```
In [24]: # TASK1 LEVEL1: TOP CUISINES
```

```
In [25]: # IMPORTING LIBRARIES
```

```
In [1]: import numpy as n
        import pandas as pd
        import matplotlib.pyplot as plt
        %matplotlib inline
        import seaborn as sns
```

```
In [26]: # DATA COLLECTION
```

```
In [2]: df=pd.read_csv('Dataset .csv',encoding='unicode_escape')
```

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

Out[3]:

	ï»¿Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	C
0	6317637	Le Petit Souffle	162	Makati City	Third Floor, Century City Mall, Kalayaan Avenu...	Century City Mall, Poblacion, Makati City	Century City Mall, Poblacion, Makati City, Mak...	121.027535	14.565443	French, Japanese, Desserts	...	B
1	6304287	Izakaya Kikufuji	162	Makati City	Little Tokyo, 2277 Chino Roces Avenue, Legaspi...	Little Tokyo, Legaspi Village, Makati City	Little Tokyo, Legaspi Village, Makati City, Ma...	121.014101	14.553708	Japanese	...	B
2	6300002	Heat - Edsa Shangri-La	162	Mandaluyong City	Edsa Shangri-La, 1 Garden Way, Ortigas, Mandal...	Edsa Shangri-La, Ortigas, Mandaluyong City	Edsa Shangri-La, Ortigas, Mandaluyong City, Ma...	121.056831	14.581404	Seafood, Asian, Filipino, Indian	...	B
3	6318506	Ooma	162	Mandaluyong City	Third Floor, Mega Fashion Hall, SM Megamall, O...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.056475	14.585318	Japanese, Sushi	...	B
4	6314302	Sambo Kojin	162	Mandaluyong City	Third Floor, Mega Atrium, SM Megamall, Ortigas...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.057508	14.584450	Japanese, Korean	...	B

5 rows × 21 columns

```
In [4]: df.shape
```

Out[4]: (9551, 21)

```
In [5]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9551 entries, 0 to 9550
Data columns (total 21 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   ï»¿Restaurant ID      9551 non-null   int64
 1   Restaurant Name       9551 non-null   object
 2   Country Code          9551 non-null   int64
 3   City                  9551 non-null   object
 4   Address               9551 non-null   object
 5   Locality              9551 non-null   object
 6   Locality Verbose      9551 non-null   object
 7   Longitude             9551 non-null   float64
 8   Latitude              9551 non-null   float64
 9   Cuisines              9542 non-null   object
 10  Average Cost for two  9551 non-null   int64
 11  Currency              9551 non-null   object
 12  Has Table booking     9551 non-null   object
 13  Has Online delivery   9551 non-null   object
 14  Is delivering now     9551 non-null   object
 15  Switch to order menu  9551 non-null   object
 16  Price range           9551 non-null   int64
 17  Aggregate rating      9551 non-null   float64
 18  Rating color          9551 non-null   object
 19  Rating text           9551 non-null   object
 20  Votes                 9551 non-null   int64
dtypes: float64(3), int64(5), object(13)
memory usage: 1.5+ MB
```

In [27]: `# TO CHECK NULL VALUES`

In [6]: `pd.isnull(df).sum()`

Out[6]:
```
ï»¿Restaurant ID        0
Restaurant Name         0
Country Code            0
City                    0
Address                 0
Locality                0
Locality Verbose        0
Longitude               0
Latitude                0
Cuisines                9
Average Cost for two    0
Currency                0
Has Table booking       0
Has Online delivery     0
Is delivering now       0
Switch to order menu    0
Price range             0
Aggregate rating        0
Rating color            0
Rating text             0
Votes                   0
dtype: int64
```

In [28]: `# TO DELETE NULL VALUES`

In [7]: `df.dropna(inplace=True)`

In [9]: `df.shape`

Out[9]: `(9542, 21)`

In [10]: `df.describe()`

Out[10]:

