```
import numpy as np
import pandas as pd
```

Series is 1D and DataFrames are 2D objects

- But why?
- And what exactly is index?

```
# series is 1D -- > qki series me kisi data ko fetch karne ke lye only one info ki need hoti hai
# dataframe is 2D -- > qki df me kisi data ko fetch karne ke lye at least two info ki need hoti hai

# can we have multiple index? Let's try
index_val = [('cse',2019),('cse',2020),('cse',2021),('cse',2022),
('ece',2019),('ece',2020),('ece',2021),('ece',2022)]
a = pd.Series([1,2,3,4,5,6,7,8],index=index_val)  # abb jo index bana hai wo do ka combination hai
a
```

```
(cse, 2019)    1
(cse, 2020)    2
(cse, 2021)    3
(cse, 2022)    4
(ece, 2019)    5
(ece, 2020)    6
(ece, 2021)    7
(ece, 2022)    8
dtype: int64
```

```
# now fetch some data
a[('cse', 2020)]
```

```
2
```

```
# the problem in this approach
# maan lo mujhe unn sare value ko fetch karna hai jaha pe cse hai
a['cse']   # ye error dega qki proper cse name ka koi index hi nahi hai to chalo ek another method ke multi indexing karte hain
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File ~\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3805, in Index.get_loc(self, key)
   3804 try:
-> 3805     return self._engine.get_loc(casted_key)
   3806 except KeyError as err:

File index.pyx:167, in pandas._libs.index.IndexEngine.get_loc()
```

```
File index.pyx:196, in pandas._libs.index.IndexEngine.get_loc()

File pandas\\_libs\\hashtable_class_helper.pxi:7081, in
pandas._libs.hashtable.PyObjectHashTable.get_item()

File pandas\\_libs\\hashtable_class_helper.pxi:7089, in
pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: 'cse'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call
last)
Cell In[11], line 3
      1 # the problem in this approach
      2 # maan lo mujhe unn sare value ko fetch karna hai jaha pe cse
hai
----> 3 a['cse']

File ~\anaconda3\Lib\site-packages\pandas\core\series.py:1121, in
Series.__getitem__(self, key)
   1118     return self._values[key]
   1120 elif key_is_scalar:
-> 1121     return self._get_value(key)
   1123 # Convert generator to list before going through hashable part
   1124 # (We will iterate through the generator there to check for
slices)
   1125 if is_iterator(key):

File ~\anaconda3\Lib\site-packages\pandas\core\series.py:1237, in
Series._get_value(self, label, takeable)
   1234     return self._values[label]
   1236 # Similar to Index.get_value, but we do not fall back to
positional
-> 1237 loc = self.index.get_loc(label)
   1239 if is_integer(loc):
   1240     return self._values[loc]

File ~\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3812,
in Index.get_loc(self, key)
   3807     if isinstance(casted_key, slice) or (
   3808         isinstance(casted_key, abc.Iterable)
   3809         and any(isinstance(x, slice) for x in casted_key)
   3810     ):
   3811         raise InvalidIndexError(key)
-> 3812     raise KeyError(key) from err
   3813 except TypeError:
   3814     # If we have a listlike key, _check_indexing_error will
```

```
raise
   3815     #  InvalidIndexError. Otherwise we fall through and re-
raise
   3816     #  the TypeError.
   3817     self._check_indexing_error(key)

KeyError: 'cse'
```

```
# The solution -> multiindex series(also known as Hierarchical
Indexing)
# multiple index levels within a single index
# multiindexing ke lye multiindex object create karo
# how to create multiindex object

# 1. pd.MultiIndex.from_tuples()
index_val = [('cse',2019),('cse',2020),('cse',2021),('cse',2022),
('ece',2019),('ece',2020),('ece',2021),('ece',2022)]
multiindex = pd.MultiIndex.from_tuples(index_val)
multiindex
```

```
MultiIndex([('cse', 2019),
            ('cse', 2020),
            ('cse', 2021),
            ('cse', 2022),
            ('ece', 2019),
            ('ece', 2020),
            ('ece', 2021),
            ('ece', 2022)],
           )
```

```
multiindex.levels   # ye first column bale index ko alag and second
column bale ko alag index bana dega ofcourse ye dono aapas me grouped
rahega
```

```
FrozenList([['cse', 'ece'], [2019, 2020, 2021, 2022]])
```

```
multiindex.levels[0]
```

```
Index(['cse', 'ece'], dtype='object')
```

```
multiindex.levels[1]
```

```
Index([2019, 2020, 2021, 2022], dtype='int64')
```

```
# 1. pd.MultiIndex.from_product()
pd.MultiIndex.from_product([['cse','ece'],[2019,2020,2021,2022]])   #
ye bhi same result dega and it will eork simillar as cartesian product
```

```
MultiIndex([('cse', 2019),
            ('cse', 2020),
            ('cse', 2021),
            ('cse', 2022),
```

```
            ('ece', 2019),
            ('ece', 2020),
            ('ece', 2021),
            ('ece', 2022)],
           )

# creating a series with multiindex object
s = pd.Series([1,2,3,4,5,6,7,8],index=multiindex)
s
```

```
cse  2019    1
     2020    2
     2021    3
     2022    4
ece  2019    5
     2020    6
     2021    7
     2022    8
dtype: int64
```

```
# how to fetch items from such a series
s['cse']   # abb jitne me ibdex cse hai wo print ho jayega
```

```
2019    1
2020    2
2021    3
2022    4
dtype: int64
```

```
s['cse',2019]
```

```
1
```

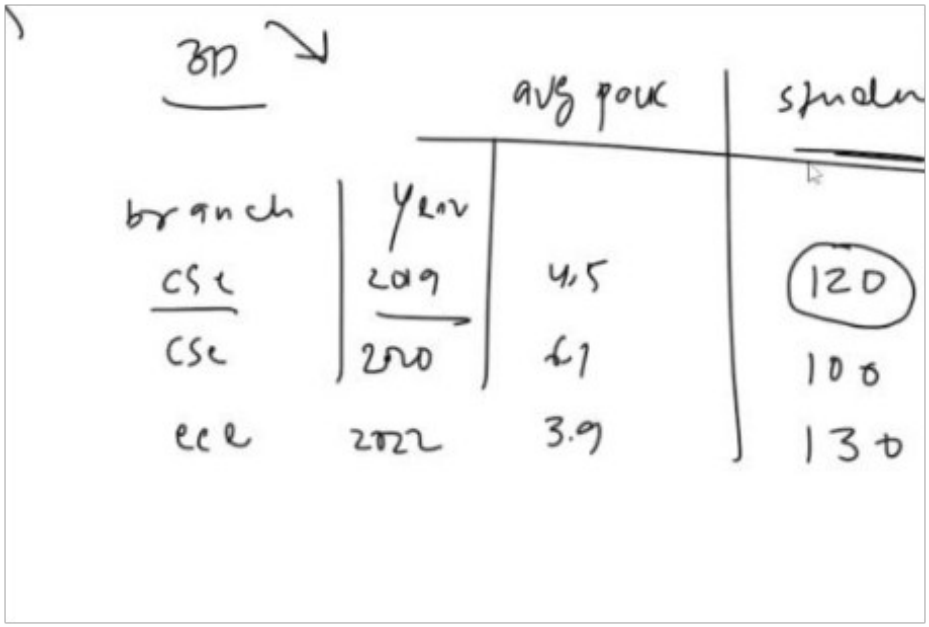
```
# a logical question -- > ki multiidex series 1D hai yaa 2D to ye 2D
hai qki kisi particularr value ko fetch karne ke lye hume 2 info ki
need hoti hai
# one question is arise ki phir to hum ye kaam dataframe se bhi kar
sakte the then why series

# unstack   => unstack ek multiindex series ko dataframe me convert
kar deta hai and pahla index pahle index ke taraah and andar ka index
                  # as a column name behave karne lagta hai
temp = s.unstack()
temp
```

```
     2019  2020  2021  2022
cse     1     2     3     4
ece     5     6     7     8
```

```
# stack          => ye dataframe ko multiindex series me concert kar deta hai just opposite of unstack
temp.stack()
```

```
cse  2019    1
     2020    2
     2021    3
     2022    4
ece  2019    5
     2020    6
     2021    7
     2022    8
dtype: int64
```

```
# but question abhi tak wahi hai jab hum same kaaam df se kar sakte hain the why we need of mulyiindexing series

# qki hum multidimension(heigher dimension) data to lower dimension data me convert kar sake

# multiindex dataframe
branch_df1 = pd.DataFrame(
    [
        [1,2],
        [3,4],
        [5,6],
        [7,8],
        [9,10],
        [11,12],
        [13,14],
        [15,16],
    ],
    index = multiindex,
    columns = ['avg_package','students']
)
```




```
# ye ek 3D data hai qki kisi bhi specific value ko find karne ke lye at least 3 info chahaiye but isse hum 2D me convert kae sakte hain
```

```
branch_df1
```

```
          avg_package  students
cse 2019            1         2
    2020            3         4
    2021            5         6
    2022            7         8
ece 2019            9        10
    2020           11        12
    2021           13        14
    2022           15        16
```

```
branch_df1.loc['cse']
```

```
      avg_package  students
2019            1         2
2020            3         4
2021            5         6
2022            7         8
```

```
branch_df1.loc['ece']
```

```
      avg_package  students
2019            9        10
2020           11        12
2021           13        14
2022           15        16
```

```
branch_df1['avg_package']
```

```
cse  2019     1
     2020     3
     2021     5
     2022     7
ece  2019     9
     2020    11
     2021    13
     2022    15
Name: avg_package, dtype: int64
```

```
# abhi tak ki jo learning hai usme ye baat pata chalta hai ki index alag chize hain and column alag chize hain but agar implementation level pe baat
# karo to pandas isse alag alag consider hi nahi karta qki agar pure data ka transpose karo to column index and index column ban jayega isse
# ye matlab nikalta hai ki hum column me bhi multi-indexing kar sakte hain jaha pe ki column me herarichy ho

# multiindex df from columns perspective
branch_df2 = pd.DataFrame(
    [
        [1,2,0,0],
        [3,4,0,0],
        [5,6,0,0],
        [7,8,0,0],
    ],
    index = [2019,2020,2021,2022],
    columns = pd.MultiIndex.from_product([['delhi','mumbai'],
['avg_package','students']])
)

branch_df2
```

