In [25]:
```
!pip install pandas openpyxl scipy statsmodels seaborn matplotlib pingouin -
```

```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 204.4/204.4 kB 3.3 MB/s eta 0:0
0:00a 0:00:01
```

In [4]:
```
import pandas as pd

# Load the Excel file
df = pd.read_excel('/kaggle/input/raw-data-adilah2/Raw data Adilah2.xlsx', s
```

In [5]:
```
df = df[~df['Name'].isin(['Tan See Ling','Dai Aihui'])]
```

In [6]:
```
df.head()
```

Out[6]:

	Subject	Name	Gender	Age	Race	Tooth	Impaction	Duration	Gr
0	1	Muhammad Amerul Akmal	Male	27	Malay	38	2A	35	Plac
1	2	Sarah Nabihah	Female	28	Malay	38	2A	28	L
2	3	Pan Anyu	Female	25	Others	38	2A	27	L
3	4	Afiqah bitni Abdul Malik	Female	26	Malay	48	2A	23	L
4	5	Tiang Nga Li	Female	20	Chinese	48	2A	18	Plac

5 rows × 42 columns

In [7]:
```
df.columns
```

Out[7]:
```
Index(['Subject', 'Name', 'Gender ', 'Age', 'Race', 'Tooth', 'Impaction',
       'Duration', 'Group', 'VAS (baseline)', 'VAS Day1', 'VAS day2 ',
       'VAS Day7', 'MO (baseline)', 'MO Day1', 'Score MO Day1', 'MO Day2',
       'Score MO Day2', 'MO Day7', 'Score MO Day7', 'S1 (baseline)', 'S1 da
y1',
       'Score S1 day1', 'S1 day2', 'Score S1 day2', 'S1 day7', 'Score S1 da
y7',
       'S2 (baseline)', 'S2 day1', 'Score S2 day1', 'S2 day2', 'Score S2 da
y2',
       'S2 day7', 'Score S2 day7', 'S3 (baseline)', 'S3 day1', 'Score S3 da
y1',
       'S3 day2', 'Score S3 day2', 'S3 day7', 'Score S3 day7',
       'Arcoxia (days)'],
      dtype='object')
```

In [14]:
```
df2 = df[['Subject', 'Gender ', 'Age', 'Race', 'Tooth', 'Impaction','Duratic
          'S1 (baseline)', 'S1 day1','S1 day2', 'S1 day7',
```

```
            'S2 (baseline)', 'S2 day1', 'S2 day2', 'S2 day7',
        'S3 (baseline)', 'S3 day1', 'S3 day2', 'S3 day7','Arcoxia (days)']]
```

In [18]:
```
df2_melted = df2.melt(
    id_vars=['Subject', 'Gender ', 'Group','Arcoxia (days)'],
    value_vars=[
        'S1 (baseline)', 'S1 day1', 'S1 day2', 'S1 day7',
        'S2 (baseline)', 'S2 day1', 'S2 day2', 'S2 day7',
        'S3 (baseline)', 'S3 day1', 'S3 day2', 'S3 day7'
    ],
    var_name='Timepoint',
    value_name='Value'
)


df2_melted
```

Out[18]:

	Subject	Gender	Group	Arcoxia (days)	Timepoint	Value
0	1	Male	Placebo	4	S1 (baseline)	98
1	2	Female	Laser	2	S1 (baseline)	93
2	3	Female	Laser	0	S1 (baseline)	105
3	4	Female	Laser	2	S1 (baseline)	100
4	5	Female	Placebo	3	S1 (baseline)	107
...	...	...	...	...	...	...
775	63	Female	Laser	1	S3 day7	155
776	64	Male	Placebo	3	S3 day7	155
777	65	Female	Placebo	4	S3 day7	125
778	66	Male	Laser	1	S3 day7	122
779	67	Male	Placebo	4	S3 day7	120

780 rows × 6 columns

In [19]:
```
import pandas as pd

# Clean column names (e.g., remove extra spaces)
df2_melted.columns = df2_melted.columns.str.strip()

# Extract Subject/Session (S1, S2, S3) and Day from Timepoint
df2_melted[['Session', 'Day']] = df2_melted['Timepoint'].str.extract(r'(S\d)

# Replace common labels
df2_melted['Day'] = df2_melted['Day'].replace({
    'baseline': 'Day0',
    'day1': 'Day1',
    'day2': 'Day2',
    'day7': 'Day7'
})
```

```
# Convert Day to categorical with order
day_order = ['Day0', 'Day1', 'Day2', 'Day7']
df2_melted['Day'] = pd.Categorical(df2_melted['Day'], categories=day_order,
```

In [21]:
```
# Group summary
summary_stats = df2_melted.groupby(['Group', 'Day'])['Value'].describe()
summary_stats
```

