Loading The Dataset

In [7]:
```
import pandas as pd
df=pd.read_csv("E:\\R.P 2\\Data-Credit Card Transactions Fraud Detection 2019-2020\
df.head()
```

Out[7]:

	Unnamed: 0	trans_date_trans_time	cc_num	merchant	category	amt	
0	0	1/1/2019 0:00	2.700000e+15	fraud_Rippin, Kub and Mann	misc_net	4.97	.
1	1	1/1/2019 0:00	6.300000e+11	fraud_Heller, Gutmann and Zieme	grocery_pos	107.23	Ste
2	2	1/1/2019 0:00	3.890000e+13	fraud_Lind-Buckridge	entertainment	220.11	
3	3	1/1/2019 0:01	3.530000e+15	fraud_Kutch, Hermiston and Farrell	gas_transport	45.00	
4	4	1/1/2019 0:03	3.760000e+14	fraud_Keeling-Crist	misc_pos	41.96	

5 rows × 23 columns

In [3]:
```
df.shape
```

Out[3]:
```
(1048575, 23)
```

Preprocessing For Column "trans_date_trans_time"

"trans_date_trans_time" Converting Format & adding new Columns And Time Gap

In [4]:
```
# 1. Convert to DateTime
```

```
df['trans_date_trans_time'] = pd.to_datetime(df['trans_date_trans_time'])
```

In [5]:
```
# 2. Extract Hour, Day, Month, Weekday (2)
df['hour'] = df['trans_date_trans_time'].dt.hour
df['day'] = df['trans_date_trans_time'].dt.day
df['month'] = df['trans_date_trans_time'].dt.month
df['weekday'] = df['trans_date_trans_time'].dt.weekday  # Monday=0
```

In [6]:
```
# 3. Time gap per card (in minutes) (2)
df = df.sort_values(by=['cc_num','trans_date_trans_time'])
df['time_gap'] = df.groupby('cc_num')['trans_date_trans_time'].diff().dt.total_seco
```

In [6]:
```
# Missing gaps (first transaction per card) → fill with 0
df['time_gap'] = df['time_gap'].fillna(0)
df.sample(5)
```

Out[6]:

	Unnamed: 0	trans_date_trans_time	cc_num	merchant	category	a
508986	508986	2019-08-09 02:14:00	3.580000e+15	fraud_Hintz-Bruen	grocery_net	52
744346	744346	2019-11-14 17:59:00	4.100000e+15	fraud_Kihn-Schuster	food_dining	4
810028	810028	2019-12-07 00:36:00	2.280000e+15	fraud_Raynor, Reinger and Hagenes	gas_transport	48
392177	392177	2019-06-28 20:07:00	5.040000e+11	fraud_Cormier LLC	shopping_net	6
39713	39713	2019-01-24 05:54:00	3.510000e+15	fraud_Romaguera, Cruickshank and Greenholt	shopping_net	91

5 rows × 28 columns

In [7]:
```
df.groupby('is_fraud')['time_gap'].describe()
```

Out[7]:

is_fraud	count	mean	std	min	25%	50%	75%	max
0	1042569.0	187.480501	421.411011	0.0	15.0	52.0	176.0	20095.0
1	6006.0	126.903430	305.115645	0.0	10.0	29.0	86.0	8388.0

In [8]:
```
df[df['time_gap'] < 5]   # transactions within 5 minutes
```

Out[8]:

	Unnamed: 0	trans_date_trans_time	cc_num	merchant	category
1017	1017	2019-01-01 12:47:00	6.041621e+10	fraud_Jones, Sawayn and Romaguera	misc_net
2726	2726	2019-01-02 08:47:00	6.041621e+10	fraud_Luettgen PLC	gas_transport
14342	14342	2019-01-08 23:24:00	6.041621e+10	fraud_Effertz, Welch and Schowalter	entertainment
28631	28631	2019-01-17 19:24:00	6.041621e+10	fraud_Nitzsche, Kessler and Wolff	shopping_pos
106628	106628	2019-03-02 22:10:00	6.041621e+10	fraud_Windler, Goodwin and Kovacek	home
...	...	...	...	...	...
920748	920748	2019-12-30 19:58:00	4.990000e+18	fraud_Bartoletti and Sons	personal_care
930193	930193	2020-01-04 16:35:00	4.990000e+18	fraud_Parker, Nolan and Trantow	entertainment
1019094	1019094	2020-02-24 19:32:00	4.990000e+18	fraud_McDermott, Osinski and Morar	home
1044168	1044168	2020-03-09 07:37:00	4.990000e+18	fraud_Brown PLC	misc_net
1045844	1045844	2020-03-09 18:52:00	4.990000e+18	fraud_Walter, Hettinger and Kessler	personal_care

