Pandas

~~dfate~~ → import pandas as PD

→ Pd.read_csv(' ') → reads the data and creates a dataframe.

→ Pd.head() → retrieves the first 5 records

→ Pd.tail() → retrieves the last 5 records

→ Pd.info() → retrieves the information of the dataset/dataframe.

→ Pd.describe() → Describes the dataframe without including categorical data

→ Pd.columns → Retrieves all the column names

→ Pd.T → Returns transpose of the dataframe

→ pd.sort_values('Age') → Returns all the values in Ascending order for column Age.

→ pd['balance'] = pd['balance'] + 1000 → sums all records of the column balance with 1000.

* Accessing rows: df[10:13]

⊛ can Analyze Age > 50. → df_new = df[df.Age > 50]

03/06/25 → Remove columns: df.pop('columnname')

(or) df.drop('columnname', axis = 1)

→ filtering ⇒ df[df.Age > 50]

→ filling null values: df['Age'].fillna(10) → fills null values with 10

→ df['Balance']

→ Apply Method: df['Balance'].apply(np.sqrt)

↳ Applying a function onto a column in a dataframe

⊛ Concating and Merging: ① Inner
② OUTER
③ LEFT
④ RIGHT

Table1:

Custid	Cust_name
1	Satyajit
2	hillary
3	raj
4	Shankar

Table2:

cust_id	Salary
2	10000
3	25000

InnerJoin:

cust_id	cust_name	Salary
2	hillary	10000
3	raj	25000

Left Join:

1	Satyajith	-
2	hillary	10000
3	raj	25000
4	Shankar	-

Right Join

2	hillary	10000
3	raj	25000

→ pd.concat : Concatenates all the dataframes → pd.concat([df1, df2])

→ Merge : merge_df = pd.merge(df1, df2, on='c', how='inner')

works same as a inner join in SQL Pothoda.

④ loc vs iloc:

① loc: returns rows and/or columns with particular labels.

② iloc: returns rows and/or columns at integer locations.

Example: s = pd.Series(list("abcdef"), index=[49,48,47,0,1,2])

49	a
48	b
47	c
0	d
1	e
2	f

i/p: s.loc[0]
o/p: 'd'

i/p: s.iloc[0]
o/p: 'a'