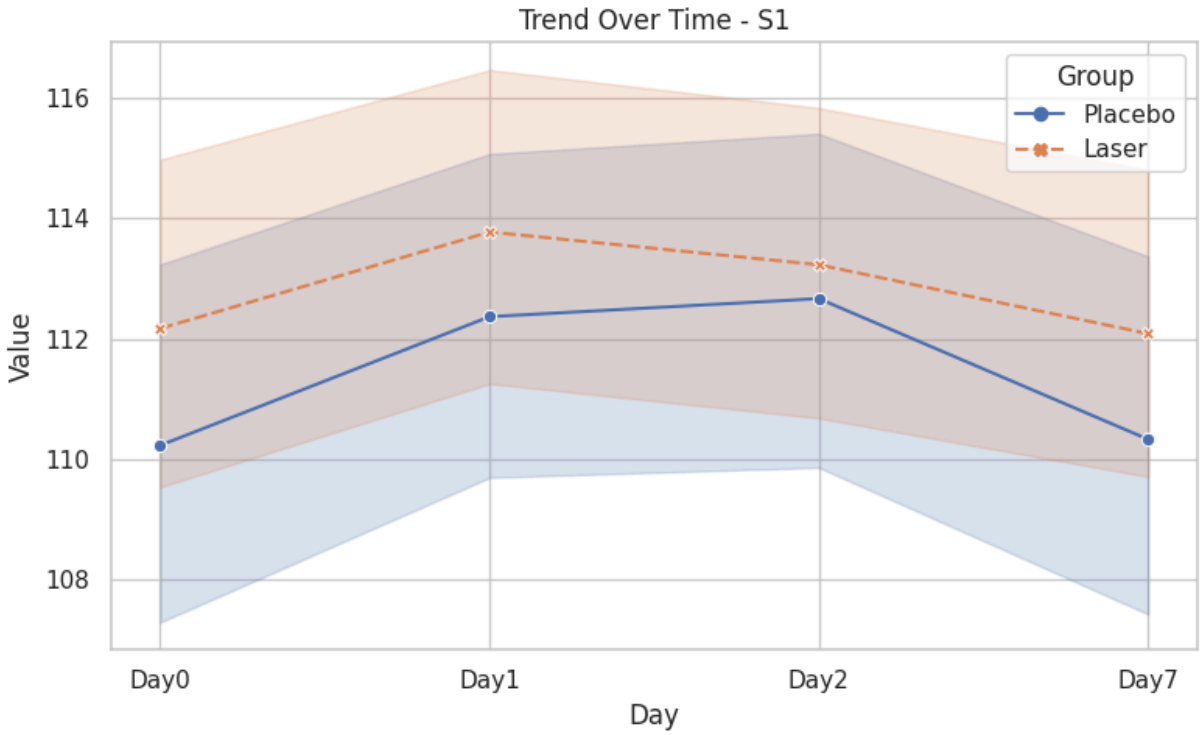
```
/tmp/ipykernel_35/2429070538.py:2: FutureWarning: The default of observed=Fa
lse is deprecated and will be changed to True in a future version of pandas.
Pass observed=False to retain current behavior or observed=True to adopt the
future default and silence this warning.
  summary_stats = df2_melted.groupby(['Group', 'Day'])['Value'].describe()
```

Out[21]:

