FOUNDATION LESSON

WEEK 4 · LESSON 11

Create columns and parse dates

Derived columns retain row meaning; date parsing follows an explicit format

THE IDEA BEHIND THE OPERATION

MENTAL MODEL

The idea behind the operation

Derived columns retain row meaning; date parsing follows an explicit format

CALCULATE EACH INVOICE REVENUE

CODE WALKTHROUGH W04-L11-S03

```
01  import pandas as pd
02  df = pd.DataFrame({"units":[2,0,5], "price":[12.0,8.0,3.0]})
03  df["revenue"] = df["units"] * df["price"]
04  print(df.to_string(index=False))
05  print("Total:", df["revenue"].sum())
```

WHAT TO TRACK

Revenue is quantity times unit price, in EUR.

PREDICT THE RESULT

Predict the output.
Explain each operation.
Pause before continuing.

CALCULATE EACH INVOICE REVENUE — RESULT

CODE WALKTHROUGH W04-L11-S04

```
01  import pandas as pd
02  df = pd.DataFrame({"units":[2,0,5], "price":[12.0,8.0,3.0]})
03  df["revenue"] = df["units"] * df["price"]
04  print(df.to_string(index=False))
05  print("Total:", df["revenue"].sum())
```

WHAT TO TRACK

Revenue is quantity times unit price, in EUR.

VALIDATE THE RESULT

```
 units  price  revenue
     2   12.0     24.0
     0    8.0      0.0
     5    3.0     15.0
Total: 39.0
```

ASSIGNMENT ALIGNS SERIES LABELS

CODE WALKTHROUGH W04-L11-S05

```
01  import pandas as pd
02  df = pd.DataFrame({"product":["A","B"]}, index=[10,20])
03  df["wrong"] = pd.Series([5,8])
04  df["right"] = pd.Series([5,8], index=df.index)
05  print(df.to_string())
```

WHAT TO TRACK

The external Series must use the intended row labels.

PREDICT THE RESULT

Predict the output.
Explain each operation.
Pause before continuing.

ASSIGNMENT ALIGNS SERIES LABELS — RESULT

CODE WALKTHROUGH W04-L11-S06

```
01  import pandas as pd
02  df = pd.DataFrame({"product":["A","B"]}, index=[10,20])
03  df["wrong"] = pd.Series([5,8])
04  df["right"] = pd.Series([5,8], index=df.index)
05  print(df.to_string())
```

WHAT TO TRACK

The external Series must use the intended row labels.

VALIDATE THE RESULT

```
   product  wrong  right
10       A    NaN      5
20       B    NaN      8
```

INVALID DATES REMAIN VISIBLE

CODE WALKTHROUGH W04-L11-S07

```
01  import pandas as pd
02  text = pd.Series(["2026-01-02", "2026-02-30", "bad"])
03  dates = pd.to_datetime(text, format="%Y-%m-%d", errors="coerce"
↪   )
04  print(dates.dt.strftime("%Y-%m-%d").fillna("INVALID").tolist())
05  print("Invalid:", int(dates.isna().sum()))
```

WHAT TO TRACK

Use year-month-day; invalid dates are flagged, not accepted.

PREDICT THE RESULT

Predict the output.
Explain each operation.
Pause before continuing.

INVALID DATES REMAIN VISIBLE — RESULT

CODE WALKTHROUGH W04-L11-S08

```
01  import pandas as pd
02  text = pd.Series(["2026-01-02", "2026-02-30", "bad"])
03  dates = pd.to_datetime(text, format="%Y-%m-%d", errors="coerce"
↪   )
04  print(dates.dt.strftime("%Y-%m-%d").fillna("INVALID").tolist())
05  print("Invalid:", int(dates.isna().sum()))
```

WHAT TO TRACK

Use year-month-day; invalid dates are flagged, not accepted.

VALIDATE THE RESULT

```
['2026-01-02', 'INVALID',
'INVALID']
Invalid: 2
```

GUIDED MODIFICATION

LEARNER PRACTICE

Guided modification

Calculate revenue for units [3, 4] and prices [5.0, 2.5]. Parse dates 2026-03-01 and 2026-03-02 with an explicit format. Print the two revenues and dates.

PREDICT · MODIFY · RUN

01 Restate the supplied rules.

02 Predict the result.

03 Modify and run the code.

04 Compare actual output.

05 Explain any difference.

INDEPENDENT APPLICATION

LEARNER PRACTICE

Independent application

Parse [2026-01-01, 2026-13-01] under the year-month-day rule. Report the invalid count and retain only valid dates in a separate report. Preserve the original text for correction.

PREDICT · MODIFY · RUN

01 Restate the supplied rules.

02 Predict the result.

03 Modify and run the code.

04 Compare actual output.

05 Explain any difference.

RECAP AND NEXT STEP

Recap and next step

Derived columns implement stated row rules. Series assignment aligns labels, and explicit date formats prevent ambiguity. Count parsing failures rather than hiding them. Next, we combine string operations with a complete offline DataFrame report.

UNDERSTAND APPLY VALIDATE EXPLAIN