101048 rows × 28 columns

Preprocessing For Column "Unnamed: 0"

In [9]:
```
df = df.drop('Unnamed: 0', axis=1)
```

Preprocessing For Column "cc_num"

In [10]:
```
df['cc_num'] = df['cc_num'].astype(str)
```

Preprocessing For Column "merchant"

In [11]:
```
df['merchant'] = df['merchant'].astype(str)
```

Preprocessing For Column "category"

In [12]:
```
df['category'] = df['category'].astype(str)
```

Preprocessing For Column "amt"

In [13]:
```
df['amt'] = df['amt'].astype(float)
```

Preprocessing For Column "first""last" (Droped its just Customer name not usefull Feature)

In [14]:
```
df = df.drop(['first','last'], axis=1)
```

Preprocessing For Column "gender"

In [70]:
```
df['gender'] = df['gender'].astype(str)
```

Preprocessing For Column "street" (Droped not usefull it is only Cutomer adress)

In [71]:
```
df = df.drop('street', axis=1)
```

Preprocessing For Column "city, state"

In [72]:
```
df['city'] = df['city'].astype(str)
df['state'] = df['state'].astype(str)
```

Preprocessing For Column "zip"

In [73]:
```
df['zip'] = df['zip'].astype(str)
```

Preprocessing For Column "lat, long"

In [74]:
```
df['lat'] = df['lat'].astype(float)
df['long'] = df['long'].astype(float)
```

Preprocessing For Column "city_pop"

In [75]:
```
df['city_pop'] = df['city_pop'].astype(int)
```

Preprocessing For Column "job"

In [76]:
```
df['job'] = df['job'].astype(str)
```

Preprocessing For Column "dob" using this Created Age

In [77]:
```
df['dob'] = pd.to_datetime(df['dob'])
df['age'] = df['trans_date_trans_time'].dt.year - df['dob'].dt.year
```

Preprocessing For Column "trans_num" (Droped not usefull overall)

In [78]:
```
df = df.drop('trans_num', axis=1)
```

Preprocessing For Column "unix_time"

In [79]:
```
df['unix_time'] = df['unix_time'].astype(int)
```

Preprocessing For Column "merch_lat, merch_long"

In [80]:
```
df['merch_lat'] = df['merch_lat'].astype(float)
df['merch_long'] = df['merch_long'].astype(float)
```

Preprocessing For Column "is_fraud"

In [81]:
```
df['is_fraud'] = df['is_fraud'].astype(int)
```

In [85]:
```
df.isnull().sum()
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1048575 entries, 1017 to 1047587
Data columns (total 24 columns):
 #   Column                 Non-Null Count    Dtype
---  ------                 --------------    -----
 0   trans_date_trans_time  1048575 non-null  datetime64[ns]
 1   cc_num                 1048575 non-null  object
 2   merchant               1048575 non-null  object
 3   category               1048575 non-null  object
 4   amt                    1048575 non-null  float64
 5   gender                 1048575 non-null  object
 6   city                   1048575 non-null  object
 7   state                  1048575 non-null  object
 8   zip                    1048575 non-null  object
 9   lat                    1048575 non-null  float64
 10  long                   1048575 non-null  float64
 11  city_pop               1048575 non-null  int64
 12  job                    1048575 non-null  object
 13  dob                    1048575 non-null  datetime64[ns]
 14  unix_time              1048575 non-null  int64
 15  merch_lat              1048575 non-null  float64
 16  merch_long             1048575 non-null  float64
 17  is_fraud               1048575 non-null  int64
 18  hour                   1048575 non-null  int32
 19  day                    1048575 non-null  int32
 20  month                  1048575 non-null  int32
 21  weekday                1048575 non-null  int32
 22  time_gap               1048575 non-null  float64
 23  age                    1048575 non-null  int32
dtypes: datetime64[ns](2), float64(6), int32(5), int64(3), object(8)
memory usage: 180.0+ MB
```

Checking Duplicate Transitions

In [83]:
```
df.duplicated().sum()
```

Out[83]:
```
np.int64(0)
```

Class Imbalance Check

In [84]:
```
df['is_fraud'].value_counts(normalize=True)*100
```

Out[84]:
```
is_fraud
0    99.427223
1     0.572777
Name: proportion, dtype: float64
```

Exporting Prprocessed Data to CSV File

In [86]:
```
# Assuming 'df' is your final preprocessed dataframe
df.to_csv('cleaned_data.csv', index=False)
```

```
print("File exported successfully as cleaned_data.csv")
```

```
File exported successfully as cleaned_data.csv
```