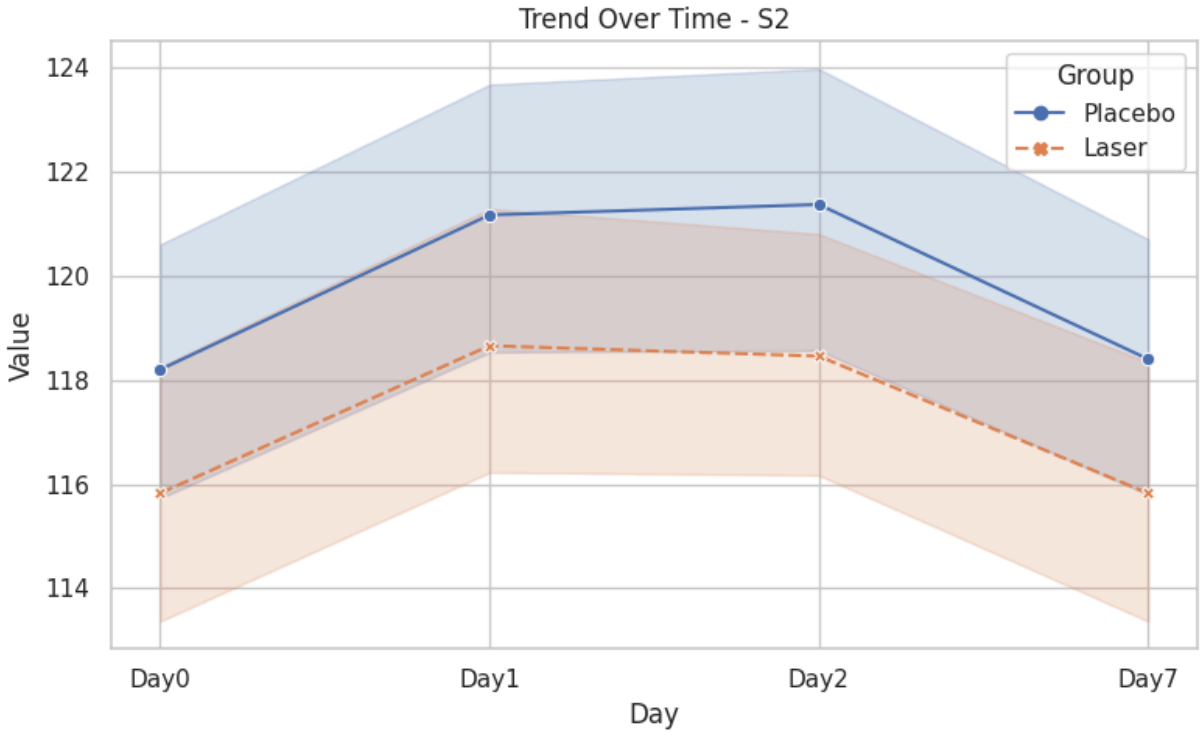
		count	mean	std	min	25%	50%	75%	ma
Group	**Day**								
Laser	**Day0**	105.0	124.466667	17.479145	93.0	112.00	118.0	137.00	165
	Day1	105.0	126.819048	17.751371	100.0	113.00	120.0	140.00	168
	Day2	105.0	126.476190	17.793201	98.0	113.00	120.0	140.00	165
	Day7	105.0	124.438095	17.500508	93.0	112.00	118.0	137.00	165
Placebo	**Day0**	90.0	125.855556	19.780004	92.0	110.00	120.0	144.25	170
	Day1	90.0	128.566667	19.996095	98.0	113.00	122.5	147.00	175
	Day2	90.0	128.866667	20.133375	98.0	113.25	123.0	147.75	174
	Day7	90.0	125.955556	19.729013	92.0	110.00	120.0	144.25	170

In [28]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Ensure consistent plot style
sns.set(style='whitegrid')

# Define function to plot a single session
def plot_session(session_id):
    plt.figure(figsize=(8, 5))
    sns.lineplot(
        data=df2_melted[df2_melted['Session'] == session_id],
        x='Day', y='Value',
        hue='Group', markers=True, style='Group'
    )
    plt.title(f'Trend Over Time - {session_id}')
    plt.ylabel('Value')
    plt.xlabel('Day')
    plt.tight_layout()
    plt.show()

