Cheatsheet 1

In **pandas**, there are a few quick + reliable ways to check the **data type of columns**, depending on what you need.

1️⃣ Check *all* column types (most common)

`df.dtypes`

Output example

```
age         int64
height    float64
name       object
active       bool
date     datetime64[ns]
dtype: object
```

2️⃣ Detailed overview (very useful for debugging)

`df.info()`

Shows:

- column name
- non-null count
- dtype
- memory usage

3️⃣ Check type of a *single column*

`df["age"].dtype`

or

`df.age.dtype`

4️⃣ Check only numeric / non-numeric columns

```
df.select_dtypes(include="number").columns
```

```
df.select_dtypes(exclude="number").columns
```

5️⃣ Check if a column is a specific type

```
import pandas as pd

pd.api.types.is_numeric_dtype(df["age"])
pd.api.types.is_string_dtype(df["name"])
pd.api.types.is_datetime64_any_dtype(df["date"])
pd.api.types.is_bool_dtype(df["active"])
```

6️⃣ Count how many columns per dtype

```
df.dtypes.value_counts()
```

7️⃣ Detect mixed or problematic types (object columns)

```
df.select_dtypes(include="object").apply(
    lambda s: s.map(type).value_counts()
)
```

Useful when:

- CSV imported badly
- numbers stored as strings
- mixed types in one column

Task	Command
See all dtypes	`df.dtypes`
Full summary	`df.info()`
Single column	`df[col].dtype`
Only numeric	`select_dtypes()`
Type check	`pd.api.types.is_*`

Revision #2
Created 23 January 2026 02:08:49 by Sandi Wibowo
Updated 23 January 2026 02:12:11 by Sandi Wibowo