int64
```

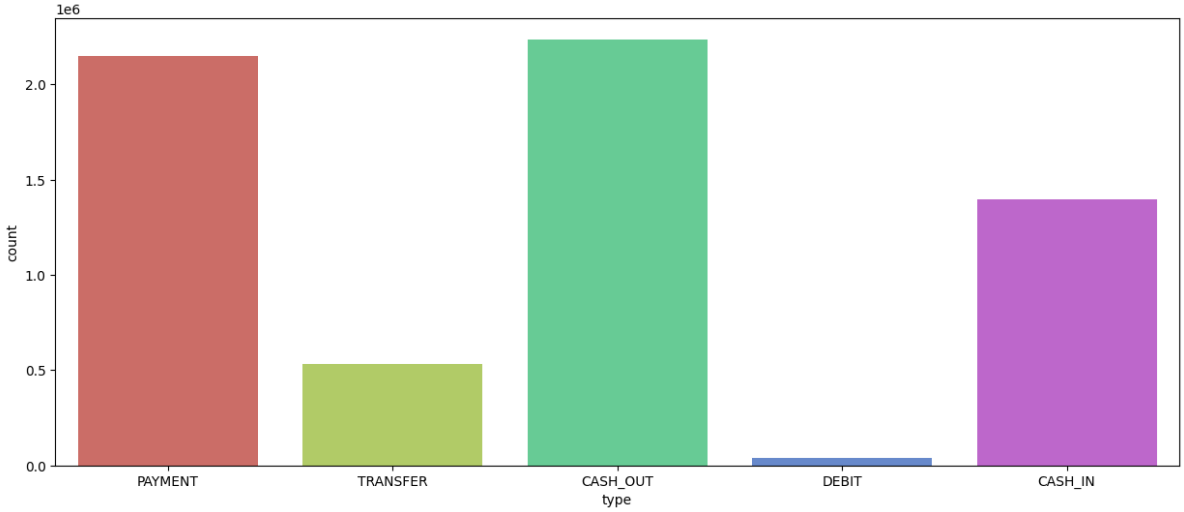
In [48]:
```
import seaborn as sns
plt.figure(figsize=(5,3))
sns.countplot(df['step'], palette= 'hls')
plt.xticks(rotation=90)
plt.show()
```



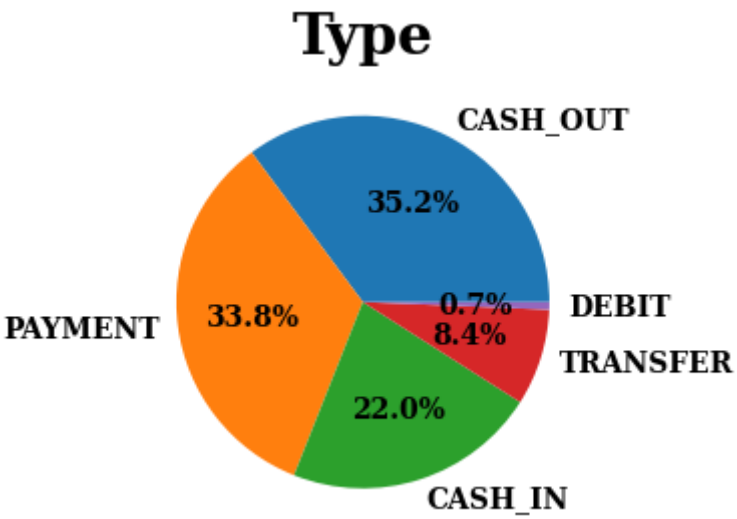
In [49]:
```
df['type'].unique()
```

Out[49]:
```
array(['PAYMENT', 'TRANSFER', 'CASH_OUT', 'DEBIT', 'CASH_IN'],
      dtype=object)
```

In [53]:
```
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(15,6))
sns.countplot(x='type', data=df, palette='hls')
plt.show()
```



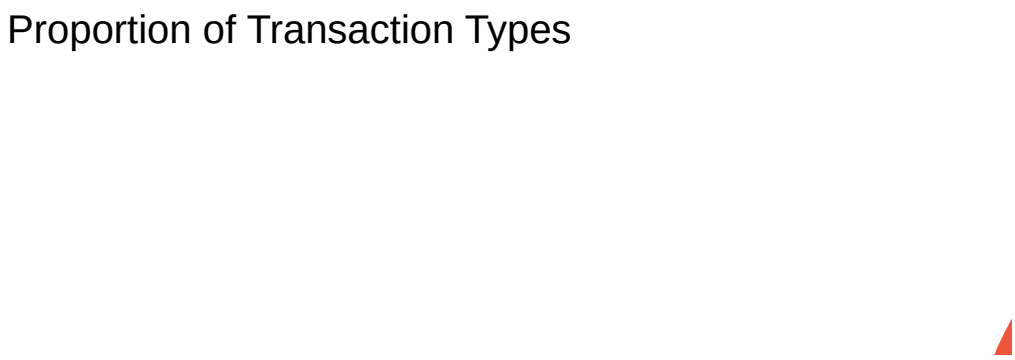
In [57]:
```
import matplotlib.pyplot as plt

plt.figure(figsize=(3, 4))

# Extract values and labels
values = df['type'].value_counts()
labels = values.index

# Define custom font properties
hfont = {'fontname':'serif', 'weight': 'bold'}

# Plot pie chart
plt.pie(values, labels=labels, autopct="%1.1f%%", textprops={'color': 'blac

# Set title with custom font properties
plt.title('Type', size=20, **hfont)

plt.show()
```



In [58]:
```
import plotly.express as px
import plotly.graph_objects as go
```

In [64]:
```
import plotly.graph_objects as go

# Create the Pie chart
pie_chart_type = go.Figure(data=[go.Pie(labels=df['type'].unique(), values=
```

```
# Update layout
pie_chart_type.update_layout(title='Proportion of Transaction Types',
                             font=dict(family="Arial", size=14, color="blac
                             legend_title=dict(font=dict(size=14, family='A
                             )