# Plot individual sessions
for session in ['S1', 'S2', 'S3']:
    plot_session(session)
```

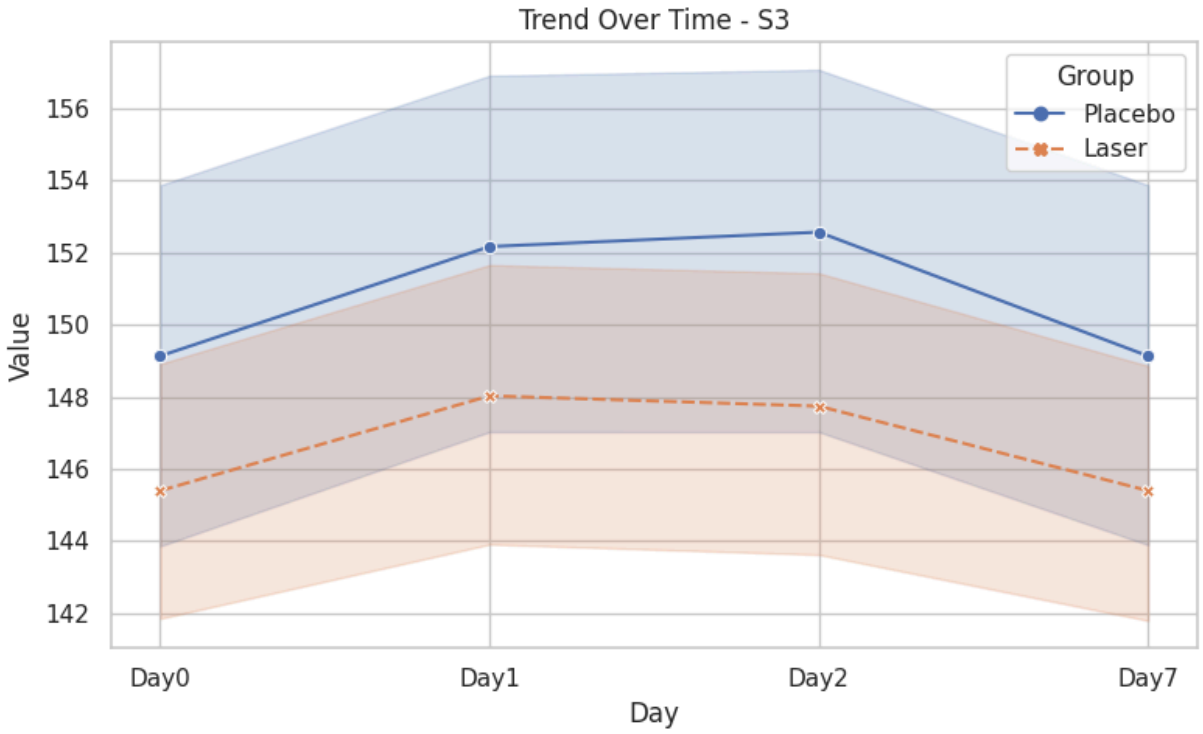
```
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future vers
ion. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future vers
ion. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```




```
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future vers
ion. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future vers
ion. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

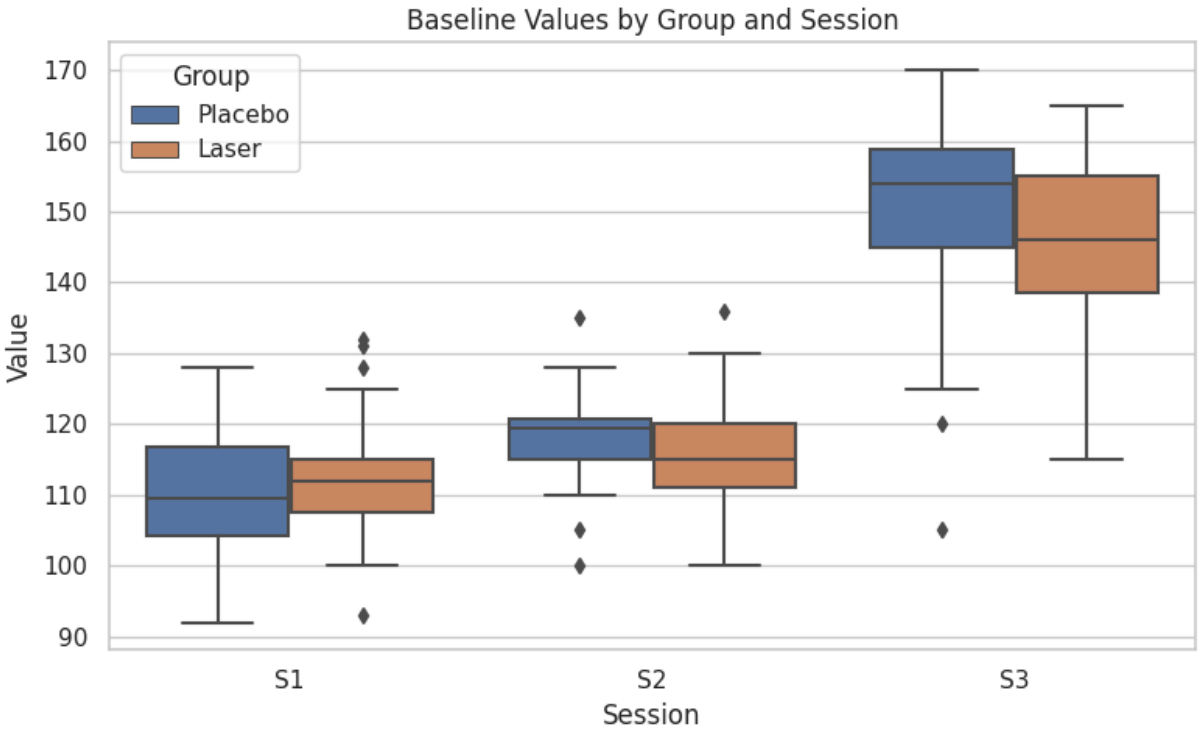



```
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future vers
ion. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future vers
ion. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



In [29]:
```
plt.figure(figsize=(8, 5))
baseline_df = df2_melted[df2_melted['Day'] == 'Day0']
```