	delhi		mumbai	
	avg_package	students	avg_package	students
2019	1	2	0	0
2020	3	4	0	0
2021	5	6	0	0
2022	7	8	0	0

```
branch_df2['delhi']
```

	avg_package	students
2019	1	2
2020	3	4
2021	5	6
2022	7	8

```
branch_df2['mumbai']['avg_package']
```

```
2019    0
2020    0
2021    0
2022    0
Name: avg_package, dtype: int64
```

```
branch_df2.loc[2019]
```

```
delhi   avg_package    1
        students       2
mumbai  avg_package    0
        students       0
Name: 2019, dtype: int64
```

```
# Multiindex df in terms of both cols and index

branch_df3 = pd.DataFrame(
    [
        [1,2,0,0],
        [3,4,0,0],
        [5,6,0,0],
        [7,8,0,0],
        [9,10,0,0],
        [11,12,0,0],
        [13,14,0,0],
        [15,16,0,0],
    ],
    index = multiindex,
    columns = pd.MultiIndex.from_product([['delhi','mumbai'],
['avg_package','students']])
)
```

```
branch_df3
```

		delhi		mumbai	
		avg_package	students	avg_package	students
cse	2019	1	2	0	0
	2020	3	4	0	0
	2021	5	6	0	0
	2022	7	8	0	0
ece	2019	9	10	0	0
	2020	11	12	0	0
	2021	13	14	0	0
	2022	15	16	0	0

```
# this above data is 4D hai  qki koi bhi value fatch karne ke lye at
least 4 peice of info ki need hogi . 1. ki kon se branch ka student
hai
# 2. kon se year ka , 3. kon se city ka , 4. kya nikalna hai
avg_package yaaa students
```

Stacking and Unstacking

```
branch_df1
```

```
          avg_package  students
cse 2019            1         2
    2020            3         4
    2021            5         6
    2022            7         8
ece 2019            9        10
    2020           11        12
    2021           13        14
    2022           15        16
```

```
branch_df1.unstack()  # andar bala index(year) column me convet ho jayega lekin existing column khatam nahi hoga balki yaha pe column me                                  #multiindexing hoga   ye abhi bhi 3D data hi hai suru me bhi 3D hi tha
```

```
    avg_package                 students
           2019 2020 2021 2022     2019 2020 2021 2022
cse           1    3    5    7        2    4    6    8
ece           9   11   13   15       10   12   14   16
```

```
branch_df1.unstack().unstack()   # abb ek level ki indexing phir se ho jayega jo row me index tha wo again column ban jayega and abb koi row bacha
#  nahi hai to ye series me convert ho jayega
```

```
avg_package  2019  cse     1
                   ece     9
             2020  cse     3
                   ece    11
             2021  cse     5
                   ece    13
             2022  cse     7
                   ece    15
students     2019  cse     2
                   ece    10
             2020  cse     4
                   ece    12
             2021  cse     6
                   ece    14
             2022  cse     8
                   ece    16
dtype: int64
```

```
branch_df1.unstack().unstack().stack()   # series me stack nahi hota hai
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_1788\2260072893.py in ?()
----> 1 branch_df1.unstack().unstack().stack()   # series me stack nahi hota hai

~\anaconda3\Lib\site-packages\pandas\core\generic.py in ?(self, name)
   6295             and name not in self._accessors
   6296             and self._info_axis._can_hold_identifiers_and_holds_name(name)
   6297         ):
   6298             return self[name]
-> 6299         return object.__getattribute__(self, name)

AttributeError: 'Series' object has no attribute 'stack'
```

```
branch_df1.unstack().stack()
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_1788\1991893145.py:1: FutureWarning: The previous implementation of stack is deprecated and will be removed in a future version of pandas. See the What's New notes for pandas 2.1.0 for details. Specify future_stack=True to adopt the new implementation and silence this warning.
  branch_df1.unstack().stack()
```

		avg_package	students
cse	2019	1	2
	2020	3	4
	2021	5	6
	2022	7	8
ece	2019	9	10
	2020	11	12
	2021	13	14
	2022	15	16

```
branch_df1.unstack().stack().stack()  # ye phir se ek series ban jayega qki sare column khatam ho gya
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_1788\2205799161.py:1: FutureWarning: The previous implementation of stack is deprecated and will be removed in a future version of pandas. See the What's New notes for pandas 2.1.0 for details. Specify future_stack=True to adopt the new implementation and silence this warning.
  branch_df1.unstack().stack().stack()  # ye phir se ek series ban jayega qki sare column khatam ho gya
```

```
cse  2019  avg_package    1
           students       2
     2020  avg_package    3
```

```
            students        4
     2021   avg_package     5
            students        6
     2022   avg_package     7
            students        8
ece  2019   avg_package     9
            students       10
     2020   avg_package    11
            students       12
     2021   avg_package    13
            students       14
     2022   avg_package    15
            students       16
dtype: int64
```

In Short

- unstack() --> andar bale row(index) ko andar ka column bana deta hai
- stack() --> andar bale column ko andar ka row bana deta hai
- basically reshape kar rahe hain stack and unstack ke help se

```
branch_df2
```

```
           delhi                mumbai
     avg_package students avg_package students
2019           1        2           0        0
2020           3        4           0        0
2021           5        6           0        0
2022           7        8           0        0
```

```
branch_df2.unstack()
```

```
delhi   avg_package  2019    1
                     2020    3
                     2021    5
                     2022    7
        students     2019    2
                     2020    4
                     2021    6
                     2022    8
mumbai  avg_package  2019    0
                     2020    0
                     2021    0
                     2022    0
        students     2019    0
                     2020    0
                     2021    0
```

```
                      2022    0
dtype: int64
```

```
branch_df2.stack()
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_1788\3132666484.py:1:
FutureWarning: The previous implementation of stack is deprecated and
will be removed in a future version of pandas. See the What's New
notes for pandas 2.1.0 for details. Specify future_stack=True to adopt
the new implementation and silence this warning.
  branch_df2.stack()
```

```
                   delhi  mumbai
2019 avg_package       1       0
     students          2       0
2020 avg_package       3       0
     students          4       0
2021 avg_package       5       0
     students          6       0
2022 avg_package       7       0
     students          8       0
```

```
branch_df2.stack().stack()
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_1788\2534568903.py:1:
FutureWarning: The previous implementation of stack is deprecated and
will be removed in a future version of pandas. See the What's New
notes for pandas 2.1.0 for details. Specify future_stack=True to adopt
the new implementation and silence this warning.
  branch_df2.stack().stack()
```

```
2019  avg_package  delhi     1
                   mumbai    0
      students     delhi     2
                   mumbai    0
2020  avg_package  delhi     3
                   mumbai    0
      students     delhi     4
                   mumbai    0
2021  avg_package  delhi     5
                   mumbai    0
      students     delhi     6
                   mumbai    0
2022  avg_package  delhi     7
                   mumbai    0
      students     delhi     8
                   mumbai    0
dtype: int64
```

```
branch_df3
```

```
                delhi                mumbai
          avg_package students avg_package students
cse 2019            1        2           0        0
    2020            3        4           0        0
    2021            5        6           0        0
    2022            7        8           0        0
ece 2019            9       10           0        0
    2020           11       12           0        0
    2021           13       14           0        0
    2022           15       16           0        0
```

```
branch_df3.unstack()
```

```
           delhi                                                  mumbai
\
    avg_package                  students                  avg_package

            2019 2020 2021 2022     2019 2020 2021 2022         2019
2020 2021
cse            1    3    5    7        2    4    6    8            0
0    0
ece            9   11   13   15       10   12   14   16            0
0    0


          students
    2022     2019 2020 2021 2022
cse    0        0    0    0    0
ece    0        0    0    0    0
```

```
branch_df3.unstack().unstack()   # series me convert ho jayega and ye
4D series hoga
```

```
delhi   avg_package  2019  cse     1
                           ece     9
                      2020  cse     3
                           ece    11
                      2021  cse     5
                           ece    13
                      2022  cse     7
                           ece    15
        students      2019  cse     2
                           ece    10
                      2020  cse     4
                           ece    12
                      2021  cse     6
                           ece    14
                      2022  cse     8
                           ece    16
```

```
mumbai  avg_package  2019  cse     0
                           ece     0
                     2020  cse     0
                           ece     0
                     2021  cse     0
                           ece     0
                     2022  cse     0
                           ece     0
        students     2019  cse     0
                           ece     0
                     2020  cse     0
                           ece     0
                     2021  cse     0
                           ece     0
                     2022  cse     0
                           ece     0
dtype: int64
```

```
branch_df3.stack()
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_1788\4148153360.py:1:
FutureWarning: The previous implementation of stack is deprecated and
will be removed in a future version of pandas. See the What's New
notes for pandas 2.1.0 for details. Specify future_stack=True to adopt
the new implementation and silence this warning.
  branch_df3.stack()
```

```
                       delhi  mumbai
cse 2019 avg_package       1       0
         students          2       0
    2020 avg_package       3       0
         students          4       0
    2021 avg_package       5       0
         students          6       0
    2022 avg_package       7       0
         students          8       0
ece 2019 avg_package       9       0
         students         10       0
    2020 avg_package      11       0
         students         12       0
    2021 avg_package      13       0
         students         14       0
    2022 avg_package      15       0
         students         16       0
```

```
branch_df3.stack().stack()
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_1788\4023844418.py:1:
FutureWarning: The previous implementation of stack is deprecated and
will be removed in a future version of pandas. See the What's New
```

```
notes for pandas 2.1.0 for details. Specify future_stack=True to adopt
the new implementation and silence this warning.
  branch_df3.stack().stack()

cse  2019  avg_package  delhi      1
                        mumbai     0
           students     delhi      2
                        mumbai     0
     2020  avg_package  delhi      3
                        mumbai     0
           students     delhi      4
                        mumbai     0
     2021  avg_package  delhi      5
                        mumbai     0
           students     delhi      6
                        mumbai     0
     2022  avg_package  delhi      7
                        mumbai     0
           students     delhi      8
                        mumbai     0
ece  2019  avg_package  delhi      9
                        mumbai     0
           students     delhi     10
                        mumbai     0
     2020  avg_package  delhi     11
                        mumbai     0
           students     delhi     12
                        mumbai     0
     2021  avg_package  delhi     13
                        mumbai     0
           students     delhi     14
                        mumbai     0
     2022  avg_package  delhi     15
                        mumbai     0
           students     delhi     16
                        mumbai     0
dtype: int64
```