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



In [65]:
```
df['isFraud'].unique()
```

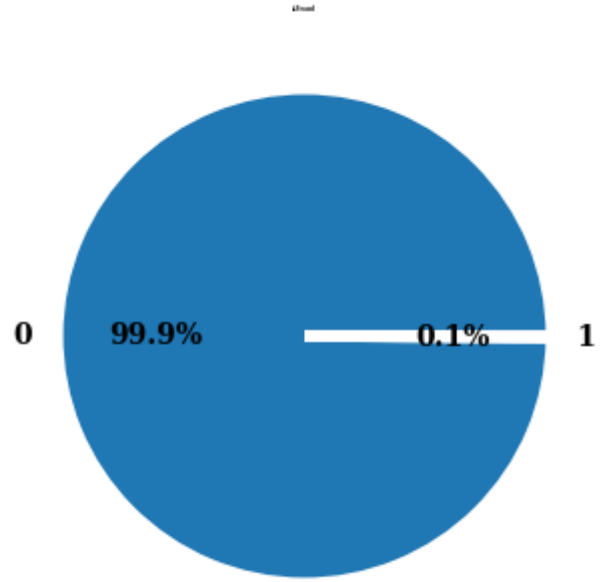
Out[65]:
```
array([0, 1])
```

In [66]:
```
df['isFraud'].value_counts()
```

Out[66]:
```
0    6354407
1       8213
Name: isFraud, dtype: int64
```

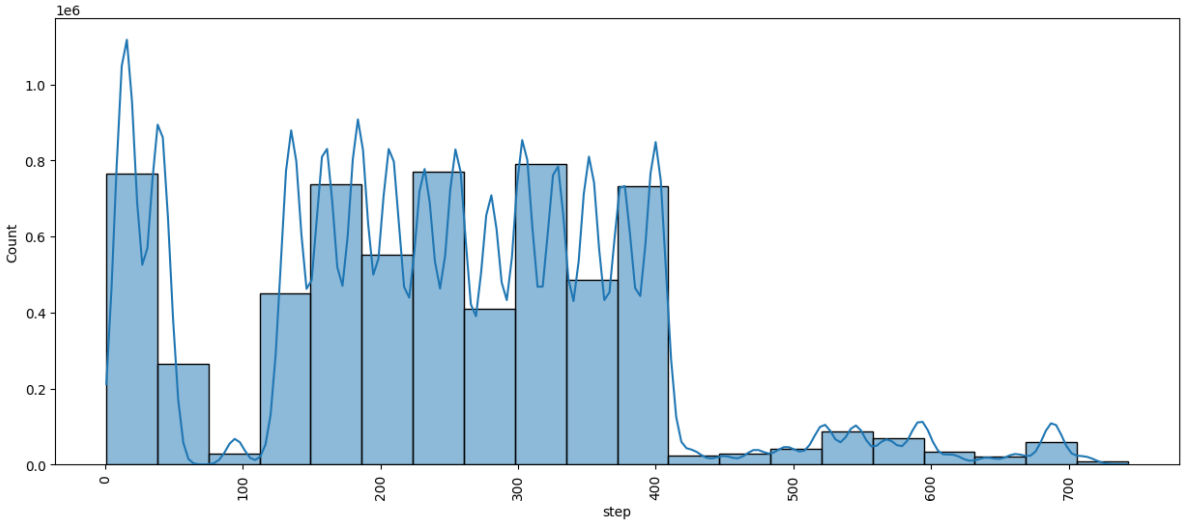
In [71]:
```
import matplotlib.pyplot as plt

# Define custom font properties
hfont = {'fontname':'serif', 'weight': 'bold'}

# Create the Pie chart
plt.figure(figsize=(5, 4))
plt.pie(df['isFraud'].value_counts(),
        labels=df['isFraud'].value_counts().index,
```

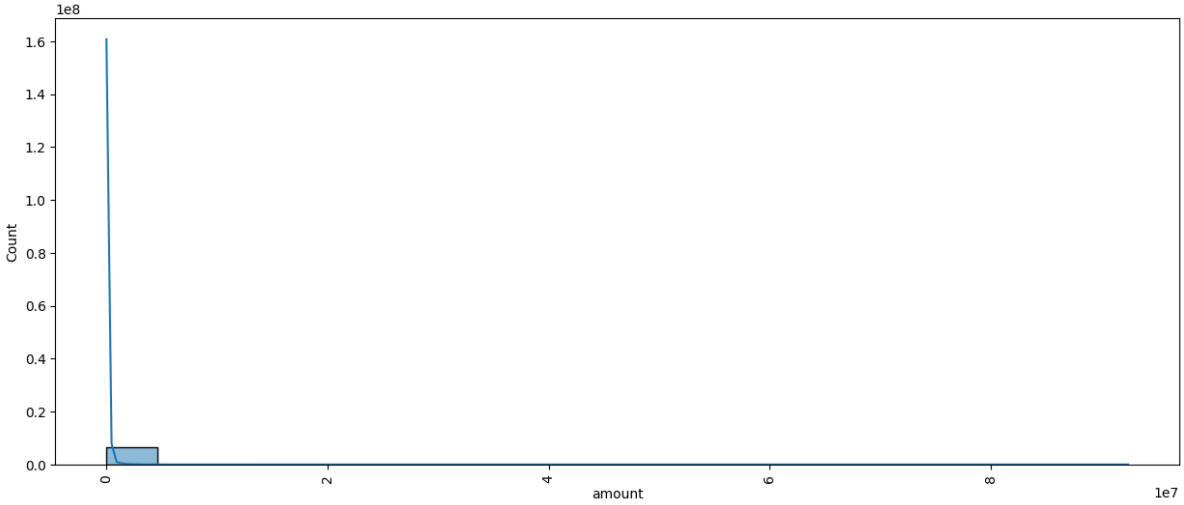
```
        autopct='%1.1f%%',
        colors=['#1f77b4', '#ff7f0e'],  # Setting colors
        textprops={'color': 'black', 'weight': 'bold', 'family': 'serif'},
        wedgeprops={'linewidth': 4, 'edgecolor': 'white'})  # Border proper

# Set title with custom font properties
plt.title('isFraud', size=2, **hfont)

plt.show()
```