	ï»¿Restaurant ID	Country Code	Longitude	Latitude	Average Cost for two	Price range	Aggregate rating	Votes
count	9.542000e+03	9542.000000	9542.000000	9542.000000	9542.000000	9542.000000	9542.000000	9542.000000
mean	9.043301e+06	18.179208	64.274997	25.848532	1200.326137	1.804968	2.665238	156.772060
std	8.791967e+06	56.451600	41.197602	11.010094	16128.743876	0.905563	1.516588	430.203324
min	5.300000e+01	1.000000	-157.948486	-41.330428	0.000000	1.000000	0.000000	0.000000
25%	3.019312e+05	1.000000	77.081565	28.478658	250.000000	1.000000	2.500000	5.000000
50%	6.002726e+06	1.000000	77.192031	28.570444	400.000000	2.000000	3.200000	31.000000
75%	1.835260e+07	1.000000	77.282043	28.642711	700.000000	2.000000	3.700000	130.000000
max	1.850065e+07	216.000000	174.832089	55.976980	800000.000000	4.000000	4.900000	10934.000000

```
In [11]:  df.columns

Out[11]:  Index(['ï»¿Restaurant ID', 'Restaurant Name', 'Country Code', 'City',
                 'Address', 'Locality', 'Locality Verbose', 'Longitude', 'Latitude',
                 'Cuisines', 'Average Cost for two', 'Currency', 'Has Table booking',
                 'Has Online delivery', 'Is delivering now', 'Switch to order menu',
                 'Price range', 'Aggregate rating', 'Rating color', 'Rating text',
                 'Votes'],
                dtype='object')

In [29]:  # TO CHECK DUPLICATES

In [16]:  df[df.duplicated()]
```

Out[16]:

	ï»¿Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Currency	Has Table booking	Or deli

0 rows × 21 columns

```
In [30]:  # TO CHECK DATATYPES

In [17]:  df.dtypes

Out[17]:  ï»¿Restaurant ID          int64
          Restaurant Name         object
          Country Code             int64
          City                    object
          Address                 object
          Locality                object
          Locality Verbose        object
          Longitude              float64
          Latitude               float64
          Cuisines                object
          Average Cost for two     int64
          Currency                object
          Has Table booking       object
          Has Online delivery     object
          Is delivering now       object
          Switch to order menu    object
          Price range              int64