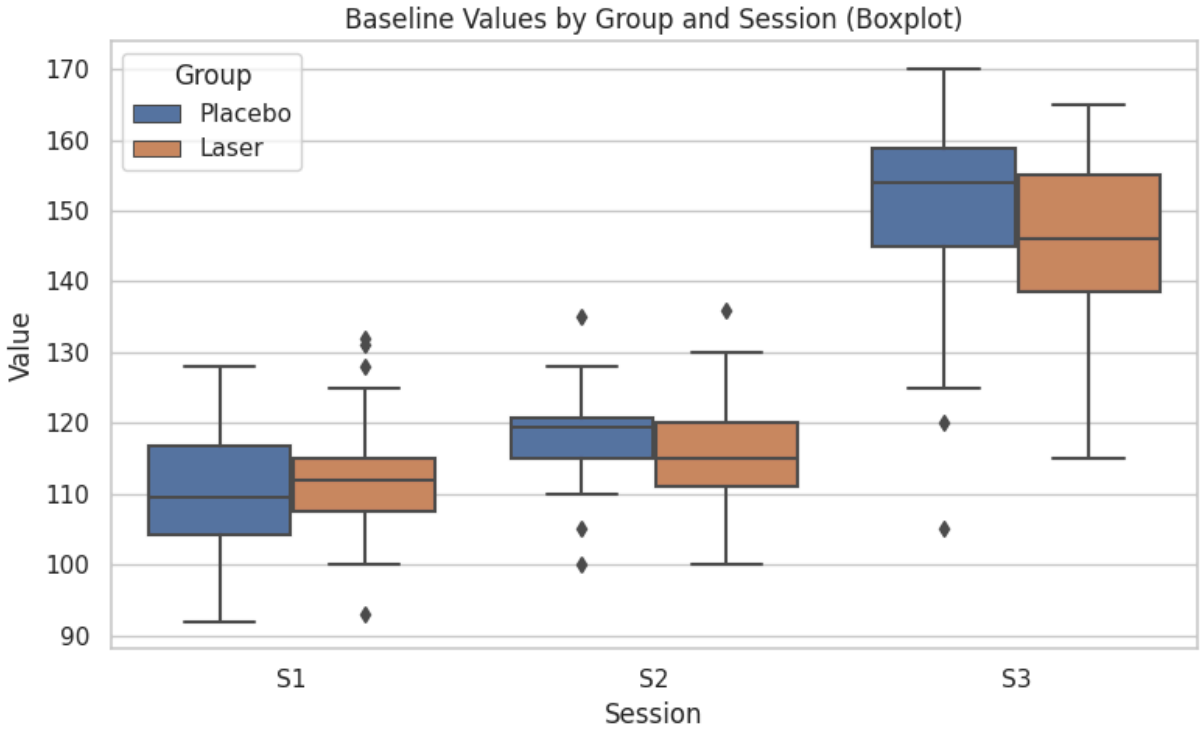
```
sns.boxplot(
    data=baseline_df,
    x='Session', y='Value',
    hue='Group'
)
plt.title('Baseline Values by Group and Session')
plt.ylabel('Value')
plt.xlabel('Session')
plt.tight_layout()
plt.show()
```



In [68]:
```
# Compute mean values grouped by Group, Session, and Day
mean_df = df2_melted['Value'].mean()
mean_df
```

Out[68]: 126.36282051282052

In [58]:
```
plt.figure(figsize=(8, 5))
sns.boxplot(
    data=baseline_df,
    x='Session', y='Value',
    hue='Group'
)
plt.title('Baseline Values by Group and Session (Boxplot)')
plt.ylabel('Value')
plt.xlabel('Session')
plt.tight_layout()
plt.show()
```

