jupyter Pandas DataFrame Last Checkpoint: 2 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

[2]:
```
# Pandas is two dimensional datastructure
```

[27]:
```
import pandas as pd
import numpy as np
```

[10]:
```
print(dir(pd),end=" ")
```

```
['ArrowDtype', 'BooleanDtype', 'Categorical', 'CategoricalDtype', 'CategoricalIndex', 'DataFrame', 'DateOffset', 'DatetimeIndex', 'DatetimeTZDtype', 'ExcelFile', 'ExcelWriter', 'Flags', 'Float32Dtype', 'Float64Dtype', 'Grouper', 'HDFStore', 'Index', 'IndexSlice', 'Int16Dtype', 'Int32Dtype', 'Int64Dtype', 'Int8Dtype', 'Interval', 'IntervalDtype', 'IntervalIndex', 'MultiIndex', 'NA', 'NaT', 'NamedAgg', 'Period', 'PeriodDtype', 'PeriodIndex', 'RangeIndex', 'Series', 'SparseDtype', 'StringDtype', 'Timedelta', 'TimedeltaIndex', 'Timestamp', 'UInt16Dtype', 'UInt32Dtype', 'UInt64Dtype', 'UInt8Dtype', '__all__', '__builtins__', '__cached__', '__doc__', '__docformat__', '__file__', '__git_version__', '__loader__', '__name__', '__package__', '__path__', '__spec__', '__version__', '_built_with_meson', '_config', '_is_numpy_dev', '_libs', '_pandas_datetime_CAPI', '_pandas_parser_CAPI', '_testing', '_typing', '_version_meson', 'annotations', 'api', 'array', 'arrays', 'bdate_range', 'compat', 'concat', 'core', 'crosstab', 'cut', 'date_range', 'describe_option', 'errors', 'eval', 'factorize', 'from_dummies', 'get_dummies', 'get_option', 'infer_freq', 'interval_range', 'io', 'isna', 'isnull', 'json_normalize', 'lreshape', 'melt', 'merge', 'merge_asof', 'merge_ordered', 'notna', 'notnull', 'offsets', 'option_context', 'options', 'pandas', 'period_range', 'pivot', 'pivot_table', 'plotting', 'qcut', 'read_clipboard', 'read_csv', 'read_excel', 'read_feather', 'read_fwf', 'read_gbq', 'read_hdf', 'read_html', 'read_json', 'read_orc', 'read_parquet', 'read_pickle', 'read_sas', 'read_spss', 'read_sql', 'read_sql_query', 'read_sql_table', 'read_stata', 'read_table', 'read_xml', 'reset_option', 'set_eng_float_format', 'set_option', 'show_versions', 'test', 'testing', 'timedelta_range', 'to_datetime', 'to_numeric', 'to_pickle', 'to_timedelta', 'tseries', 'unique', 'util', 'value_counts', 'wide_to_long']
```

[12]:
```
a=[[10,20,30,40],[50,60,70,80]]
```

[14]:
```
a
```

[14]:
```
[[10, 20, 30, 40], [50, 60, 70, 80]]
```

[16]:
```
type(a)
```

[16]:
```
list
```

[19]:
```
pd.DataFrame(a)
```

[19]:

	0	1	2	3
0	10	20	30	40
1	50	60	70	80

[23]:
```
b=pd.Series([[10, 20, 30, 40], [50, 60, 70, 80]])
```

[25]:
```
pd.DataFrame(b)
```

[25]:

	0
0	[10, 20, 30, 40]
1	[50, 60, 70, 80]

[29]:
```
c=np.array([[10, 20, 30, 40], [50, 60, 70, 80]])
```

[31]:
```
c
```

[31]:
```
array([[10, 20, 30, 40],
       [50, 60, 70, 80]])
```

[33]:
```
pd.DataFrame(c)
```

[33]:

	0	1	2	3
0	10	20	30	40
1	50	60	70	80

[45]:
```
d = {"roll_no":["101","102","103","104"],
    "name":["chandru","rohit","Ashish","Devdata"],
    "maths":[72,88,56,59],
    "science":[73,89,57,60],
}
```

[47]:
```
d
```

[47]:
```
{'roll_no': ['101', '102', '103', '104'],
 'name': ['chandru', 'rohit', 'Ashish', 'Devdata'],
 'maths': [72, 88, 56, 59],
 'science': [73, 89, 57, 60]}
```