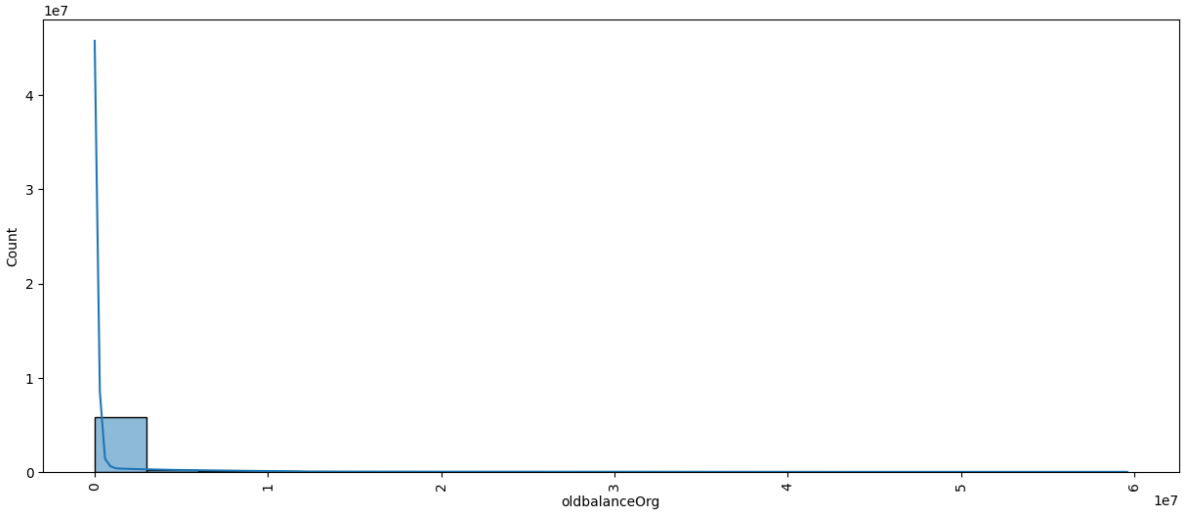


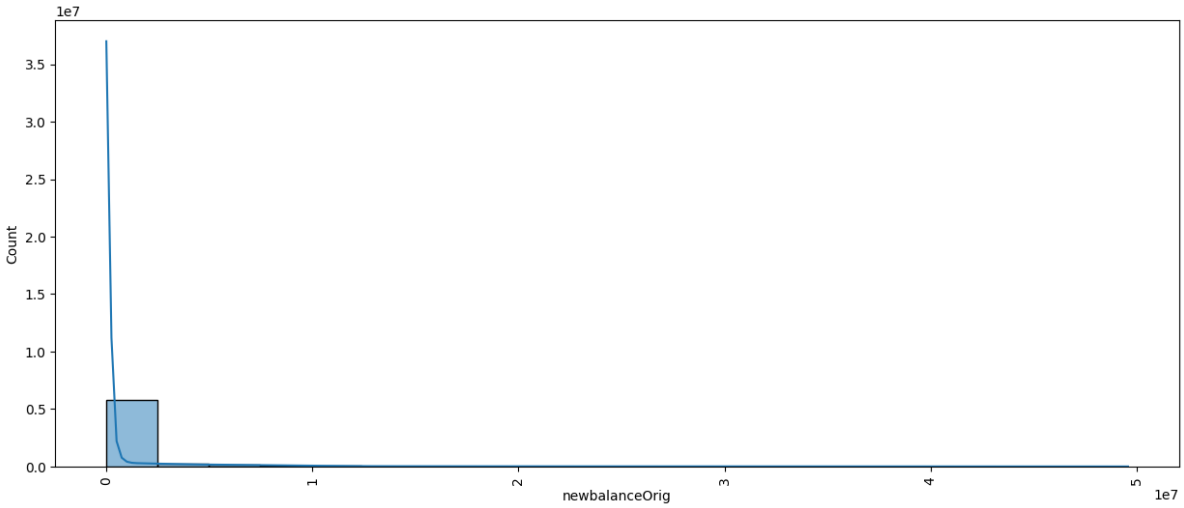
In [73]:

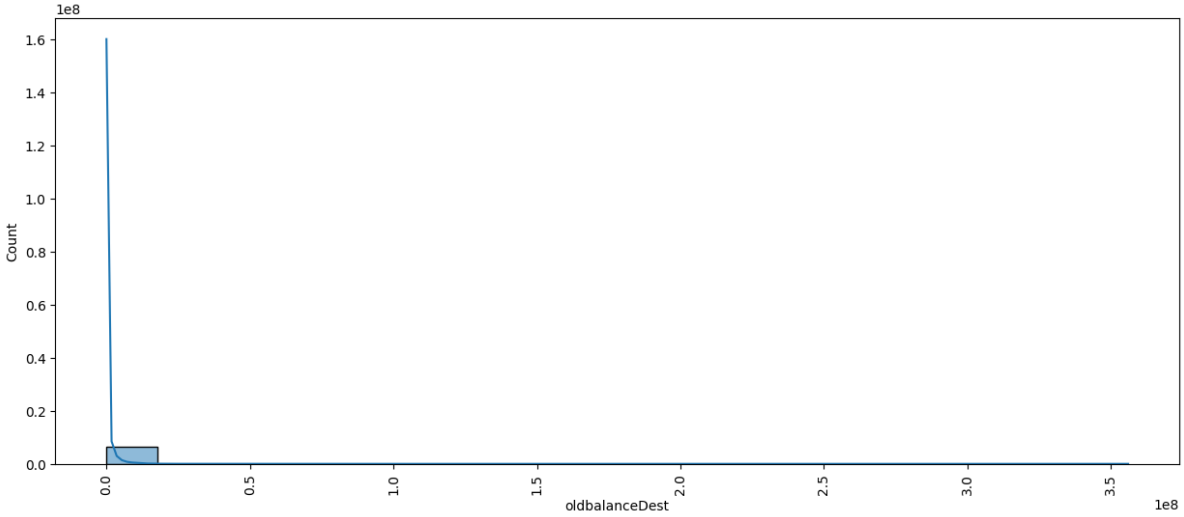
```
import seaborn as sns
import matplotlib.pyplot as plt

for i in numerical_columns:
    if i != 'isFraud' and i != 'isFlaggedFraud':
        plt.figure(figsize=(15,6))
        sns.histplot(df[i], kde=True, bins=20, palette='hls')
        plt.xticks(rotation=90)
        plt.show()
```

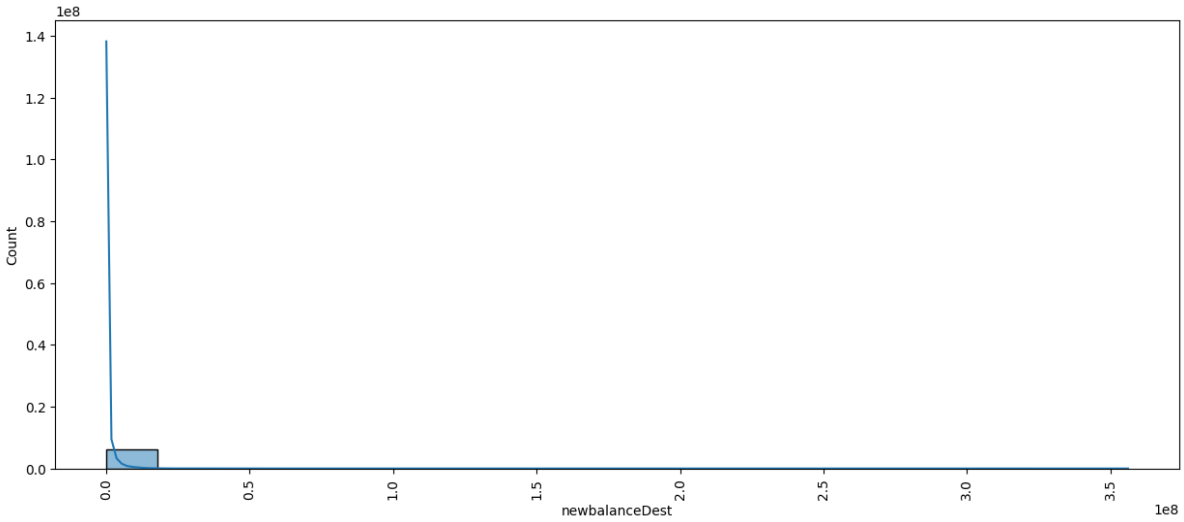



1e8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
Count
0
2
4
6
8
amount
1e7

