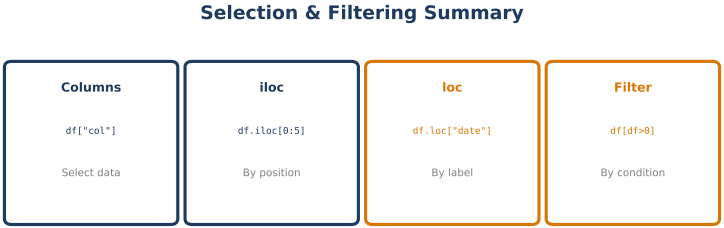
# Lesson 06 Summary: Selection and Filtering

## Data Science with Python – Key Concepts

Data Science Program

# Selection Overview




**Selection and filtering extract relevant data for analysis**

# Column Selection

**Column Selection Methods**

| | |
|---|---|
| `df['AAPL']` | Single column (Series) |
| `df[['AAPL', 'MSFT']]` | Multiple columns (DataFrame) |
| `df.AAPL` | Attribute access (simple names) |
| `df.loc[:, 'AAPL':'GOOGL']` | Range of columns |

**Single brackets return Series; double brackets return DataFrame**

# iloc vs loc

**Two ways to access rows and columns:**

**iloc (Integer Location):**

- Position-based: 0, 1, 2, ...
- Exclusive end: `df.iloc[0:5]` gets rows 0-4
- Example: `df.iloc[0, 1]` = first row, second column

**loc (Label Location):**

- Label-based: dates, column names
- Inclusive end: `df.loc['2024-01-01':'2024-01-05']`
- Example: `df.loc['2024-01-01', 'AAPL']`

---

**iloc: integer position; loc: label-based**

# Boolean Filtering

**Filter rows using conditions:**

```
# Create boolean mask
mask = df["AAPL"] > 190

# Apply mask to filter
filtered = df[mask]

# Or in one step
filtered = df[df["AAPL"] > 190]
```

**Result:** Only rows where condition is True

**Boolean mask selects rows where condition is True**

# Combining Conditions

**Logical operators for multiple conditions:**

- **&** (AND): Both conditions must be True
- — (OR): Either condition can be True
- ~ (NOT): Inverts the condition

**Example:**

```
df[(df["AAPL"] > 185) & (df["MSFT"] > 375)]
```

**Important:** Always use parentheses around each condition!

**Use parentheses around each condition!**

# query() Method

**More readable alternative for complex filters:**

**Traditional:**

```
df[(df["AAPL"] > 185) & (df["Volume"] > 1e6)]
```

**Using query():**

```
df.query("AAPL > 185 and Volume > 1e6")
```

**isin() for membership:**

```
df[df["Symbol"].isin(["AAPL", "MSFT"])]
```

---

**query() method is more readable for complex filters**

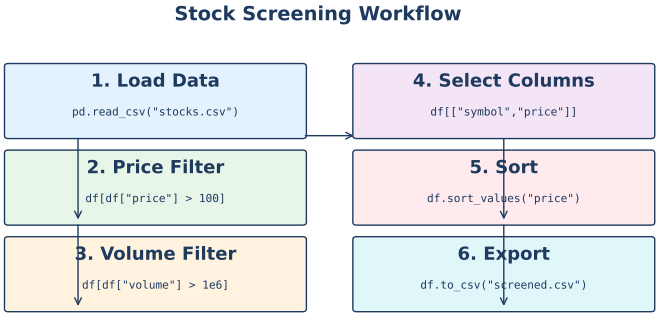
# Selection Methods

**Selection Methods Comparison**

| Method | Use Case | Returns |
|---|---|---|
| `df["col"]` | Single column | Series |
| `df[["col1","col2"]]` | Multiple columns | DataFrame |
| `df.iloc[0]` | Row by position | Series |
| `df.loc["date"]` | Row by label | Series |
| `df[df.col > x]` | Filter rows | DataFrame |

**Choose method based on what you need to select**

# Stock Screening




*Combine filters to build powerful stock screeners*

**Stock screening = loading + filtering + selecting + sorting**

# Quick Reference

**Essential Selection Operations:**

| Operation | Syntax |
|---|---|
| Single column | `df["col"]` (Series) |
| Multiple columns | `df[["col1", "col2"]]` |
| Row by position | `df.iloc[0]` |
| Rows by range | `df.iloc[0:5]` |
| Row by label | `df.loc["label"]` |
| Filter by condition | `df[df["col"] > 0]` |
| AND conditions | `(cond1) & (cond2)` |
| OR conditions | `(cond1) \| (cond2)` |
| Query method | `df.query("col > 0")` |

**Master selection for efficient data manipulation**