Working with multiindex dataframes

```
# head and tail
branch_df3.head()
```

		delhi		mumbai	
		avg_package	students	avg_package	students
cse	2019	1	2	0	0
	2020	3	4	0	0
	2021	5	6	0	0
	2022	7	8	0	0
ece	2019	9	10	0	0

```
branch_df3.tail()
```

```
               delhi                mumbai
         avg_package students avg_package students
cse 2022           7        8           0        0
ece 2019           9       10           0        0
    2020          11       12           0        0
    2021          13       14           0        0
    2022          15       16           0        0
```

```
# shape
branch_df3.shape
```

```
(8, 4)
```

```
# info
branch_df3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
MultiIndex: 8 entries, ('cse', 2019) to ('ece', 2022)
Data columns (total 4 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   (delhi, avg_package)   8 non-null      int64
 1   (delhi, students)      8 non-null      int64
 2   (mumbai, avg_package)  8 non-null      int64
 3   (mumbai, students)     8 non-null      int64
dtypes: int64(4)
memory usage: 932.0+ bytes
```

```
branch_df3.unstack().info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 2 entries, cse to ece
Data columns (total 16 columns):
 #   Column                         Non-Null Count  Dtype
---  ------                         --------------  -----
 0   (delhi, avg_package, 2019)   2 non-null      int64
 1   (delhi, avg_package, 2020)   2 non-null      int64
 2   (delhi, avg_package, 2021)   2 non-null      int64
 3   (delhi, avg_package, 2022)   2 non-null      int64
 4   (delhi, students, 2019)      2 non-null      int64
 5   (delhi, students, 2020)      2 non-null      int64
 6   (delhi, students, 2021)      2 non-null      int64
 7   (delhi, students, 2022)      2 non-null      int64
 8   (mumbai, avg_package, 2019)  2 non-null      int64
 9   (mumbai, avg_package, 2020)  2 non-null      int64
 10  (mumbai, avg_package, 2021)  2 non-null      int64
 11  (mumbai, avg_package, 2022)  2 non-null      int64
 12  (mumbai, students, 2019)     2 non-null      int64
 13  (mumbai, students, 2020)     2 non-null      int64
```

```
 14  (mumbai, students, 2021)     2 non-null      int64
 15  (mumbai, students, 2022)     2 non-null      int64
dtypes: int64(16)
memory usage: 272.0+ bytes
```

```
# duplicated -> isnull
branch_df3.duplicated()
```

```
cse  2019    False
     2020    False
     2021    False
     2022    False
ece  2019    False
     2020    False
     2021    False
     2022    False
dtype: bool
```

```
branch_df3.isnull()
```

		delhi		mumbai	
		avg_package	students	avg_package	students
cse	2019	False	False	False	False
	2020	False	False	False	False
	2021	False	False	False	False
	2022	False	False	False	False
ece	2019	False	False	False	False
	2020	False	False	False	False
	2021	False	False	False	False
	2022	False	False	False	False

```
#### ho jo bhi normal df ke sath karte hain wo sab kuch yaha bhi kar sakte hain

# Extract rows single
branch_df3
```

		delhi		mumbai	
		avg_package	students	avg_package	students
cse	2019	1	2	0	0
	2020	3	4	0	0
	2021	5	6	0	0
	2022	7	8	0	0
ece	2019	9	10	0	0
	2020	11	12	0	0
	2021	13	14	0	0
	2022	15	16	0	0

```
branch_df3.loc[('cse',2022)]
```

```
delhi   avg_package    7
        students       8
mumbai  avg_package    0
        students       0
Name: (cse, 2022), dtype: int64
```

```
# multiple
# 1st, 3rd and 5th row nikal
branch_df3[('cse',2019):('ece',2020):2]
```

```
               delhi               mumbai
         avg_package students avg_package students
cse 2019           1        2           0        0
    2021           5        6           0        0
ece 2019           9       10           0        0
```

```
# using iloc
branch_df3.iloc[0]   # ye internal bala index ko pakad ke kaam karta
hai
```

```
delhi   avg_package    1
        students       2
mumbai  avg_package    0
        students       0
Name: (cse, 2019), dtype: int64
```

```
branch_df3.iloc[0:5:2]
```

```
               delhi               mumbai
         avg_package students avg_package students
cse 2019           1        2           0        0
    2021           5        6           0        0
ece 2019           9       10           0        0
```

```
 # extracting cols
branch_df3['delhi']
```

```
          avg_package  students
cse 2019            1         2
    2020            3         4
    2021            5         6
    2022            7         8
ece 2019            9        10
    2020           11        12
    2021           13        14
    2022           15        16
```

```
branch_df3['delhi']['students']
```

```
cse  2019     2
     2020     4
```

```
     2021     6
     2022     8
ece  2019    10
     2020    12
     2021    14
     2022    16
Name: students, dtype: int64
```

```
# multiple cols
branch_df3
```

		delhi		mumbai	
		avg_package	students	avg_package	students
cse	2019	1	2	0	0
	2020	3	4	0	0
	2021	5	6	0	0
	2022	7	8	0	0
ece	2019	9	10	0	0
	2020	11	12	0	0
	2021	13	14	0	0
	2022	15	16	0	0

```
# hume delhi ka student and mumbai ka avg_package nikalna hai
branch_df3.iloc[:,1:3]
```

		delhi	mumbai
		students	avg_package
cse	2019	2	0
	2020	4	0
	2021	6	0
	2022	8	0
ece	2019	10	0
	2020	12	0
	2021	14	0
	2022	16	0

```
# extracting both (row, cols)

branch_df3
```