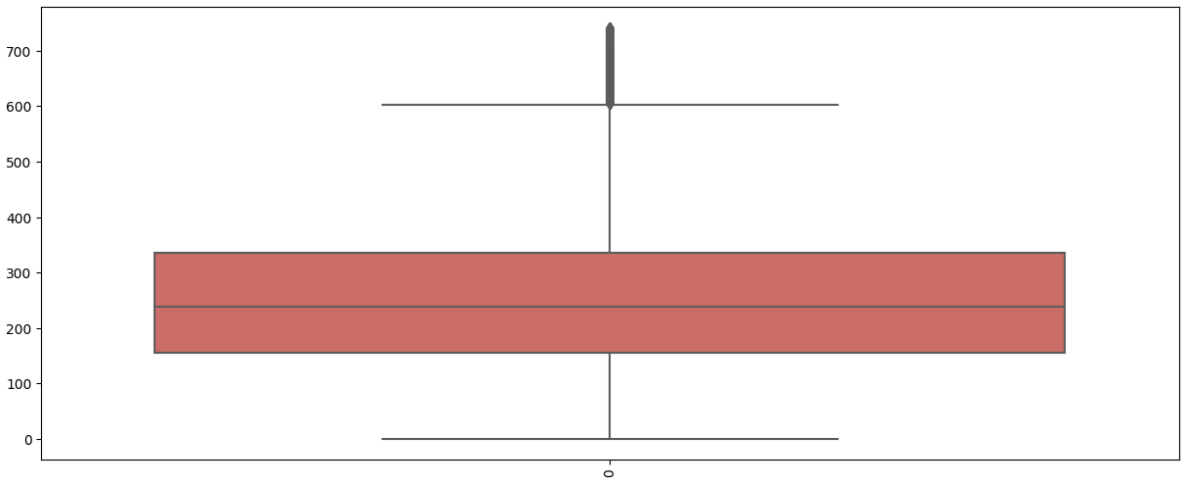

1e7
4
3
2
1
0
Count
0
1
2
3
4
5
6
oldbalanceOrg
1e7

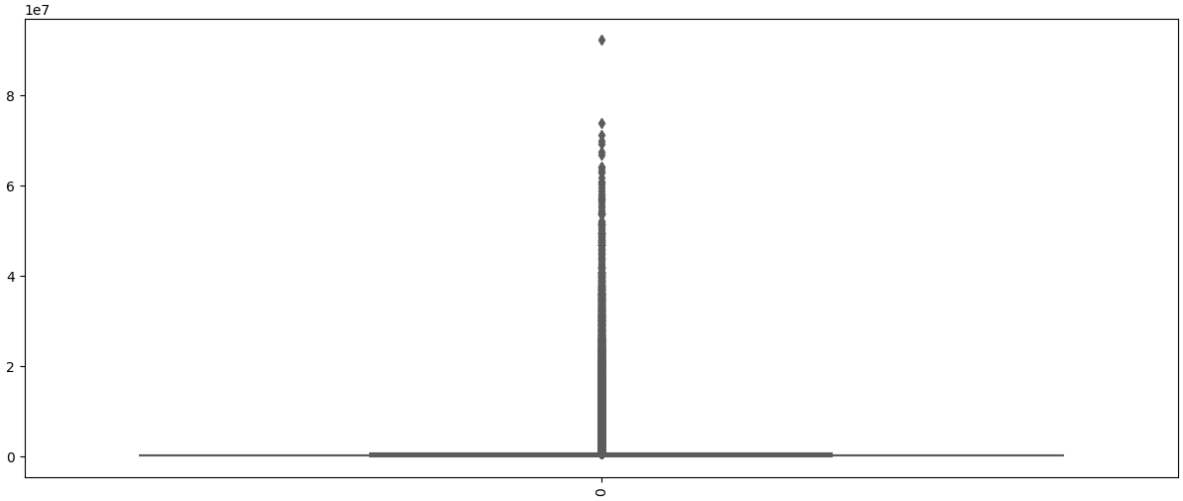

1e7
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Count
0
1
2
3
4
5
newbalanceOrig
1e7