[49]:
```
pd.DataFrame(d)
```

[49]:

	roll_no	name	maths	science
0	101	chandru	72	73
1	102	rohit	88	89
2	103	Ashish	56	57
3	104	Devdata	59	60

[53]:
```
#importing data from excel

data = pd.read_excel(r'C:\Users\CHANDRASHEKAR\Downloads\student_pbi.xlsx')
```

[57]:
```
student = pd.DataFrame(data)
```

[61]:
```
#select * from student
student
```

[61]:

	ROLL_NO	NAME	DEPT	MATHS	SCIENCE	SOCIAL	KANNADA	ENGLISH	HINDI
0	101	RAMESH	ANALYTICS	70	58	67	92	36	60
1	102	SURESH	MBA	87	96	43	42	69	68

2	103	GANESH	BBA	46	86	68	71	72	79
3	104	CHANDRU	ANALYTICS	58	74	91	38	73	88
4	105	SOMU	MBA	72	61	81	47	80	97
5	106	SHIVU	BBA	46	41	41	94	53	69
6	107	VINOD	ANALYTICS	43	45	74	62	62	93
7	108	ROOPA	MBA	62	58	70	64	63	38
8	109	VIGNESH	BBA	63	72	93	85	67	98
9	110	VYLESH	ANALYTICS	64	47	95	71	50	58
10	111	KUMAR	MBA	46	65	74	94	92	43
11	112	SAKSHI	BBA	96	58	78	66	51	72
12	113	ANIL	ANALYTICS	68	66	61	96	38	67
13	114	SUJITH	MBA	75	84	39	71	69	39
14	115	KRISHNA	BBA	51	84	74	61	63	73
15	116	VASANTH	ANALYTICS	96	58	78	66	51	72
16	117	QUEEN	ANALYTICS	68	66	61	96	38	67

*[67]:
```
#select * from student order by roll_no asc limit 3
student.head(3)
```

[67]:

	ROLL_NO	NAME	DEPT	MATHS	SCIENCE	SOCIAL	KANNADA	ENGLISH	HINDI
0	101	RAMESH	ANALYTICS	70	58	67	92	36	60
1	102	SURESH	MBA	87	96	43	42	69	68
2	103	GANESH	BBA	46	86	68	71	72	79

[71]:
```
#select * from student order by roll_no desc limit 3
student.tail(3)
```

[71]:

	ROLL_NO	NAME	DEPT	MATHS	SCIENCE	SOCIAL	KANNADA	ENGLISH	HINDI
14	115	KRISHNA	BBA	51	84	74	61	63	73
15	116	VASANTH	ANALYTICS	96	58	78	66	51	72
16	117	QUEEN	ANALYTICS	68	66	61	96	38	67

[73]:
```
student.columns
```

[73]:
```
Index(['ROLL_NO', 'NAME', 'DEPT', 'MATHS', 'SCIENCE', 'SOCIAL', 'KANNADA',
       'ENGLISH', 'HINDI'],
      dtype='object')
```

[77]:
```
student.shape
```

[77]:
```
(17, 9)
```

[81]:
```
student.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17 entries, 0 to 16
Data columns (total 9 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   ROLL_NO   17 non-null     int64 
 1   NAME      17 non-null     object
 2   DEPT      17 non-null     object
 3   MATHS     17 non-null     int64 
 4   SCIENCE   17 non-null     int64 
 5   SOCIAL    17 non-null     int64 
 6   KANNADA   17 non-null     int64 
 7   ENGLISH   17 non-null     int64 
 8   HINDI     17 non-null     int64 
dtypes: int64(7), object(2)
memory usage: 1.3+ KB
```

[83]:
```
student.describe()
```

[83]:

	ROLL_NO	MATHS	SCIENCE	SOCIAL	KANNADA	ENGLISH	HINDI
count	17.000000	17.000000	17.000000	17.000000	17.000000	17.000000	17.000000
mean	109.000000	65.352941	65.823529	69.882353	71.529412	60.411765	69.470588
std	5.049752	16.624309	15.322513	16.951705	18.917973	15.455819	18.439488
min	101.000000	43.000000	41.000000	39.000000	38.000000	36.000000	38.000000
25%	105.000000	51.000000	58.000000	61.000000	62.000000	51.000000	60.000000
50%	109.000000	64.000000	65.000000	74.000000	71.000000	63.000000	69.000000
75%	113.000000	72.000000	74.000000	78.000000	92.000000	69.000000	79.000000
max	117.000000	96.000000	96.000000	95.000000	96.000000	92.000000	98.000000

[ ]:
```

```