THAM KHẢO NHANH PANDAS

DataFrames, lựa chọn, tổng hợp, kết hợp và nhiều hơn

DataFrames

Tạo DataFrames

```
import pandas as pd
df = pd.DataFrame({
    "name": ["Alice", "Bob", "Carol"],
    "age": [25, 30, 35],
    "score": [88, 92, 79]
})
```

Kiểm Tra

`df.head(n)`	n hàng đầu tiên (mặc định 5)
`df.tail(n)`	n hàng cuối cùng
`df.shape`	Tuple của (hàng, cột)
`df.dtypes`	Kiểu dữ liệu của mỗi cột
`df.info()`	Kiểu cột, số lượng không null
`df.describe()`	Thống kê cho cột số
`df.columns`	Tên cột dạng Index
`df.index`	Nhãn hàng

Đọc Dữ Liệu

Các Hàm Đọc Phổ Biến

```
df = pd.read_csv("data.csv")
df = pd.read_excel("data.xlsx")
df = pd.read_json("data.json")
df = pd.read_sql(query, connection)
```

Ghi Dữ Liệu

```
df.to_csv("out.csv", index=False)
df.to_excel("out.xlsx", index=False)
df.to_json("out.json", orient="records")
```

Tùy Chọn Đọc

`sep=";"`	Dấu phân cách tùy chỉnh
`header=None`	File không có hàng tiêu đề
`usecols=[0,2]`	Chỉ đọc cột cụ thể
`nrows=100`	Đọc 100 hàng đầu tiên
`na_values=["N/A"]`	Coi là NaN

Lựa Chọn

Cột

```
df["name"]              # single column (Series)
df[["name", "age"]]     # multiple columns (DataFrame)
df.name                 # attribute access (simple names)
```

Hàng Với loc / iloc

```
df.loc[0]             # row by label
df.loc[0:2, "name"]   # rows 0-2, column "name"
df.iloc[0]            # row by position
df.iloc[0:2, 0:2]     # first 2 rows, 2 cols
```

loc vs iloc

`df.loc[row, col]`	Chọn theo **nhãn** (cuối bao gồm)
`df.iloc[row, col]`	Chọn theo **vị trí** (cuối không bao gồm)
`df.at[row, col]`	Truy cập scalar nhanh theo nhãn
`df.iat[row, col]`	Truy cập scalar nhanh theo vị trí

Lọc

Lọc Boolean

```
df[df["age"] > 25]
df[df["name"].str.contains("li")]
df[(df["age"] > 25) & (df["score"] > 80)]
df[df["name"].isin(["Alice", "Bob"])]
```

Xử Lý Dữ Liệu Thiếu

```
df.isna().sum()          # NaN count per column
df.dropna()              # drop rows with any NaN
df.fillna(0)             # fill NaN with 0
df["col"].fillna(df["col"].mean())
```

Sắp Xếp

```
df.sort_values("age")                # ascending
df.sort_values("age", ascending=False)
df.sort_values(["age", "score"])     # multi
```

Tổng Hợp

Tổng Hợp Phổ Biến

`df["col"].sum()`	Tổng của cột
`df["col"].mean()`	Giá trị trung bình
`df["col"].median()`	Trung vị
`df["col"].std()`	Độ lệch chuẩn
`df["col"].min() / .max()`	Nhỏ nhất / lớn nhất
`df["col"].count()`	Số lượng không null
`df["col"].nunique()`	Số giá trị duy nhất
`df["col"].value_counts()`	Tần suất của mỗi giá trị

Nhiều Tổng Hợp

```
df.agg({"age": "mean", "score": ["min", "max"]})
df.describe()  # summary stats for all numeric
```

GroupBy

Nhóm Cơ Bản

```
df.groupby("dept")["salary"].mean()
df.groupby("dept").agg(
    avg_sal=("salary", "mean"),
    count=("salary", "count")
)
```

Nhiều Nhóm

```
df.groupby(["dept", "year"])["sales"].sum()
df.groupby("dept").size()  # rows per group
```

Transform & Apply

```
df["z_score"] = df.groupby("dept")["salary"] \
    .transform(lambda x: (x - x.mean()) / x.std())
df.groupby("dept").apply(lambda g: g.nlargest(3, "salary"))
```

Kết Hợp

Merge (Join kiểu SQL)

```
pd.merge(df1, df2, on="id")          # inner
pd.merge(df1, df2, on="id", how="left")
pd.merge(df1, df2, left_on="uid",
         right_on="user_id")
```

Các Loại Join

`how="inner"`	Chỉ giữ hàng khớp (mặc định)
`how="left"`	Giữ tất cả hàng trái, NaN nếu không khớp
`how="right"`	Giữ tất cả hàng phải
`how="outer"`	Giữ tất cả hàng từ cả hai phía

Nối

```
pd.concat([df1, df2])            # stack rows
pd.concat([df1, df2], axis=1)    # side by side
pd.concat([df1, df2], ignore_index=True)
```

Pivot Tables

Pivot Table

```
df.pivot_table(
    values="sales", index="region",
    columns="quarter", aggfunc="sum"
)
```

Biến Đổi Hình Dạng

```
df.melt(id_vars=["name"],
        value_vars=["q1", "q2"],
        var_name="quarter", value_name="sales")
```

Bảng Chéo

```
pd.crosstab(df["dept"], df["gender"])
pd.crosstab(df["dept"], df["gender"],
            normalize="index")  # row percentages
```

Chuỗi Thời Gian

Cơ Bản DateTime

```
df["date"] = pd.to_datetime(df["date"])
df["year"] = df["date"].dt.year
df["month"] = df["date"].dt.month
df["weekday"] = df["date"].dt.day_name()
```

Khoảng Ngày & Resample

```
pd.date_range("2025-01-01", periods=12, freq="ME")
df.set_index("date").resample("ME")["sales"].sum()
```

Thuộc Tính Accessor

`.dt.year / .dt.month / .dt.day`	Trích xuất thành phần ngày
`.dt.hour / .dt.minute`	Trích xuất thành phần giờ
`.dt.day_name()`	Tên thứ trong tuần (Monday, v.v.)
`.dt.days_in_month`	Số ngày trong tháng đó

Mẫu Phổ Biến

Đổi Tên Cột

```
df.rename(columns={"old": "new"})
df.columns = ["a", "b", "c"]  # replace all
```

Thêm / Sửa Cột

```
df["total"] = df["q1"] + df["q2"]
df["grade"] = df["score"].apply(
    lambda x: "A" if x >= 90 else "B"
)
```

Xóa Cột / Hàng

```
df.drop(columns=["temp"])
df.drop_duplicates(subset=["name"])
df.reset_index(drop=True)
```

Thao Tác Chuỗi

```
df["name"].str.lower()
df["name"].str.contains("ali", case=False)
df["name"].str.split(" ").str[0]  # first name
```