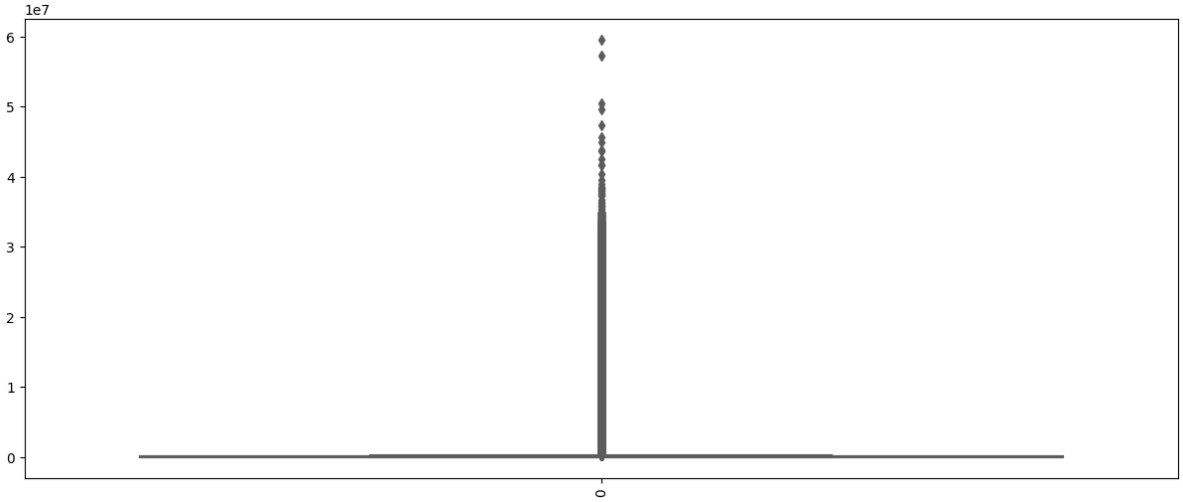


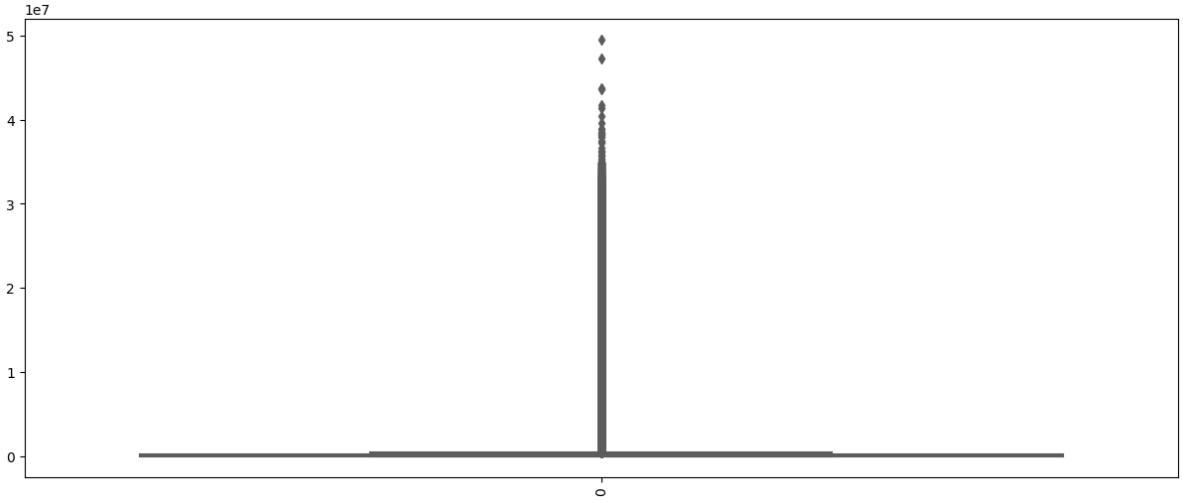


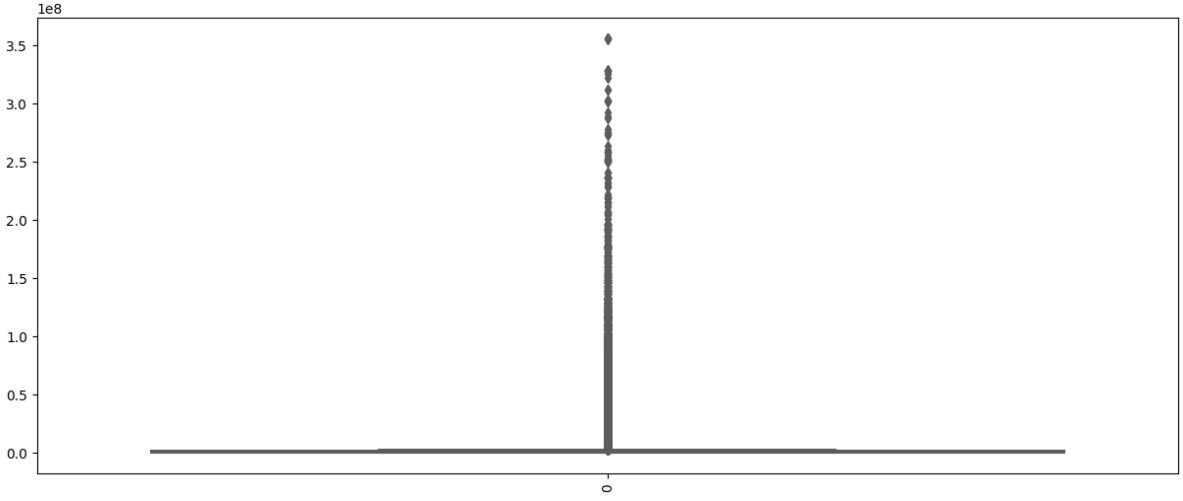
In [79]:
```
import seaborn as sns
import matplotlib.pyplot as plt

for i in numerical_columns:
    if i != 'isFraud':
        if i != 'isFlaggedFraud':
            plt.figure(figsize=(15,6))
            sns.boxplot(df[i], palette='hls')
            plt.xticks(rotation=90)
            plt.show()
```