		delhi		mumbai	
		avg_package	students	avg_package	students
cse	2019	1	2	0	0
	2020	3	4	0	0
	2021	5	6	0	0
	2022	7	8	0	0
ece	2019	9	10	0	0
	2020	11	12	0	0
	2021	13	14	0	0
	2022	15	16	0	0

```
# suppose cse ka 1st row and ece ka 1st row chahaiye and delhi ka
student and mumbai ka avg_package bala column chahaiye
branch_df3.iloc[[0,4],[1,2]]
```

```
            delhi      mumbai
         students avg_package
cse 2019        2           0
ece 2019       10           0
```

```
# sort index
# both -> descending -> diff order
# based on one level
branch_df3.sort_index(ascending=False)  # dono level ke index pe
sorting ho jayega
```

```
               delhi                mumbai
         avg_package students avg_package students
ece 2022          15       16           0        0
    2021          13       14           0        0
    2020          11       12           0        0
    2019           9       10           0        0
cse 2022           7        8           0        0
    2021           5        6           0        0
    2020           3        4           0        0
    2019           1        2           0        0
```

```
# suppose branch decending order me and year ascending order me
chahaiye matbal ki alag alag level pe different sorting
branch_df3.sort_index(ascending = [False,True]) # matlab ki level 0 ko
decending and level 1  ascending
```

```
               delhi                mumbai
         avg_package students avg_package students
ece 2019           9       10           0        0
    2020          11       12           0        0
    2021          13       14           0        0
    2022          15       16           0        0
cse 2019           1        2           0        0
    2020           3        4           0        0
    2021           5        6           0        0
    2022           7        8           0        0
```

```
# what iff only one level pe sorting karna ho abhi tak dono level pe
sorting ho raha hai
# suppose sirf year pe sirf sorting karna hai
branch_df3.sort_index(level=1,ascending=[False])     #level column
position ko denote karta hai ki kon se column ke according ke sort
                                                    #karna hai suru 0 se
hota hai islye yaha level me 1 dale hain
```

		delhi		mumbai	
		avg_package	students	avg_package	students
cse	2022	7	8	0	0
ece	2022	15	16	0	0
cse	2021	5	6	0	0
ece	2021	13	14	0	0
cse	2020	3	4	0	0
ece	2020	11	12	0	0
cse	2019	1	2	0	0
ece	2019	9	10	0	0

```
branch_df3.sort_index(level=0,ascending=[False])
```

		delhi		mumbai	
		avg_package	students	avg_package	students
ece	2019	9	10	0	0
	2020	11	12	0	0
	2021	13	14	0	0
	2022	15	16	0	0
cse	2019	1	2	0	0
	2020	3	4	0	0
	2021	5	6	0	0
	2022	7	8	0	0

```
# multiindex dataframe(col)   ---> transpose
branch_df3
```

		delhi		mumbai	
		avg_package	students	avg_package	students
cse	2019	1	2	0	0
	2020	3	4	0	0
	2021	5	6	0	0
	2022	7	8	0	0
ece	2019	9	10	0	0
	2020	11	12	0	0
	2021	13	14	0	0
	2022	15	16	0	0

```
branch_df3.transpose()
```

		cse				ece			
		2019	2020	2021	2022	2019	2020	2021	2022
delhi	avg_package	1	3	5	7	9	11	13	15
	students	2	4	6	8	10	12	14	16
mumbai	avg_package	0	0	0	0	0	0	0	0
	students	0	0	0	0	0	0	0	0

```
# swaplevel
# level ko change kar sakte hain
branch_df3
```

```
               delhi                mumbai
         avg_package students avg_package students
cse 2019           1        2           0        0
    2020           3        4           0        0
    2021           5        6           0        0
    2022           7        8           0        0
ece 2019           9       10           0        0
    2020          11       12           0        0
    2021          13       14           0        0
    2022          15       16           0        0
```

```
branch_df3.swaplevel()
```

```
               delhi                mumbai
         avg_package students avg_package students
2019 cse           1        2           0        0
2020 cse           3        4           0        0
2021 cse           5        6           0        0
2022 cse           7        8           0        0
2019 ece           9       10           0        0
2020 ece          11       12           0        0
2021 ece          13       14           0        0
2022 ece          15       16           0        0
```

```
# hum column ke sath bhi swap kar sakte hain
branch_df3.swaplevel(axis=1)
```

```
         avg_package students avg_package students
               delhi    delhi      mumbai   mumbai
cse 2019           1        2           0        0
    2020           3        4           0        0
    2021           5        6           0        0
    2022           7        8           0        0
ece 2019           9       10           0        0
    2020          11       12           0        0
    2021          13       14           0        0
    2022          15       16           0        0
```

Long Vs Wide Data

Name	Height	Weight
John	160	67
Christopher	182	78

Name	Attribute	Value
John	Height	160
John	Weight	67
Christopher	Height	182
Christopher	Weight	78

Wide format is where we have a single row for every data point with multiple columns to hold the values of various attributes.

Long format is where, for each data point we have as many rows as the number of attributes and each row contains the value of a particular attribute for a given data point.

```
# jab dono data set same information deta hai to why we have different
way to organise to ye islye qki kisi spacific problem ke lye long data
use
# ho sakta hai and kisi ke lye wide data
```