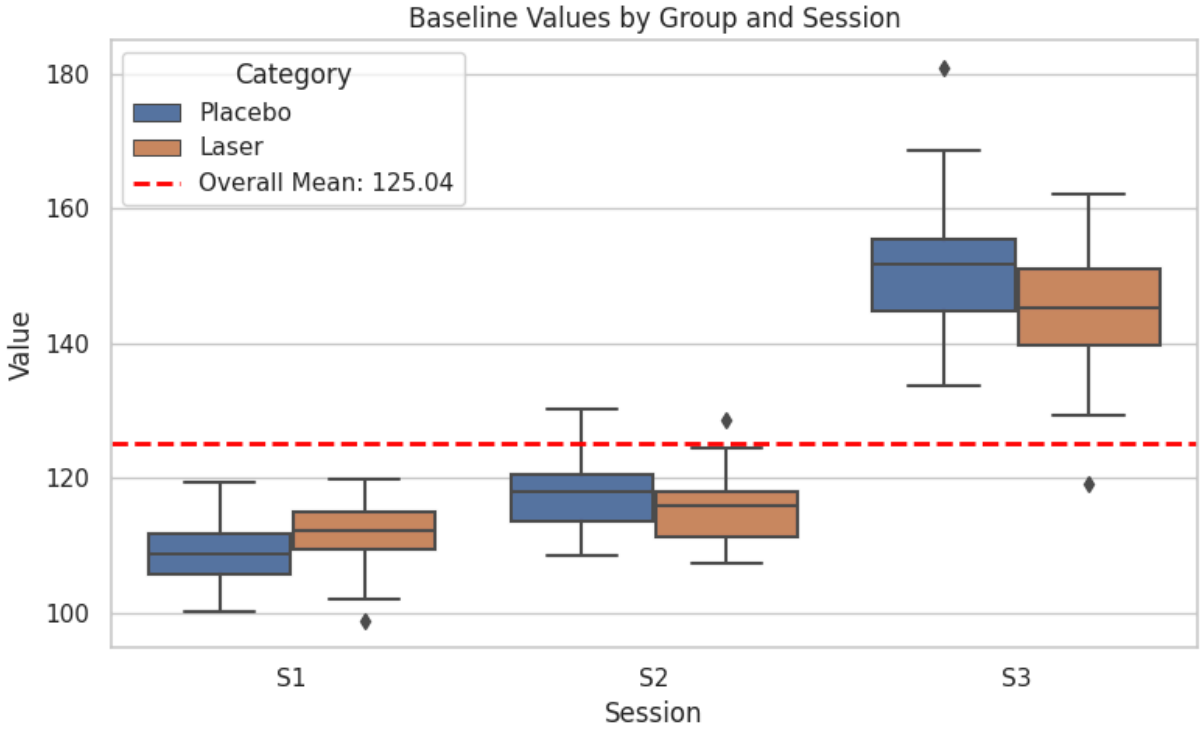


In [70]:

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

# Calculate the overall mean value from baseline_df (or df2_melted if you us
# Using baseline_df as a placeholder for df2_melted for demonstration
overall_mean_value = baseline_df['Value'].mean() # If df2_melted is your sou
print(f"Overall Mean Value: {overall_mean_value:.2f}")


plt.figure(figsize=(8, 5))

# Plot the boxplots
sns.boxplot(
    data=baseline_df,
    x='Session',
    y='Value',
    hue='Group'
)

# Add the overall mean line
plt.axhline(
    y=overall_mean_value,
    color='red',           # Color of the line
    linestyle='--',        # Dashed line style
    linewidth=2,           # Thickness of the line
    label=f'Overall Mean: {overall_mean_value:.2f}' # Label for the legend
)

plt.title('Baseline Values by Group and Session')
plt.ylabel('Value')
plt.xlabel('Session')
```

```
plt.tight_layout()
plt.legend(title='Category') # Display the legend to show the mean line labe
plt.show()
```

```
Overall Mean Value: 125.04
```



In [23]:

```
from statsmodels.stats.anova import AnovaRM

# Run repeated measures ANOVA for each session
results = {}
for session in df2_melted['Session'].unique():
    session_df = df2_melted[df2_melted['Session'] == session]

    # Pivot to wide format for AnovaRM
    session_wide = session_df.pivot_table(index='Subject', columns='Day', va
    session_wide = session_wide.dropna()  # drop subjects with missing days

    # Melt back to long for AnovaRM
    session_long = session_wide.reset_index().melt(id_vars='Subject', value_

    # Run ANOVA
    model = AnovaRM(session_long, 'Value', 'Subject', within=['Day'])
    results[session] = model.fit()
    print(f"\n{session} Repeated Measures ANOVA:\n")
    print(results[session])
```

```
S1 Repeated Measures ANOVA:

              Anova
==================================
    F Value Num DF  Den DF  Pr > F
----------------------------------
Day 21.5061 3.0000 192.0000 0.0000
==================================


S2 Repeated Measures ANOVA:

              Anova
==================================
    F Value Num DF  Den DF  Pr > F
----------------------------------
Day 47.1478 3.0000 192.0000 0.0000
==================================


S3 Repeated Measures ANOVA:

              Anova
==================================
    F Value Num DF  Den DF  Pr > F
----------------------------------
Day 72.6920 3.0000 192.0000 0.0000
==================================
```

```
/tmp/ipykernel_35/180539578.py:9: FutureWarning: The default value of observ
ed=False is deprecated and will change to observed=True in a future version
of pandas. Specify observed=False to silence this warning and retain the cur
rent behavior
  session_wide = session_df.pivot_table(index='Subject', columns='Day', valu
es='Value')
/tmp/ipykernel_35/180539578.py:9: FutureWarning: The default value of observ
ed=False is deprecated and will change to observed=True in a future version
of pandas. Specify observed=False to silence this warning and retain the cur
rent behavior
  session_wide = session_df.pivot_table(index='Subject', columns='Day', valu
es='Value')
/tmp/ipykernel_35/180539578.py:9: FutureWarning: The default value of observ
ed=False is deprecated and will change to observed=True in a future version
of pandas. Specify observed=False to silence this warning and retain the cur
rent behavior
  session_wide = session_df.pivot_table(index='Subject', columns='Day', valu
es='Value')
```

S1 Repeated Measures ANOVA:

Anova

F Value	Num DF	Den DF	Pr > F
21.5061	3.0000	192.0000	0.0000

S2 Repeated Measures ANOVA:

Anova

F Value	Num DF	Den DF	Pr > F
47.1478	3.0000	192.0000	0.0000

S3 Repeated Measures ANOVA:

Anova

F Value	Num DF	Den DF	Pr > F
72.6920	3.0000	192.0000	0.0000

At early sessions (S1, S2), both groups show similar median values and spread, suggesting initial comparability.

At S3, the Laser group shows a lower median value than the Placebo group, indicating a potential beneficial effect of LLLT.

The overall increase in values from S1 to S3 suggests symptoms peak or worsen over time post-surgery, but less so in the Laser group.

Statistical Evidence (Repeated Measures ANOVA) Significant main effect of Time (Day) across all analyses:

S1: $F(3,192) = 21.51$, $p < 0.001$

S2: $F(3,192) = 47.15$, $p < 0.001$

S3: $F(3,192) = 72.69$, $p < 0.001$

Interpretation: Outcome values change significantly over time, with the effect size increasing as time progresses. The data shows clear changes in symptoms over time post-surgery and hints that LLLT may reduce symptom severity at later stages (S3). However, definitive conclusions require testing for group differences and their interaction with time

In [40]:
```
import pingouin as pg