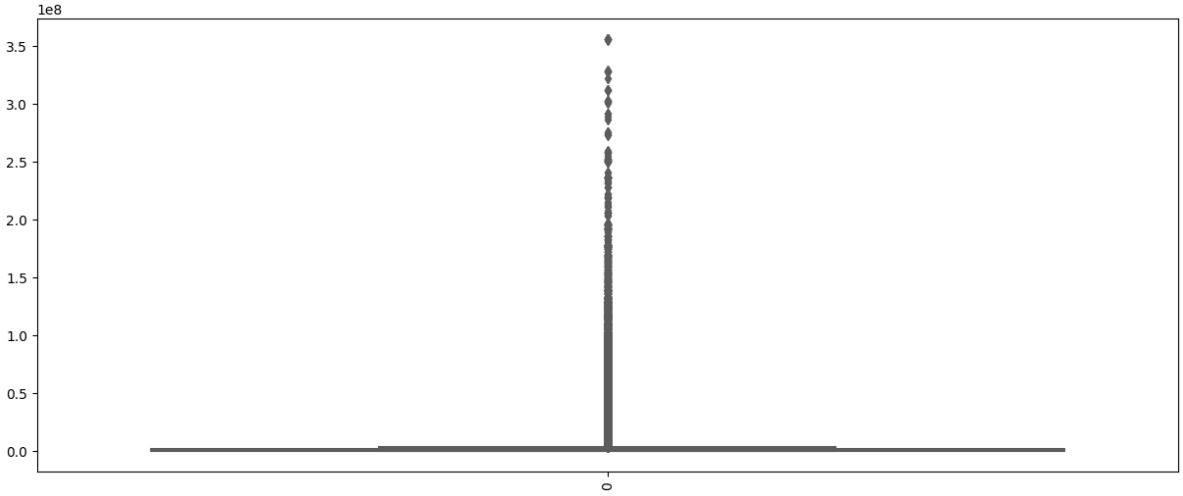



1e7
8
6
4
2
0
0

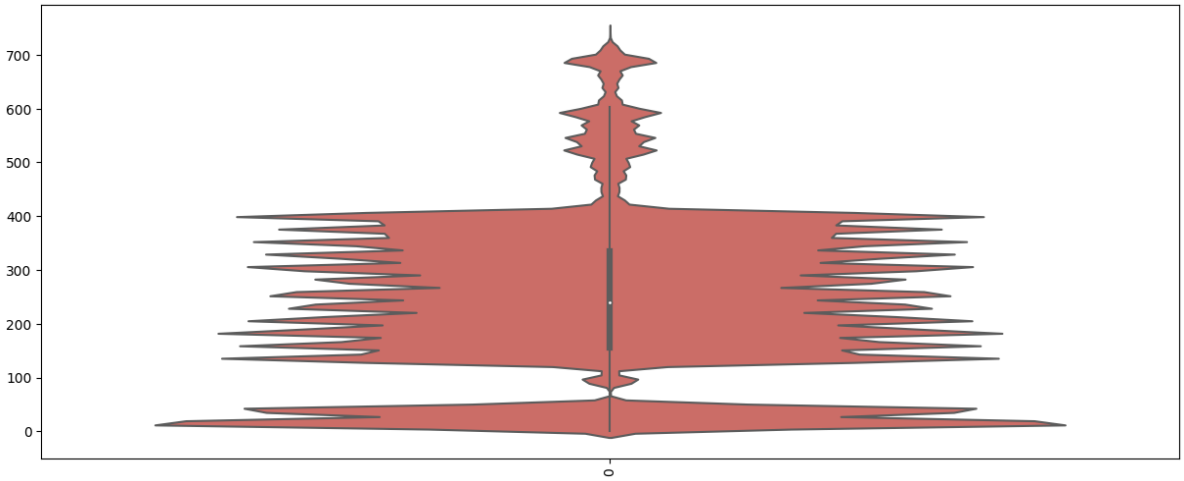

1e7
6
5
4
3
2
1
0
0

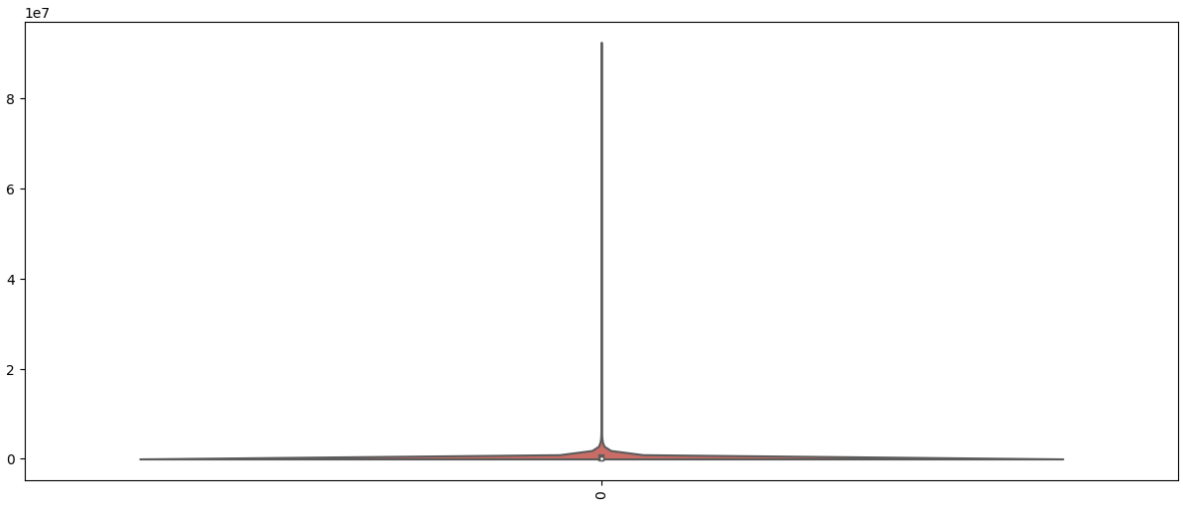

1e7
5
4
3
2
1
0
0

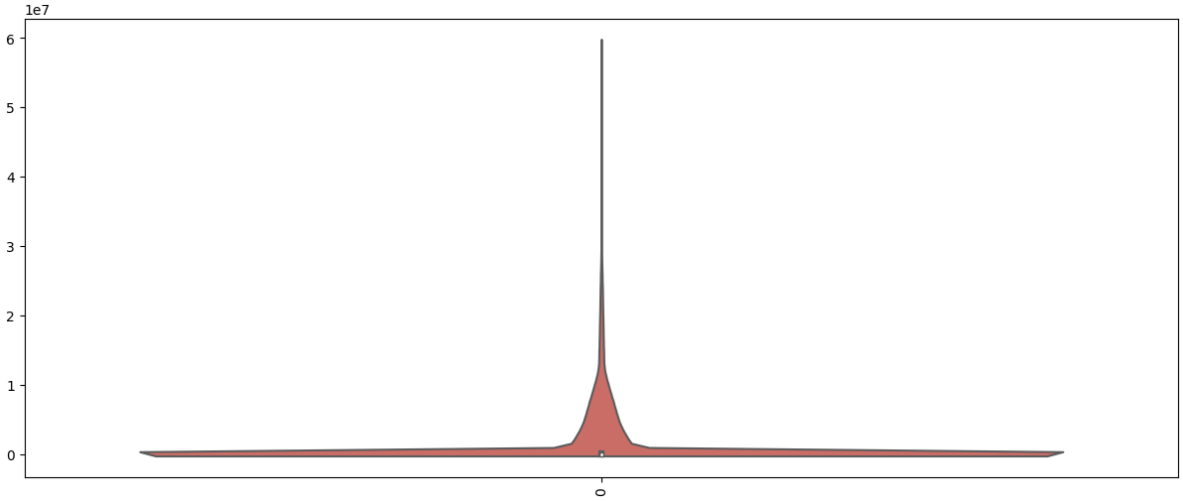

1e8
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0

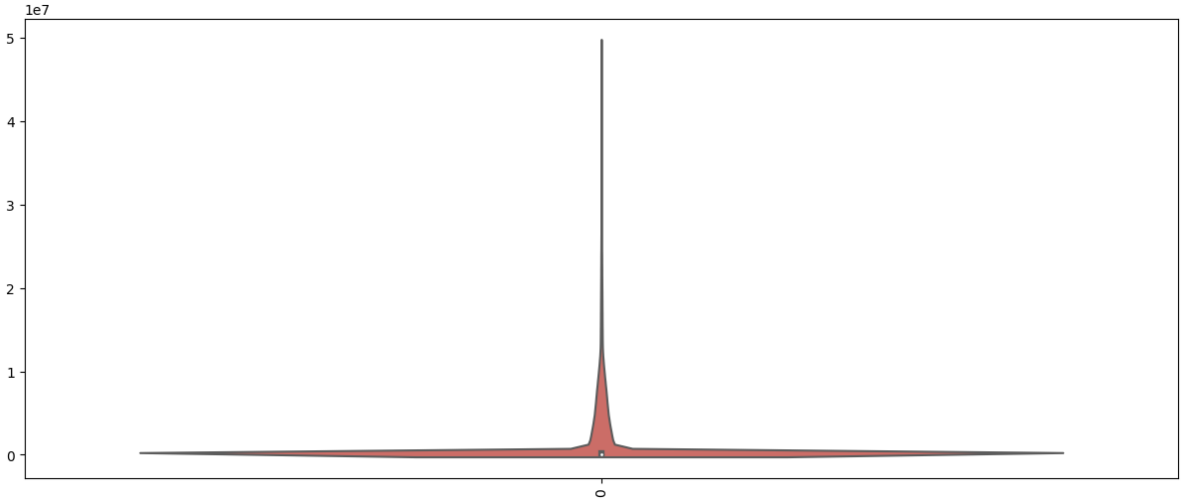


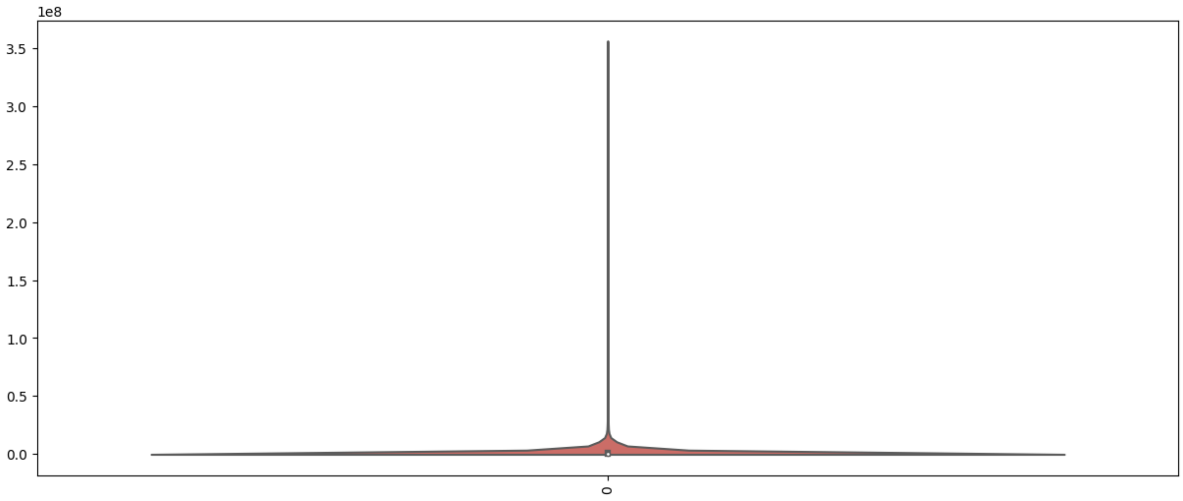
In [83]:
```
for i in numerical_columns:
    if i != 'isFraud':
        if i != 'isFlaggedFraud':
            plt.figure(figsize=(15,6))
            sns.violinplot(df[i], palette = 'hls')
            plt.xticks(rotation = 90)
            plt.show()
```