melt and pivot

- melt => wide data format ko long me convert karta hai
- pivot => long data format ko wide me convert karta hai

```
# melt   => wide to long
pd.DataFrame({'cse':[120]})

   cse
0  120

pd.DataFrame({'cse':[120]}).melt()

  variable  value
0      cse    120

pd.DataFrame({'cse':[120],'ece':[100],'mech':[50]}) # ye ek wide data
hai

   cse  ece  mech
0  120  100    50
```

```
pd.DataFrame({'cse':[120],'ece':[100],'mech':[50]}).melt()  # ye ek long me converted data hai
```

```
  variable  value
0      cse    120
1      ece    100
2     mech     50
```

```
# yaha pe apne aap  column ka kuch name aa gya hai aap isse apne according change kar sakte hain
pd.DataFrame({'cse':[120],'ece':[100],'mech':
[50]}).melt(var_name='branch',value_name='num_students')
```

```
  branch  num_students
0    cse           120
1    ece           100
2   mech            50
```

```
df=pd.DataFrame(
    {
        'branch':['cse','ece','mech'],
        '2020':[100,150,60],
        '2021':[120,130,80],
        '2022':[150,140,70]
    }
)
df
```

```
  branch  2020  2021  2022
0    cse   100   120   150
1    ece   150   130   140
2   mech    60    80    70
```

```
df.melt()  # ye data to kuvh ajib sa ho gya yee islye hua qki hume sare column ko row banane ki need nahi thi hum 2020,2021 and 2022 ko to row banana chahate
           #hain but brancgh bale column ko convert nahi karmna chahate hain
```

```
   variable value
0    branch   cse
1    branch   ece
2    branch  mech
3      2020   100
4      2020   150
5      2020    60
6      2021   120
7      2021   130
8      2021    80
9      2022   150
```

```
10     2022   140
11     2022    70
```

```
df.melt(id_vars=['branch'])    # id_vars me jo daat pass karte hain wo convert nahi hota hai
```

```
  branch variable  value
0    cse     2020    100
1    ece     2020    150
2   mech     2020     60
3    cse     2021    120
4    ece     2021    130
5   mech     2021     80
6    cse     2022    150
7    ece     2022    140
8   mech     2022     70
```

```
df.melt(id_vars=['branch'],var_name='year',value_name='students')
```

```
  branch  year  students
0    cse  2020       100
1    ece  2020       150
2   mech  2020        60
3    cse  2021       120
4    ece  2021       130
5   mech  2021        80
6    cse  2022       150
7    ece  2022       140
8   mech  2022        70
```

```
#  melt ---> real world example
death = pd.read_csv('time_series_covid19_deaths_global.csv')
death.head()
```

```
  Province/State Country/Region       Lat       Long  1/22/20  1/23/20
\
0            NaN    Afghanistan  33.93911  67.709953        0        0

1            NaN        Albania  41.15330  20.168300        0        0

2            NaN        Algeria  28.03390   1.659600        0        0

3            NaN        Andorra  42.50630   1.521800        0        0

4            NaN         Angola -11.20270  17.873900        0        0


   1/24/20  1/25/20  1/26/20  1/27/20  ...  12/24/22  12/25/22
12/26/22  \
0        0        0        0        0  ...      7845      7846
7846
```

```
1        0        0        0        0  ...      3595      3595
3595
2        0        0        0        0  ...      6881      6881
6881
3        0        0        0        0  ...       165       165
165
4        0        0        0        0  ...      1928      1928
1928

   12/27/22  12/28/22  12/29/22  12/30/22  12/31/22  1/1/23  1/2/23
0      7846      7846      7847      7847      7849    7849    7849
1      3595      3595      3595      3595      3595    3595    3595
2      6881      6881      6881      6881      6881    6881    6881
3       165       165       165       165       165     165     165
4      1930      1930      1930      1930      1930    1930    1930

[5 rows x 1081 columns]
```

```
confirm = pd.read_csv('time_series_covid19_confirmed_global.csv')
confirm.head()
```

```
  Province/State Country/Region       Lat       Long  1/22/20  1/23/20
\
0            NaN    Afghanistan  33.93911  67.709953        0        0

1            NaN        Albania  41.15330  20.168300        0        0

2            NaN        Algeria  28.03390   1.659600        0        0

3            NaN        Andorra  42.50630   1.521800        0        0

4            NaN         Angola -11.20270  17.873900        0        0


   1/24/20  1/25/20  1/26/20  1/27/20  ...  12/24/22  12/25/22
12/26/22  \
0        0        0        0        0  ...    207310    207399
207438
1        0        0        0        0  ...    333749    333749
333751
2        0        0        0        0  ...    271194    271198
271198
3        0        0        0        0  ...     47686     47686
47686
4        0        0        0        0  ...    104973    104973
104973

   12/27/22  12/28/22  12/29/22  12/30/22  12/31/22  1/1/23  1/2/23
0    207460    207493    207511    207550    207559  207616  207627
1    333751    333776    333776    333806    333806  333811  333812
2    271202    271208    271217    271223    271228  271229  271229
```

```
3     47686     47751     47751     47751     47751   47751   47751
4    105095    105095    105095    105095    105095  105095  105095

[5 rows x 1081 columns]
```

```
# suppose inn dono ko mila ke ek new data frame banana hai jisme ki 4
columns hoga jisme ki -> country,date,confirm,death bale column hoga
# steps =>pahle melt karenge and the merge karenge
death=death.melt(id_vars=['Province/State','Country/Region','Lat','Lon
g'],var_name='date',value_name='num_death')
death
```

```
       Province/State        Country/Region        Lat         Long
date  \
0                 NaN           Afghanistan  33.939110   67.709953
1/22/20
1                 NaN               Albania  41.153300   20.168300
1/22/20
2                 NaN               Algeria  28.033900    1.659600
1/22/20
3                 NaN               Andorra  42.506300    1.521800
1/22/20
4                 NaN                Angola -11.202700   17.873900
1/22/20
...               ...                   ...        ...         ...
...
311248            NaN    West Bank and Gaza  31.952200   35.233200
1/2/23
311249            NaN  Winter Olympics 2022  39.904200  116.407400
1/2/23
311250            NaN                 Yemen  15.552727   48.516388
1/2/23
311251            NaN                Zambia -13.133897   27.849332
1/2/23
311252            NaN              Zimbabwe -19.015438   29.154857
1/2/23

        num_death
0               0
1               0
2               0
3               0
4               0
...           ...
311248       5708
311249          0
311250       2159
311251       4024
311252       5637
```

```
[311253 rows x 6 columns]
```

```
confirm=confirm.melt(id_vars=['Province/State','Country/
Region','Lat','Long'],var_name='date',value_name='num_cases')
confirm
```

```
       Province/State        Country/Region        Lat        Long
date  \
0                 NaN           Afghanistan  33.939110   67.709953
1/22/20
1                 NaN               Albania  41.153300   20.168300
1/22/20
2                 NaN               Algeria  28.033900    1.659600
1/22/20
3                 NaN               Andorra  42.506300    1.521800
1/22/20
4                 NaN                Angola -11.202700   17.873900
1/22/20
...               ...                   ...        ...         ...
...
311248            NaN    West Bank and Gaza  31.952200   35.233200
1/2/23
311249            NaN  Winter Olympics 2022  39.904200  116.407400
1/2/23
311250            NaN                 Yemen  15.552727   48.516388
1/2/23
311251            NaN                Zambia -13.133897   27.849332
1/2/23
311252            NaN              Zimbabwe -19.015438   29.154857
1/2/23

        num_cases
0               0
1               0
2               0
3               0
4               0
...           ...
311248     703228
311249        535
311250      11945
311251     334661
311252     259981

[311253 rows x 6 columns]
```

```
confirm.merge(death,on=['Province/State','Country/
Region','Lat','Long','date'])
```

```
       Province/State        Country/Region        Lat        Long
date  \
0                 NaN           Afghanistan  33.939110   67.709953
1/22/20
1                 NaN               Albania  41.153300   20.168300
1/22/20
2                 NaN               Algeria  28.033900    1.659600
1/22/20
3                 NaN               Andorra  42.506300    1.521800
1/22/20
4                 NaN                Angola -11.202700   17.873900
1/22/20
...               ...                   ...        ...         ...
...
311248            NaN    West Bank and Gaza  31.952200   35.233200
1/2/23
311249            NaN  Winter Olympics 2022  39.904200  116.407400
1/2/23
311250            NaN                 Yemen  15.552727   48.516388
1/2/23
311251            NaN                Zambia -13.133897   27.849332
1/2/23
311252            NaN              Zimbabwe -19.015438   29.154857
1/2/23

        num_cases  num_death
0               0          0
1               0          0
2               0          0
3               0          0
4               0          0
...           ...        ...
311248     703228       5708
311249        535          0
311250      11945       2159
311251     334661       4024
311252     259981       5637

[311253 rows x 7 columns]
```

```
confirm.merge(death,on=['Province/State','Country/
Region','Lat','Long','date'])[['Country/
Region','date','num_cases','num_death']] # required output
```

```
              Country/Region     date  num_cases  num_death
0               Afghanistan  1/22/20          0          0
1                   Albania  1/22/20          0          0
2                   Algeria  1/22/20          0          0
3                   Andorra  1/22/20          0          0
4                    Angola  1/22/20          0          0
```

```
...                         ...      ...        ...         ...
311248    West Bank and Gaza   1/2/23     703228        5708
311249  Winter Olympics 2022   1/2/23        535           0
311250                 Yemen   1/2/23      11945        2159
311251                Zambia   1/2/23     334661        4024
311252              Zimbabwe   1/2/23     259981        5637

[311253 rows x 4 columns]
```

Pivot Table

The pivot table takes simple column-wise data as input, and groups the entries into a two-dimensional table that provides a multidimensional summarization of the data.

- pivot table generally categorical column ke lye use karte hain

```
import numpy as np
import pandas as pd
import seaborn as sns



df = sns.load_dataset('tips')    # seaborn jo library hai usme inbuilt toy dataset available hota hai usme se ek hai tips(ye ek resturant ka data hai)
df.head()
```

```
   total_bill   tip     sex smoker  day    time  size
0       16.99  1.01  Female     No  Sun  Dinner     2
1       10.34  1.66    Male     No  Sun  Dinner     3
2       21.01  3.50    Male     No  Sun  Dinner     3
3       23.68  3.31    Male     No  Sun  Dinner     2
4       24.59  3.61  Female     No  Sun  Dinner     4
```

```
# we have to find gender ke hisab se total bill
# m-1
df.groupby('sex')[['total_bill']].mean()
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_5096\1049992168.py:3: FutureWarning: The default of observed=False is deprecated and will be changed to True in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence this warning.
  df.groupby('sex')[['total_bill']].mean()
```

```
        total_bill
sex
Male     20.744076
Female   18.056897
```

```
# abb hume janna hai ki smoker female jo hai wo kitna bill pay karti hai and jo non-smoker female hai wo kitna bill pay karti hai and also for men
df.groupby(['sex','smoker'])[['total_bill']].mean()   # ye abhi multiindex data hai we can unstack this
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_5096\3129696968.py:2: FutureWarning: The default of observed=False is deprecated and will be changed to True in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence this warning.
  df.groupby(['sex','smoker'])[['total_bill']].mean()   # ye abhi multiindex data hai we can unstack this
```

		total_bill
sex	smoker	
Male	Yes	22.284500
	No	19.791237
Female	Yes	17.977879
	No	18.105185

```
df.groupby(['sex','smoker'])[['total_bill']].mean().unstack()
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_5096\884363850.py:1: FutureWarning: The default of observed=False is deprecated and will be changed to True in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence this warning.
  df.groupby(['sex','smoker'])[['total_bill']].mean().unstack()
```