          Aggregate rating       float64
          Rating color            object
          Rating text             object
          Votes                    int64
          dtype: object

In [18]:  df.columns

Out[18]:  Index(['ï»¿Restaurant ID', 'Restaurant Name', 'Country Code', 'City',
                 'Address', 'Locality', 'Locality Verbose', 'Longitude', 'Latitude',
                 'Cuisines', 'Average Cost for two', 'Currency', 'Has Table booking',
                 'Has Online delivery', 'Is delivering now', 'Switch to order menu',
                 'Price range', 'Aggregate rating', 'Rating color', 'Rating text',
                 'Votes'],
                dtype='object')

In [31]:  # DETERMINE THE TOP THREE MOST COMMON CUISINES IN THE DATASET.

In [19]:  df['Cuisines'].value_counts()

Out[19]:  Cuisines
          North Indian                                            936
          North Indian, Chinese                                   511
          Chinese                                                 354
          Fast Food                                               354
          North Indian, Mughlai                                   334
                                                                  ...
          Bengali, Fast Food                                        1
          North Indian, Rajasthani, Asian                          1
          Chinese, Thai, Malaysian, Indonesian                     1
          Bakery, Desserts, North Indian, Bengali, South Indian    1
          Italian, World Cuisine                                   1
          Name: count, Length: 1825, dtype: int64

In [20]:  df['Cuisines'].value_counts().head(3)
```
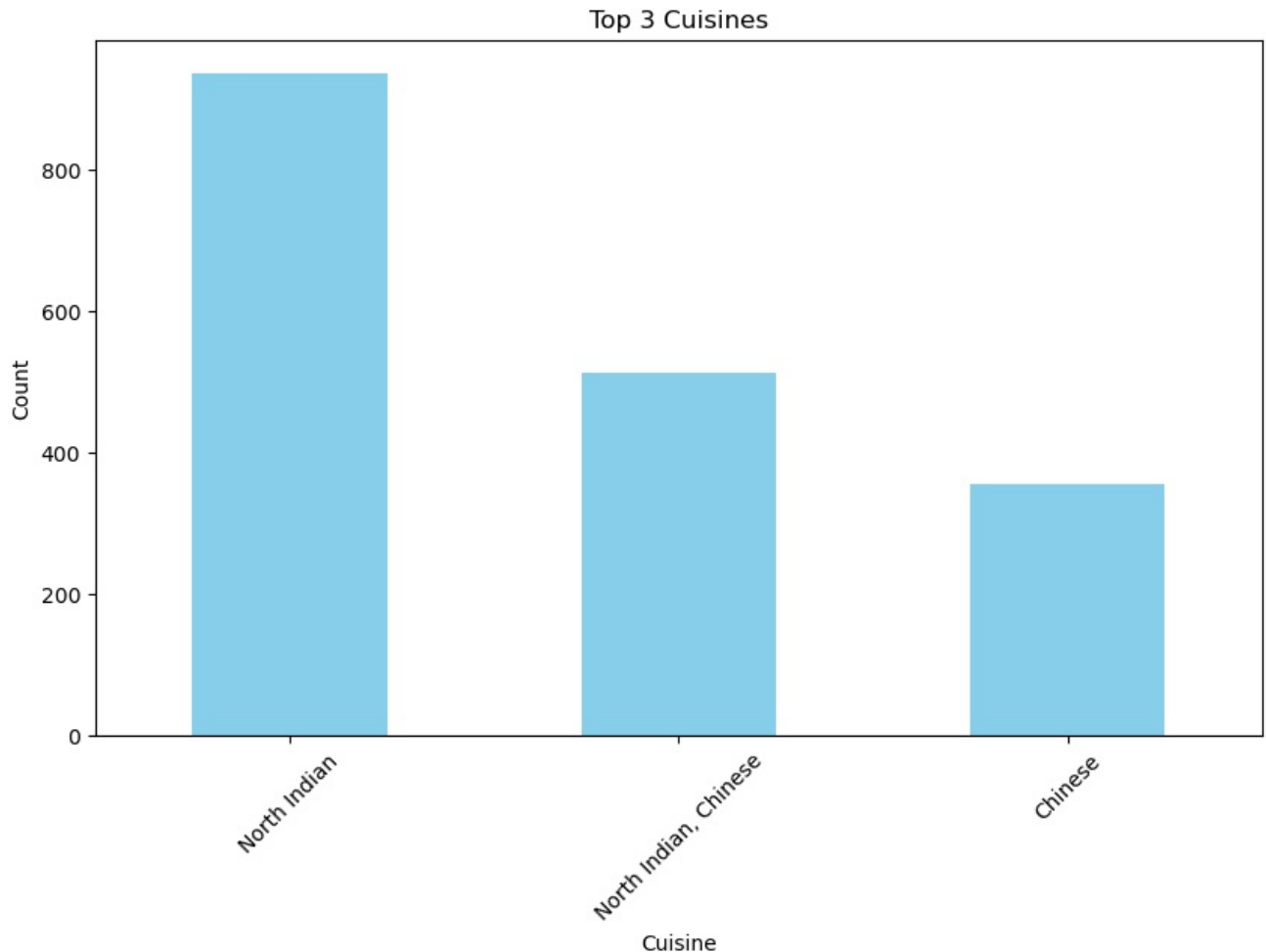
```
Out[20]:  Cuisines
          North Indian             936
          North Indian, Chinese    511
          Chinese                  354
          Name: count, dtype: int64
```

```
In [21]:  Top3_Cuisines=df['Cuisines'].value_counts().head(3)
```

```
In [32]:   plt.figure(figsize=(10, 6))
           Top3_Cuisines.plot(kind='bar', color='skyblue')
           plt.title('Top 3 Cuisines')
           plt.xlabel('Cuisine')
           plt.ylabel('Count')
           plt.xticks(rotation=45)  # Rotate x-axis labels for better readability
           plt.show()
```