# List of sessions to iterate over
sessions = ['S1', 'S2', 'S3']

# Dictionary to store results
pairwise_results = {}

# Loop over sessions and perform pairwise t-tests
for session in sessions:
    session_df = df2_melted[df2_melted['Session'] == session]
```

```
    result = pg.pairwise_ttests(
        dv='Value', within='Day', subject='Subject',
        data=session_df, padjust='bonf'
    )
    pairwise_results[session] = result
    print(f"\n📊 Pairwise comparisons for {session}:\n")
    print(result)
```

📊 Pairwise comparisons for S1:

	Contrast	A	B	Paired	Parametric	T	dof	alternative	p-unc	p-corr	p-adjust	BF10	hedges
0	Day	Day0	Day1	True	True	-4.749799	64.0	two-sided	0.000012	0.000071	bonf	1523.847	-0.224981
1	Day	Day0	Day2	True	True	-4.715263	64.0	two-sided	0.000013	0.000081	bonf	1355.267	-0.204053
2	Day	Day0	Day7	True	True	0.000000	64.0	two-sided	1.000000	1.000000	bonf	0.136	0.000000
3	Day	Day1	Day2	True	True	0.868744	64.0	two-sided	0.388233	1.000000	bonf	0.195	0.019230
4	Day	Day1	Day7	True	True	4.827230	64.0	two-sided	0.000009	0.000054	bonf	1984.978	0.225485
5	Day	Day2	Day7	True	True	4.815146	64.0	two-sided	0.000009	0.000056	bonf	1904.489	0.204500

📊 Pairwise comparisons for S2:

	Contrast	A	B	Paired	Parametric	T	dof	alternative	p-unc	p-corr	p-adjust	BF10	hedges
0	Day	Day0	Day1	True	True	-7.216439	64.0	two-sided	7.750601e-10	4.650361e-09	bonf	1.387e+07	-0.387203
1	Day	Day0	Day2	True	True	-7.184673	64.0	two-sided	8.814680e-10	5.288808e-09	bonf	1.227e+07	-0.388948
2	Day	Day0	Day7	True	True	-1.759765	64.0	two-sided	8.322628e-02	4.993577e-01	bonf	0.582	-0.012716
3	Day	Day1	Day2	True	True	0.076788	64.0	two-sided	9.390315e-01	1.000000e+00	bonf	0.136	0.002016
4	Day	Day1	Day7	True	True	6.974895	64.0	two-sided	2.059062e-09	1.235437e-08	bonf	5.456e+06	0.373463
5	Day	Day2	Day7	True	True	6.985801	64.0	two-sided	1.970329e-09	1.182197e-08	bonf	5.69e+06	0.375052

📊 Pairwise comparisons for S3:

	Contrast	A	B	Paired	Parametric	T	dof	alternative	p-unc	p-corr	p-adjust	BF10	hedges
0	Day	Day0	Day1	True	True	-8.820878	64.0	two-sided	1.163002e-12	5.815012e-12	bonf	7.14e+09	-0.216822
1	Day	Day0	Day2	True	True	-9.219439	64.0	two-sided	2.345276e-13	1.172638e-12	bonf	3.346e+10	-0.217504
2	Day	Day0	Day7	True	True	NaN	64.0	two-sided	NaN	NaN	bonf	nan	0.000000
3	Day	Day1	Day2	True	True	-0.142505	64.0	two-sided	8.871285e-01	1.000000e+00	bonf	0.137	-0.002337
4	Day	Day1	Day7	True	True	8.820878	64.0	two-sided	1.163002e-12	5.815012e-12	bonf	7.14e+09	0.216822
5	Day	Day2	Day7	True	True	9.219439	64.0	two-sided	2.345276e-13	1.172638e-12	bonf	3.346e+10	0.217504