	total_bill	
smoker	Yes	No
sex		
Male	22.284500	19.791237
Female	17.977879	18.105185

```
# m-2 using pivot table   isme 3 chize batani hoti hai 1. index kon ban raha hai 2. column kon ban raha hai 3. kin values ke upar analysis karna hai

df.pivot_table(index='sex',columns='smoker',values='total_bill')   # ye ek shortcut way hai
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_5096\627329560.py:1: FutureWarning: The default value of observed=False is deprecated and will change to observed=True in a future version of pandas. Specify observed=False to silence this warning and retain the current behavior
  df.pivot_table(index='sex',columns='smoker',values='total_bill')   # ye ek shortcut way hai
```

```
smoker        Yes         No
sex
Male    22.284500  19.791237
Female  17.977879  18.105185
```

```
# by default aggrigate function me mean hota hai but we can change
accorrdingly
df.pivot_table(index='sex',columns='smoker',values='total_bill',aggfun
c='sum')
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_1788\2557101372.py:2:
FutureWarning: The default value of observed=False is deprecated and
will change to observed=True in a future version of pandas. Specify
observed=False to silence this warning and retain the current behavior

df.pivot_table(index='sex',columns='smoker',values='total_bill',aggfun
c='sum')
```

```
smoker      Yes       No
sex
Male    1337.07  1919.75
Female   593.27   977.68
```

```
df.pivot_table(index='sex',columns='smoker',values='total_bill',aggfun
c={'total_bill' :'sum'}) # another method of applying aggrigate
function
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_5096\2431434623.py:1:
FutureWarning: The default value of observed=False is deprecated and
will change to observed=True in a future version of pandas. Specify
observed=False to silence this warning and retain the current behavior

df.pivot_table(index='sex',columns='smoker',values='total_bill',aggfun
c={'total_bill' :'sum'})
```

```
smoker      Yes       No
sex
Male    1337.07  1919.75
Female   593.27   977.68
```

```
df.pivot_table(index='sex',columns='smoker',values='total_bill',aggfun
c='count')
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_1788\2608939217.py:1:
FutureWarning: The default value of observed=False is deprecated and
will change to observed=True in a future version of pandas. Specify
observed=False to silence this warning and retain the current behavior

df.pivot_table(index='sex',columns='smoker',values='total_bill',aggfun
c='count')
```

```
smoker  Yes  No
sex
Male     60  97
Female   33  54
```

```
# all columns together
df.pivot_table(index='sex',columns='smoker',aggfunc={'total_bill'  :
'mean','tip' :'mean' ,'size':'mean'})  # ye by default mean find karke
deta hai too
# jo bhi numerical columns hai uske lye mean specify karna hoga
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_5096\3885653735.py:2:
FutureWarning: The default value of observed=False is deprecated and
will change to observed=True in a future version of pandas. Specify
observed=False to silence this warning and retain the current behavior
  df.pivot_table(index='sex',columns='smoker',aggfunc={'total_bill'  :
'mean','tip' :'mean' ,'size':'mean'})
```

```
            size                 tip           total_bill
smoker       Yes        No       Yes        No        Yes         No
sex
Male    2.500000  2.711340  3.051167  3.113402  22.284500  19.791237
Female  2.242424  2.592593  2.931515  2.773519  17.977879  18.105185
```

```
# now you vcan extract single column according to your need
df.pivot_table(index='sex',columns='smoker',aggfunc={'total_bill'  :
'mean','tip' :'mean' ,'size':'mean'})['tip']
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_5096\1674464473.py:2:
FutureWarning: The default value of observed=False is deprecated and
will change to observed=True in a future version of pandas. Specify
observed=False to silence this warning and retain the current behavior
  df.pivot_table(index='sex',columns='smoker',aggfunc={'total_bill'  :
'mean','tip' :'mean' ,'size':'mean'})['tip']
```

```
smoker       Yes        No
sex
Male    3.051167  3.113402
Female  2.931515  2.773519
```

```
df.pivot_table(index='sex',columns='smoker',aggfunc={'total_bill'  :
'mean','tip' :'mean' ,'size':'mean'})['size']
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_5096\1336985347.py:1:
FutureWarning: The default value of observed=False is deprecated and
will change to observed=True in a future version of pandas. Specify
observed=False to silence this warning and retain the current behavior
  df.pivot_table(index='sex',columns='smoker',aggfunc={'total_bill'  :
'mean','tip' :'mean' ,'size':'mean'})['size']
```

```
smoker       Yes        No
sex
Male    2.500000  2.711340
Female  2.242424  2.592593
```

```
# multidimensional
df.pivot_table(index=['sex','smoker'],columns=['day','time'],values='t
otal_bill')
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_5096\3218821712.py:2:
FutureWarning: The default value of observed=False is deprecated and
will change to observed=True in a future version of pandas. Specify
observed=False to silence this warning and retain the current behavior

df.pivot_table(index=['sex','smoker'],columns=['day','time'],values='t
otal_bill')
```

```
day                  Thur                Fri                 Sat
Sun
time                Lunch Dinner      Lunch  Dinner     Dinner
Dinner
sex    smoker

Male   Yes     19.171000    NaN  11.386667  25.892  21.837778
26.141333
       No      18.486500    NaN        NaN  17.475  19.929063
20.403256
Female Yes     19.218571    NaN  13.260000  12.200  20.266667
16.540000
       No      15.899167  18.78  15.980000  22.750  19.003846
20.824286
```

```
df.pivot_table(index=['sex','smoker'],columns=['day','time']) # sare
data ko analye karke de dega
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_5096\1883326029.py:1:
FutureWarning: The default value of observed=False is deprecated and
will change to observed=True in a future version of pandas. Specify
observed=False to silence this warning and retain the current behavior
  df.pivot_table(index=['sex','smoker'],columns=['day','time'])
```

```
                   size
tip  \
day                Thur              Fri              Sat       Sun
Thur
time              Lunch Dinner     Lunch Dinner    Dinner    Dinner
Lunch
sex    smoker

Male   Yes     2.300000    NaN  1.666667    2.4  2.629630  2.600000
3.058000
```

```
       No      2.500000    NaN       NaN    2.0  2.656250  2.883721
2.941500
Female Yes     2.428571    NaN  2.000000    2.0  2.200000  2.500000
2.990000
       No      2.500000    2.0  3.000000    2.0  2.307692  3.071429
2.437083

                                                          total_bill
\
day                     Fri               Sat       Sun       Thur

time           Dinner Lunch Dinner    Dinner    Dinner      Lunch
Dinner
sex    smoker

Male   Yes        NaN  1.90  3.246  2.879259  3.521333  19.171000
NaN
       No         NaN   NaN  2.500  3.256563  3.115349  18.486500
NaN
Female Yes        NaN  2.66  2.700  2.868667  3.500000  19.218571
NaN
       No         3.0  3.00  3.250  2.724615  3.329286  15.899167
18.78


day                   Fri                Sat        Sun
time                Lunch  Dinner     Dinner     Dinner
sex    smoker
Male   Yes     11.386667  25.892  21.837778  26.141333
       No            NaN  17.475  19.929063  20.403256
Female Yes     13.260000  12.200  20.266667  16.540000
       No      15.980000  22.750  19.003846  20.824286

df.pivot_table(index=['sex','smoker'],columns=['day','time'],aggfunc={
'size':'mean','tip':'max','total_bill':'sum'})

C:\Users\jayra\AppData\Local\Temp\ipykernel_5096\2382294253.py:1:
FutureWarning: The default value of observed=False is deprecated and
will change to observed=True in a future version of pandas. Specify
observed=False to silence this warning and retain the current behavior

df.pivot_table(index=['sex','smoker'],columns=['day','time'],aggfunc={
'size':'mean','tip':'max','total_bill':'sum'})

                    size
                                                                    tip  \
day                 Thur              Fri              Sat       Sun
Thur
time              Lunch Dinner     Lunch Dinner    Dinner    Dinner
Lunch
```

```
sex    smoker

Male   Yes     2.300000    NaN  1.666667    2.4  2.629630  2.600000
5.00
       No      2.500000    NaN       NaN    2.0  2.656250  2.883721
6.70
Female Yes     2.428571    NaN  2.000000    2.0  2.200000  2.500000
5.00
       No      2.500000    2.0  3.000000    2.0  2.307692  3.071429
5.17

                                                    total_bill
\
day                    Fri           Sat    Sun       Thur
Fri
time          Dinner Lunch Dinner Dinner Dinner      Lunch Dinner
Lunch
sex    smoker

Male   Yes       NaN  2.20   4.73  10.00    6.5     191.71   0.00
34.16
       No        NaN   NaN   3.50   9.00    6.0     369.73   0.00
0.00
Female Yes       NaN  3.48   4.30   6.50    4.0     134.53   0.00
39.78
       No        3.0  3.00   3.25   4.67    5.2     381.58  18.78
15.98


day                     Sat           Sun
time           Dinner Lunch  Dinner Lunch  Dinner
sex    smoker
Male   Yes     129.46   0.0  589.62   0.0  392.12
       No       34.95   0.0  637.73   0.0  877.34
Female Yes      48.80   0.0  304.00   0.0   66.16
       No       22.75   0.0  247.05   0.0  291.54
```

```
# margins
df.pivot_table(index='sex',columns='smoker',values='total_bill',aggfun
c='sum')
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_5096\1740248909.py:2:
FutureWarning: The default value of observed=False is deprecated and
will change to observed=True in a future version of pandas. Specify
observed=False to silence this warning and retain the current behavior

df.pivot_table(index='sex',columns='smoker',values='total_bill',aggfun
c='sum')
```

```
smoker      Yes       No
sex
Male    1337.07  1919.75
Female   593.27   977.68
```

```
df.pivot_table(index='sex',columns='smoker',values='total_bill',aggfun
c='sum',margins=True) # margins bass total nikal ke de deta hai
```

```
C:\Users\jayra\AppData\Local\Temp\ipykernel_5096\521140019.py:1:
FutureWarning: The default value of observed=False is deprecated and
will change to observed=True in a future version of pandas. Specify
observed=False to silence this warning and retain the current behavior
```

```
df.pivot_table(index='sex',columns='smoker',values='total_bill',aggfun
c='sum',margins=True)
```

```
smoker      Yes       No      All
sex
Male    1337.07  1919.75  3256.82
Female   593.27   977.68  1570.95
All     1930.34  2897.43  4827.77
```

```
# plotting graph
df=pd.read_csv('expense_data.csv')
df
```

```
                  Date                Account        Category
Subcategory  \
0      3/2/2022 10:11  CUB - online payment            Food
NaN
1      3/2/2022 10:11  CUB - online payment           Other
NaN
2      3/1/2022 19:50  CUB - online payment            Food
NaN
3      3/1/2022 18:56  CUB - online payment  Transportation
NaN
4      3/1/2022 18:22  CUB - online payment            Food
NaN
..                ...                 ...             ...           .
..
272  11/22/2021 14:16  CUB - online payment            Food
NaN
273  11/22/2021 14:16  CUB - online payment            Food
NaN
274  11/21/2021 17:07  CUB - online payment  Transportation
NaN
275  11/21/2021 15:50  CUB - online payment            Food
NaN
276  11/21/2021 13:30  CUB - online payment           Other
NaN
```

```
                    Note     INR Income/Expense  Note.1  Amount
Currency  \
0                Brownie    50.0        Expense     NaN    50.0
INR
1      To lended people   300.0        Expense     NaN   300.0
INR
2                 Dinner    78.0        Expense     NaN    78.0
INR
3                  Metro    30.0        Expense     NaN    30.0
INR
4                 Snacks    67.0        Expense     NaN    67.0
INR
..                   ...     ...            ...     ...     ...       ..
.
272               Dinner    90.0        Expense     NaN    90.0
INR
273  Lunch with company    97.0        Expense     NaN    97.0
INR
274               Rapido   130.0        Expense     NaN   130.0
INR
275                Lunch   875.0        Expense     NaN   875.0
INR
276       Got from gobi  2000.0         Income     NaN  2000.0
INR

     Account.1
0         50.0
1        300.0
2         78.0
3         30.0
4         67.0
..         ...
272       90.0
273       97.0
274      130.0
275      875.0
276     2000.0

[277 rows x 11 columns]
```

```
df['Category'].value_counts()
```

```
Category
Food                156
Other                60
Transportation       31
Apparel               7
Household             6
Allowance             6
Social Life           5
```

```
Education              1
Salary                 1
Self-development       1
Beauty                 1
Gift                   1
Petty cash             1
Name: count, dtype: int64

df.info

<bound method DataFrame.info of                    Date
Account        Category  Subcategory  \
0      3/2/2022 10:11  CUB - online payment             Food
NaN
1      3/2/2022 10:11  CUB - online payment            Other
NaN
2      3/1/2022 19:50  CUB - online payment             Food
NaN
3      3/1/2022 18:56  CUB - online payment  Transportation
NaN
4      3/1/2022 18:22  CUB - online payment             Food
NaN
..                ...                    ...              ...           .
..
272  11/22/2021 14:16  CUB - online payment             Food
NaN
273  11/22/2021 14:16  CUB - online payment             Food
NaN
274  11/21/2021 17:07  CUB - online payment  Transportation
NaN
275  11/21/2021 15:50  CUB - online payment             Food
NaN
276  11/21/2021 13:30  CUB - online payment            Other
NaN

                    Note     INR Income/Expense  Note.1  Amount
Currency  \
0               Brownie    50.0        Expense     NaN    50.0
INR
1      To lended people   300.0        Expense     NaN   300.0
INR
2                Dinner    78.0        Expense     NaN    78.0
INR
3                 Metro    30.0        Expense     NaN    30.0
INR
4                Snacks    67.0        Expense     NaN    67.0
INR
..                  ...     ...            ...     ...     ...       ..
.
272              Dinner    90.0        Expense     NaN    90.0
```

```
INR
273  Lunch with company    97.0        Expense     NaN    97.0
INR
274               Rapido   130.0        Expense     NaN   130.0
INR
275                Lunch   875.0        Expense     NaN   875.0
INR
276       Got from gobi  2000.0         Income     NaN  2000.0
INR

     Account.1
0         50.0
1        300.0
2         78.0
3         30.0
4         67.0
..         ...
272       90.0
273       97.0
274      130.0
275      875.0
276     2000.0

[277 rows x 11 columns]>

df["Date"]=pd.to_datetime(df["Date"])

df.info

<bound method DataFrame.info of                    Date
Account        Category  Subcategory  \
0   2022-03-02 10:11:00  CUB - online payment            Food
NaN
1   2022-03-02 10:11:00  CUB - online payment           Other
NaN
2   2022-03-01 19:50:00  CUB - online payment            Food
NaN
3   2022-03-01 18:56:00  CUB - online payment  Transportation
NaN
4   2022-03-01 18:22:00  CUB - online payment            Food
NaN
..                  ...                   ...             ...
...
272 2021-11-22 14:16:00  CUB - online payment            Food
NaN
273 2021-11-22 14:16:00  CUB - online payment            Food
NaN
274 2021-11-21 17:07:00  CUB - online payment  Transportation
NaN
275 2021-11-21 15:50:00  CUB - online payment            Food
```

```
NaN
276 2021-11-21 13:30:00  CUB - online payment           Other
NaN

                    Note     INR Income/Expense  Note.1  Amount
Currency  \
0               Brownie    50.0        Expense     NaN    50.0
INR
1      To lended people   300.0        Expense     NaN   300.0
INR
2                Dinner    78.0        Expense     NaN    78.0
INR
3                 Metro    30.0        Expense     NaN    30.0
INR
4                Snacks    67.0        Expense     NaN    67.0
INR
..                  ...     ...            ...     ...     ...       ..
.
272              Dinner    90.0        Expense     NaN    90.0
INR
273  Lunch with company    97.0        Expense     NaN    97.0
INR
274              Rapido   130.0        Expense     NaN   130.0
INR
275               Lunch   875.0        Expense     NaN   875.0
INR
276       Got from gobi  2000.0         Income     NaN  2000.0
INR

     Account.1
0         50.0
1        300.0
2         78.0
3         30.0
4         67.0
..         ...
272       90.0
273       97.0
274      130.0
275      875.0
276     2000.0

[277 rows x 11 columns]>

df['month']=df["Date"].dt.month_name()
df['month']

0         March
1         March
2         March
```

```
3         March
4         March
         ...
272    November
273    November
274    November
275    November
276    November
Name: month, Length: 277, dtype: object
```

```
df.head()
```

```
                  Date                Account        Category
Subcategory  \
0 2022-03-02 10:11:00  CUB - online payment            Food
NaN
1 2022-03-02 10:11:00  CUB - online payment           Other
NaN
2 2022-03-01 19:50:00  CUB - online payment            Food
NaN
3 2022-03-01 18:56:00  CUB - online payment  Transportation
NaN
4 2022-03-01 18:22:00  CUB - online payment            Food
NaN

                Note    INR Income/Expense  Note.1  Amount Currency
Account.1  \
0           Brownie   50.0        Expense     NaN    50.0      INR
50.0
1  To lended people  300.0        Expense     NaN   300.0      INR
300.0
2            Dinner   78.0        Expense     NaN    78.0      INR
78.0
3             Metro   30.0        Expense     NaN    30.0      INR
30.0
4            Snacks   67.0        Expense     NaN    67.0      INR
67.0

   month
0  March
1  March
2  March
3  March
4  March
```

```
df.pivot_table(index='month',columns='Category',values='INR',aggfunc='
sum')
```