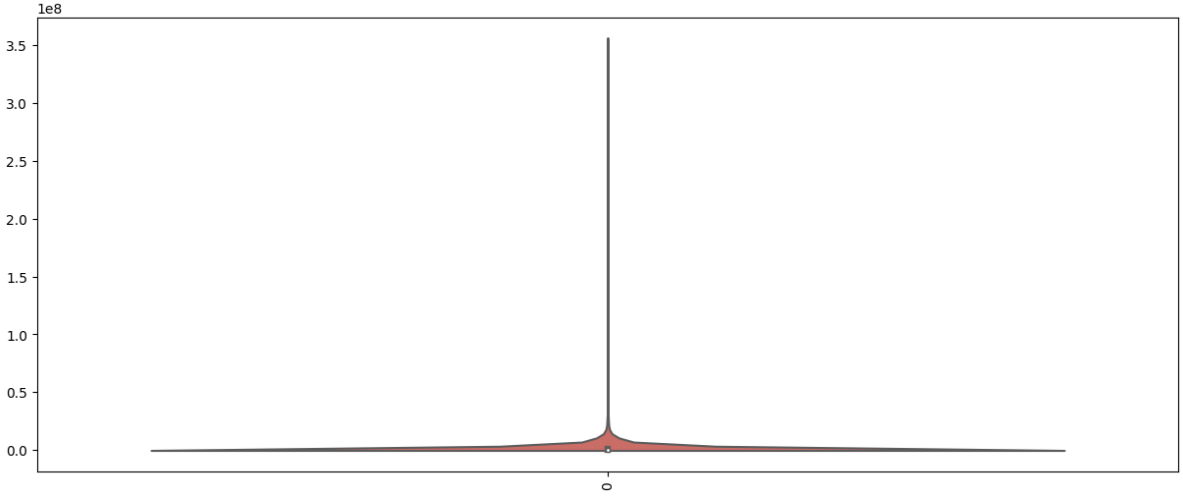




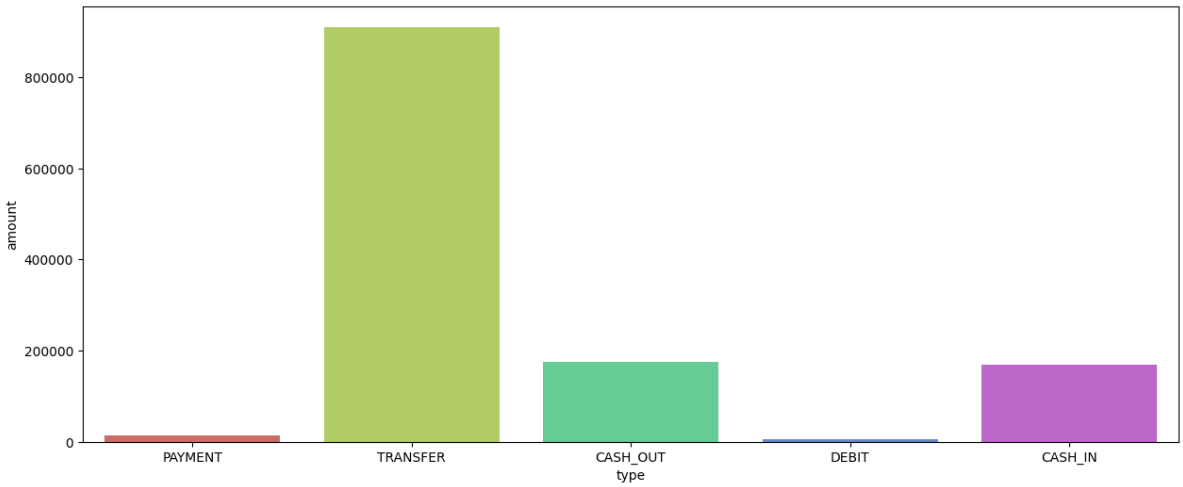

1e7
6
5
4
3
2
1
0
0


1e7
5
4
3
2
1
0
0

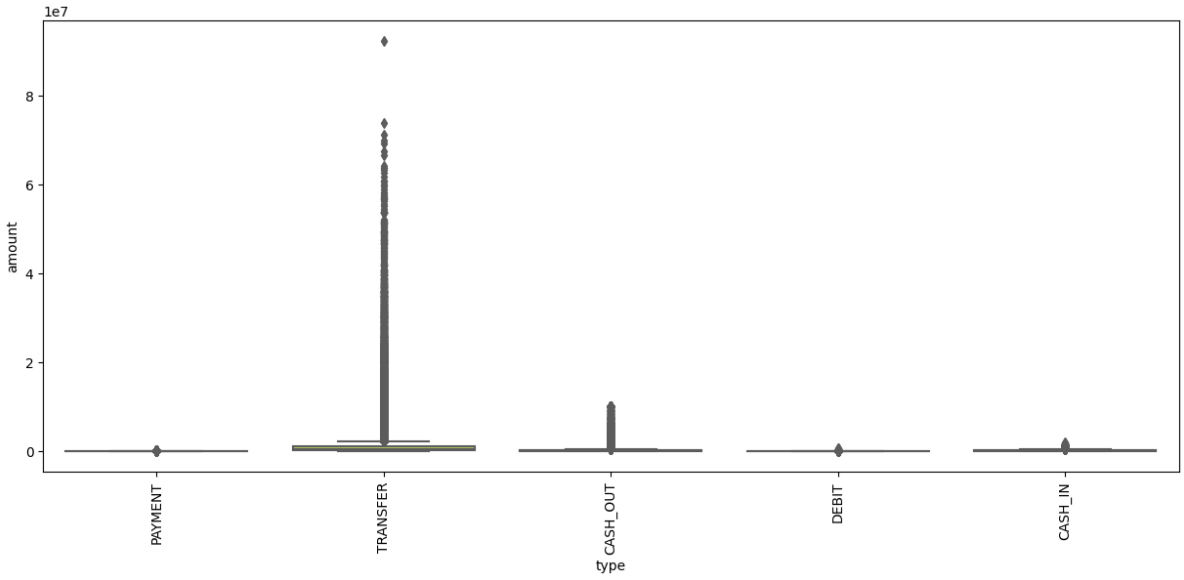

1e8
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0


1e8
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0

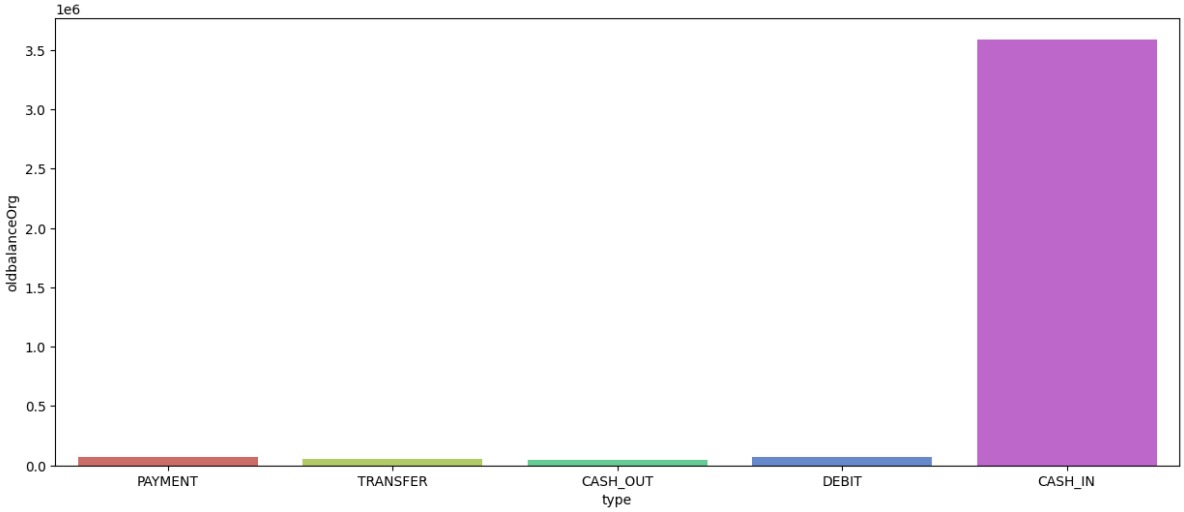
In [86]:
```
plt.figure(figsize=(15,6))
sns.barplot(x = df['type'], y = df['amount'], data = df, ci = None, palette
plt.show()
```



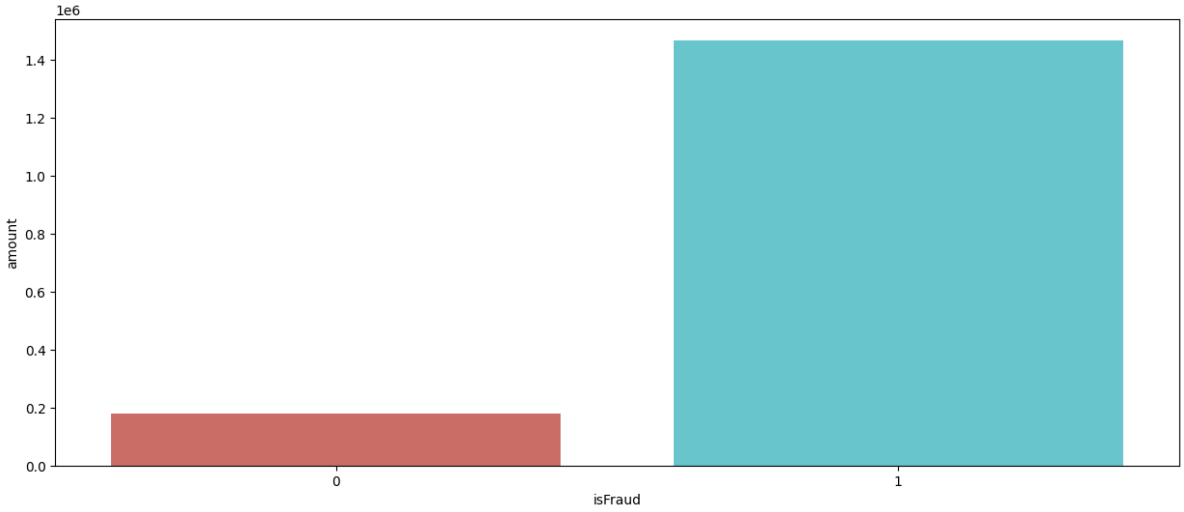
In [87]:
```
plt.figure(figsize=(15,6))
