Data analytics using Pandas in Python is one of the most popular ways to manipulate and analyze data due to its powerful, easy-to-use data structures like Series and Data Frame. Pandas provides a lot of flexibility for reading data, cleaning it, and performing complex operations. Here's a basic overview of how you can get started with data analytics in Pandas:

1. **Installing Pandas**
If you don't have Pandas installed, you can install it using `pip`:

2. **Importing Pandas**
You can import pandas using the following code:

python import pandas as pd

3. **Creating Data Frames**
A Data Frame is the main data structure in Pandas. It’s like a table (similar to an Excel spreadsheet or SQL table). You can create one from a variety of sources, including lists, dictionaries, or external files (like CSV or Excel).

From a Dictionary:
python
```
data = {'Name': ['Alice', 'Bob', 'Charlie'],
        'Age': [25, 30, 35],
        'City': ['New York', 'Los Angeles', 'Chicago']}

df = pd.DataFrame(data)
print(df)
```

From a CSV file:
python
```
df = pd.read_csv('path_to_file.csv')
```

4. **Exploring Data**
Pandas offers several ways to explore the data.

Check the first few rows:**
```
 python
 df.head()  # by default returns the first 5 rows
```

Check the data types and summary info:
```
 python
 df.info()
```

Statistical summary of numerical columns:
```
 python
 df.describe()
```

5. **Indexing and Selecting Data**
You can select data from a DataFrame in multiple ways.

- **By column name:**
  python
  df['Age']

- **By row index (using `iloc` for integer-location based indexing or `loc` for label-based):**
python
  df.iloc[0] # Select the first row by index
  df.loc[0]   # Select the first row by label (index)
  ```


- **Selecting multiple columns:**
  ```python
  df[['Name', 'City']]
  ```

6. **Filtering Data**
You can filter rows based on conditions.

```python
df[df['Age'] > 30]
```

7. **Handling Missing Data**
You can handle missing data in several ways:

- **Detect missing data:**
python
  df.isnull()

Drop rows with missing data:
  python
  df.dropna()

- **Fill missing data with a value:**
  python
  df.fillna(0) # Replace NaN with 0

8. **Data Transformation**
Pandas makes it easy to manipulate data.

-**Adding a new column:**
 python
  df['Country'] = 'USA'
```

- **Applying functions to columns:**
python
  df['Age'] = df['Age'].apply(lambda x: x + 1) # Adds 1 to each value in the Age column

- **Renaming columns:**
  python
  df.rename(columns={'Name': 'Full Name'}, inplace=True)

9. **Grouping Data**
You can group data and perform aggregations using the `groupby` method.

python
df.groupby('City')['Age'].mean() # Calculate the average age per city

10. **Merging DataFrames**
If you have multiple DataFrames that you want to combine, you can use `merge()`:

python
df1 = pd.DataFrame({'ID': [1, 2, 3], 'Name': ['Alice', 'Bob', 'Charlie']})
df2 = pd.DataFrame({'ID': [1, 2, 3], 'Age': [25, 30, 35]})

merged_df = pd.merge(df1, df2, on='ID')