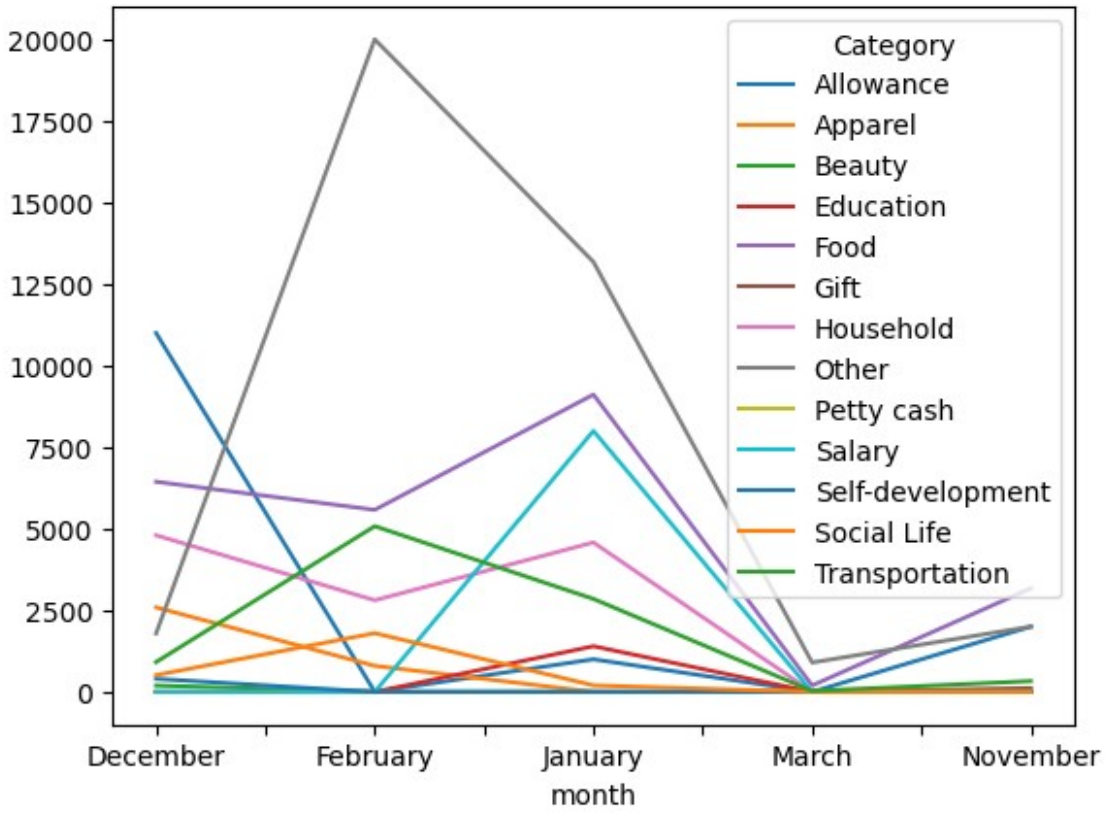
```
Category  Allowance  Apparel  Beauty  Education     Food   Gift
Household  \
```

```
month

December    11000.0   2590.0   196.0        NaN  6440.72    NaN
4800.0
February        NaN    798.0     NaN        NaN  5579.85    NaN
2808.0
January      1000.0      NaN     NaN     1400.0  9112.51    NaN
4580.0
March           NaN      NaN     NaN        NaN   195.00    NaN
NaN
November     2000.0      NaN     NaN        NaN  3174.40  115.0
NaN

Category    Other  Petty cash  Salary  Self-development  Social
Life  \
month

December   1790.0         NaN     NaN             400.0       513.72

February  20000.0         NaN     NaN               NaN      1800.00

January   13178.0         NaN  8000.0               NaN       200.00

March       900.0         NaN     NaN               NaN          NaN

November   2000.0         3.0     NaN               NaN          NaN


Category  Transportation
month
December           914.0
February          5078.8
January           2850.0
March               30.0
November           331.0

df.pivot_table(index='month',columns='Category',values='INR',aggfunc='
sum',fill_value=0)  #dropna bhi kar sakte hain

Category  Allowance  Apparel  Beauty  Education     Food   Gift
Household  \
month

December    11000.0   2590.0   196.0        0.0  6440.72    0.0
4800.0
February        0.0    798.0     0.0        0.0  5579.85    0.0
2808.0
January      1000.0      0.0     0.0     1400.0  9112.51    0.0
4580.0
March           0.0      0.0     0.0        0.0   195.00    0.0
0.0
```

```
November     2000.0      0.0     0.0         0.0  3174.40  115.0
0.0

Category    Other  Petty cash  Salary  Self-development  Social
Life  \
month

December   1790.0          0.0     0.0              400.0        513.72

February  20000.0          0.0     0.0                0.0      1800.00

January   13178.0          0.0  8000.0                0.0        200.00

March       900.0          0.0     0.0                0.0          0.00

November   2000.0          3.0     0.0                0.0          0.00


Category  Transportation
month
December           914.0
February          5078.8
January           2850.0
March               30.0
November           331.0
```