sns.boxplot(x = df['type'], y = df['amount'], data = df, palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



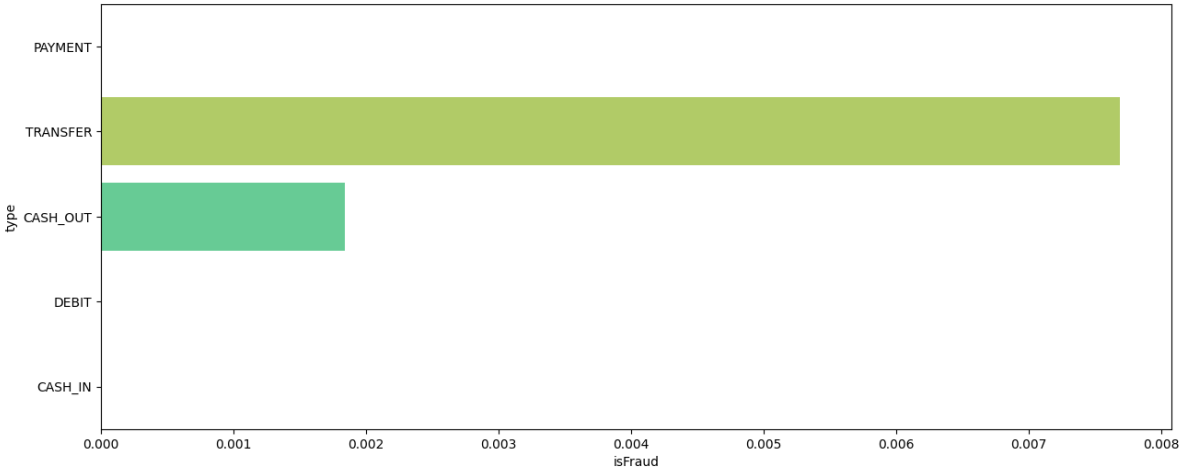
In [89]:
```
plt.figure(figsize=(15,6))
sns.barplot(x=df['type'], y=df['oldbalanceOrg'], data=df, ci=None, palette=
plt.show()
```



In [90]:
```
plt.figure(figsize=(15,6))
sns.barplot(x = df['isFraud'], y = df['amount'], data = df, ci = None, pale
plt.show()
```



In [91]:
```
plt.figure(figsize=(15,6))
sns.barplot(x = df['isFraud'], y = df['type'], data = df, ci = None, palett
plt.show()
```



In [95]:
```

```