Top 3 Cuisines

```
In [33]:  # CALCULATE THE PERCENTAGE OF RESTAURANTS THAT SERVE EACH OF THE TOP CUISINES.
```

```
In [34]:  df.columns
```

```
Out[34]:  Index(['ï»¿Restaurant ID', 'Restaurant Name', 'Country Code', 'City',
                 'Address', 'Locality', 'Locality Verbose', 'Longitude', 'Latitude',
                 'Cuisines', 'Average Cost for two', 'Currency', 'Has Table booking',
                 'Has Online delivery', 'Is delivering now', 'Switch to order menu',
                 'Price range', 'Aggregate rating', 'Rating color', 'Rating text',
                 'Votes'],
                dtype='object')
```

```
In [36]:  Top3_Cuisines=df['Cuisines'].value_counts().head(3)
```

```
In [37]:  # Total number of restaurants
          total_restaurants = len(df)
```

```
In [38]:  # Calculate the percentage of restaurants for each of the top cuisines
          percentage_cuisines = (Top3_Cuisines / total_restaurants) * 100
```

```
In [39]:  # Display the percentage of restaurants for each cuisine
          print("Percentage of restaurants serving each of the top cuisines:")
          print(percentage_cuisines)
```

```
Percentage of restaurants serving each of the top cuisines:
Cuisines
North Indian            9.809264
North Indian, Chinese   5.355271
Chinese                 3.709914
Name: count, dtype: float64
```
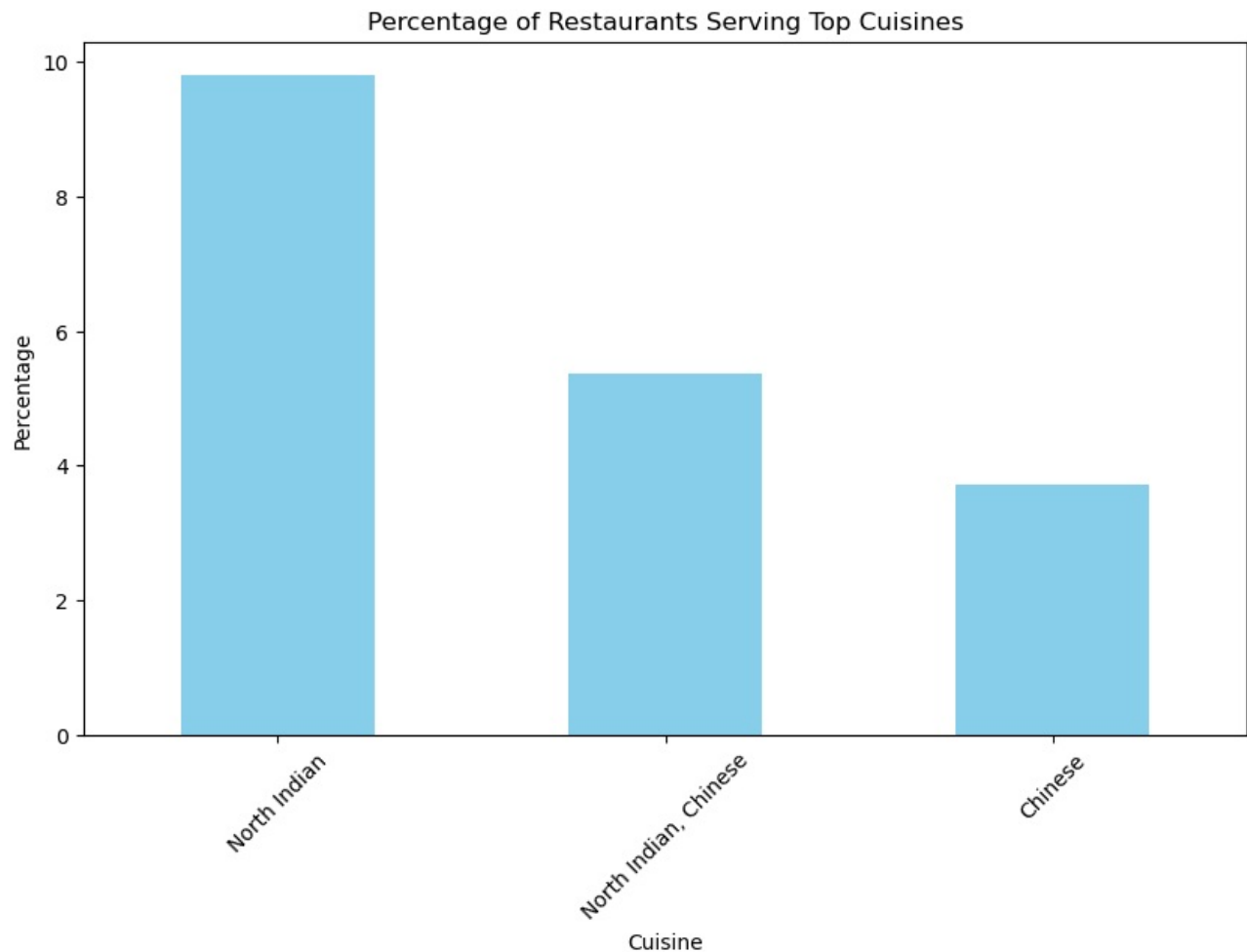
In [40]:
```python
plt.figure(figsize=(10, 6))
percentage_cuisines.plot(kind='bar', color='skyblue')
plt.title('Percentage of Restaurants Serving Top Cuisines')
plt.xlabel('Cuisine')
plt.ylabel('Percentage')
plt.xticks(rotation=45)  # Rotate x-axis labels for better readability
plt.show()
```


Percentage of Restaurants Serving Top Cuisines

In [41]:
```python
# THANKYOU
```

In [42]:
```python
# CONNECT WITH ME:
# LinkedIn: https://www.linkedin.com/in/harshita-sharma-b68154220/
# GitHub: https://github.com/DATAPREDICTS
# Instagram: https://www.instagram.com/datapredicts?utm_source=qr&igsh=czVzc2k5c3oxOWQ4
# YouTube: https://youtube.com/@Datapredicts?si=eDKAqVciVxg23zab
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js