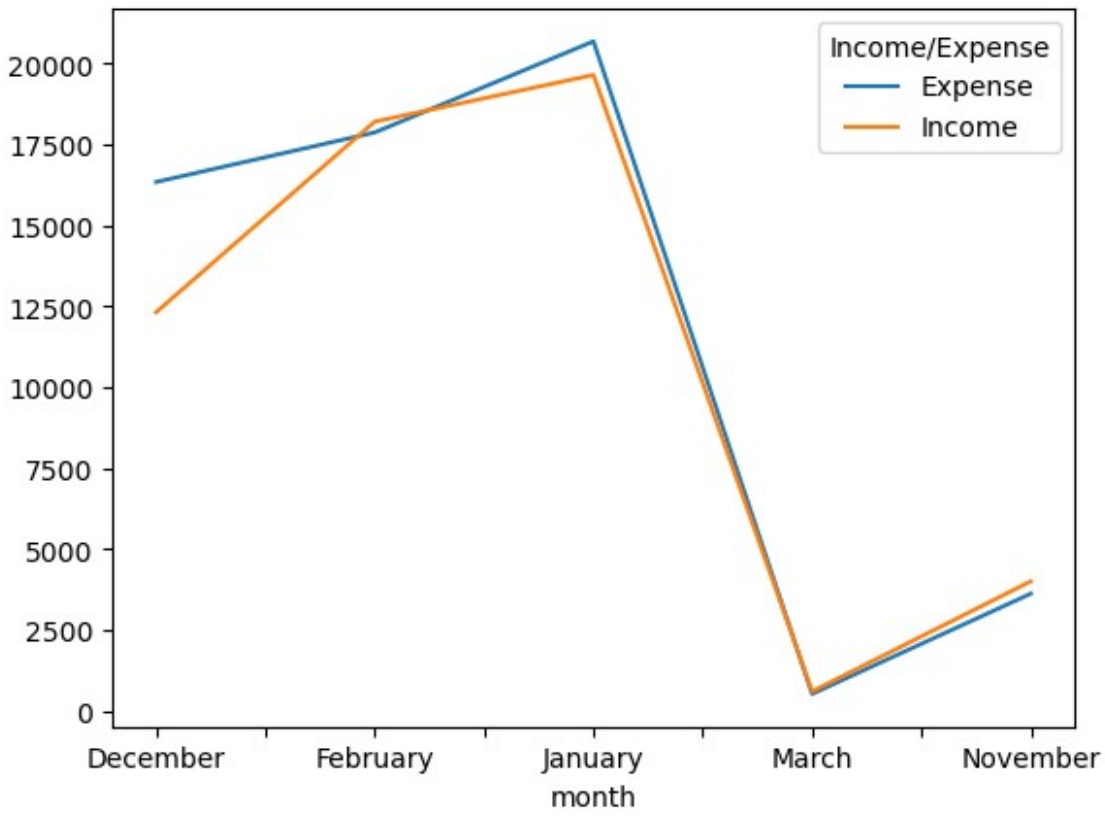
```
df.pivot_table(index='month',columns='Category',values='INR',aggfunc='
sum',fill_value=0).plot()
```

```
<Axes: xlabel='month'>
```



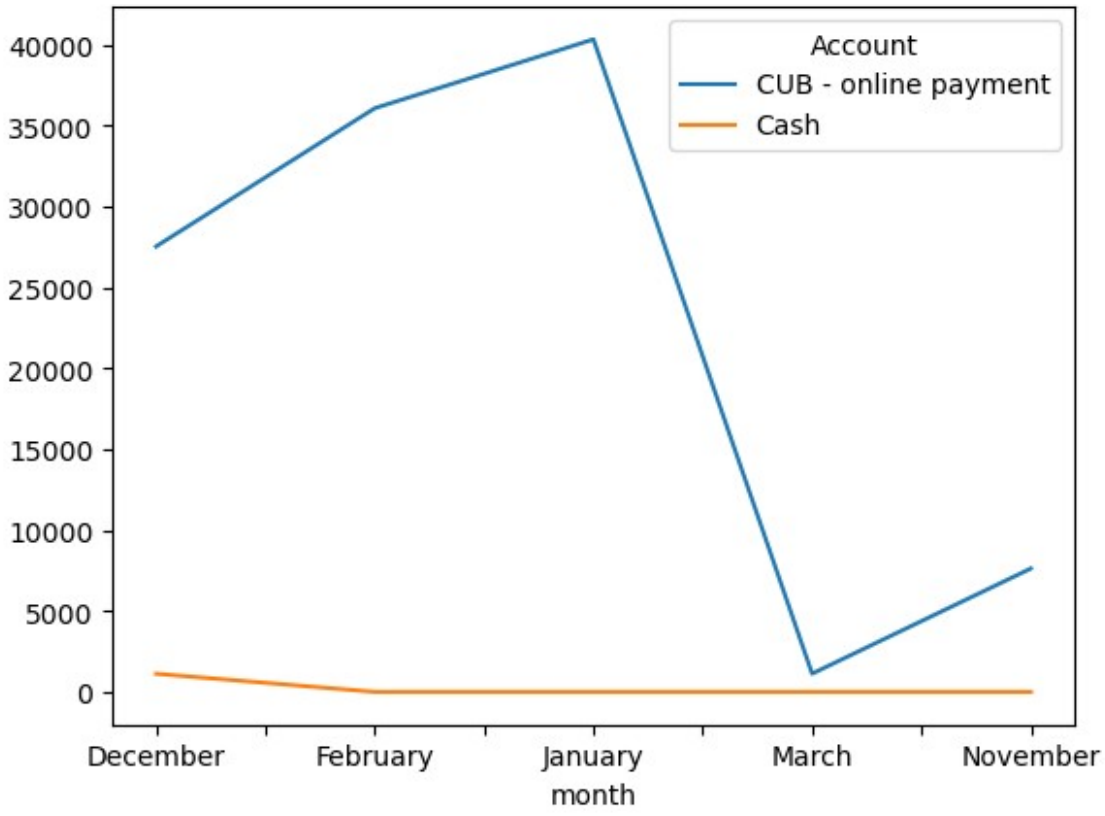

```
df.pivot_table(index='month',columns='Income/
Expense',values='INR',aggfunc='sum',fill_value=0).plot()
```

```
<Axes: xlabel='month'>
```




```
df.pivot_table(index='month',columns='Account',values='INR',aggfunc='sum',fill_value=0).plot()
```

```
<Axes: xlabel='month'>
```


Account
CUB - online payment
Cash
40000
35000
30000
25000
20000
15000
10000
5000
0
December
February
January
March
November
month