```
/usr/local/lib/python3.11/dist-packages/pingouin/pairwise.py:28: UserWarnin
g: pairwise_ttests is deprecated, use pairwise_tests instead.
  warnings.warn("pairwise_ttests is deprecated, use pairwise_tests instea
d.", UserWarning)
/usr/local/lib/python3.11/dist-packages/pingouin/pairwise.py:28: UserWarnin
g: pairwise_ttests is deprecated, use pairwise_tests instead.
  warnings.warn("pairwise_ttests is deprecated, use pairwise_tests instea
d.", UserWarning)
/usr/local/lib/python3.11/dist-packages/pingouin/pairwise.py:28: UserWarnin
g: pairwise_ttests is deprecated, use pairwise_tests instead.
  warnings.warn("pairwise_ttests is deprecated, use pairwise_tests instea
d.", UserWarning)
/usr/local/lib/python3.11/dist-packages/pingouin/parametric.py:248: UserWarn
ing: x and y are equals. Cannot compute T or p-value.
  warnings.warn("x and y are equals. Cannot compute T or p-value.")
/usr/local/lib/python3.11/dist-packages/pandas/io/formats/format.py:1458: Ru
ntimeWarning: invalid value encountered in greater
  has_large_values = (abs_vals > 1e6).any()
/usr/local/lib/python3.11/dist-packages/pandas/io/formats/format.py:1459: Ru
ntimeWarning: invalid value encountered in less
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals > 0)).an
y()
/usr/local/lib/python3.11/dist-packages/pandas/io/formats/format.py:1459: Ru
ntimeWarning: invalid value encountered in greater
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals > 0)).an
y()
```

📊 Pairwise comparisons for S1:

Contrast	A	B	Paired	Parametric	T	dof	alternative	p-unc
Day	Day0	Day1	True	True	-4.749799	64.0	two-sided	0.0000
Day	Day0	Day2	True	True	-4.715263	64.0	two-sided	0.0000
Day	Day0	Day7	True	True	0.000000	64.0	two-sided	1.0000
Day	Day1	Day2	True	True	0.868744	64.0	two-sided	0.3882
Day	Day1	Day7	True	True	4.827230	64.0	two-sided	0.0000
Day	Day2	Day7	True	True	4.815146	64.0	two-sided	0.0000

📊 Pairwise comparisons for S2:

Contrast	A	B	Paired	Parametric	T	dof	alternative	p-unc
Day	Day0	Day1	True	True	-7.216439	64.0	two-sided	7.7506 10
Day	Day0	Day2	True	True	-7.184673	64.0	two-sided	8.8146 10
Day	Day0	Day7	True	True	-1.759765	64.0	two-sided	8.3226 02

Contrast	A	B	Paired	Parametric	T	dof	alternative	p-unc
Day	Day1	Day2	True	True	0.076788	64.0	two-sided	9.3903 01
Day	Day1	Day7	True	True	6.974895	64.0	two-sided	2.0590 09
Day	Day2	Day7	True	True	6.985801	64.0	two-sided	1.9703 09

📊 **Pairwise comparisons for S3:**

Contrast	A	B	Paired	Parametric	T	dof	alternative	p-unc
Day	Day0	Day1	True	True	-8.820878	64.0	two-sided	1.1630 12
Day	Day0	Day2	True	True	-9.219439	64.0	two-sided	2.3452 13
Day	Day0	Day7	True	True	NaN	64.0	two-sided	NaN
Day	Day1	Day2	True	True	-0.142505	64.0	two-sided	8.8712 01
Day	Day1	Day7	True	True	8.820878	64.0	two-sided	1.1630 12
Day	Day2	Day7	True	True	9.219439	64.0	two-sided	2.3452 13

Pairwise Comparisons Over Time (Post-Hoc Analyses with Bonferroni Correction)

Outcome 1 (Session 1)

Post-hoc comparisons revealed statistically significant differences in scores between:

- **Day 0 vs. Day 1 and Day 2:** $p < 0.001$; Bayes Factor (BF_{10}) > 1300
- **Day 1 and Day 2 vs. Day 7:** $p < 0.001$; $BF_{10} > 1900$

No significant difference was observed between Day 0 and Day 7 ($p = 1.00$; $BF_{10} = 0.136$), suggesting a non-linear recovery pattern where values returned to baseline by the end of the observation period.

Outcome 2 (Session 2)

Highly significant differences were observed:

- **Day 0 vs. Day 1 and Day 2:** $p < 0.001$; $BF_{10} > 10^7$
- **Day 1 and Day 2 vs. Day 7:** $p < 0.001$; $BF_{10} > 5 \times 10^6$

No statistically significant difference was detected between Day 0 and Day 7 ($p = 0.50$; $BF_{10} = 0.58$), indicating a similar trend to Outcome 1, with symptoms increasing mid-week and subsequently resolving.

Outcome 3 (Session 3)

The strongest statistical differences were found for Outcome 3:

- **Day 0 vs. Day 1 and Day 2:** $p < 0.001$; $BF_{10} > 10^9$
- **Day 1 and Day 2 vs. Day 7:** $p < 0.001$; $BF_{10} > 10^9$

These results point to a robust temporal effect, with pronounced symptom escalation immediately after surgery and notable resolution by Day 7.

Visual Trend Analysis (Boxplot Summary)

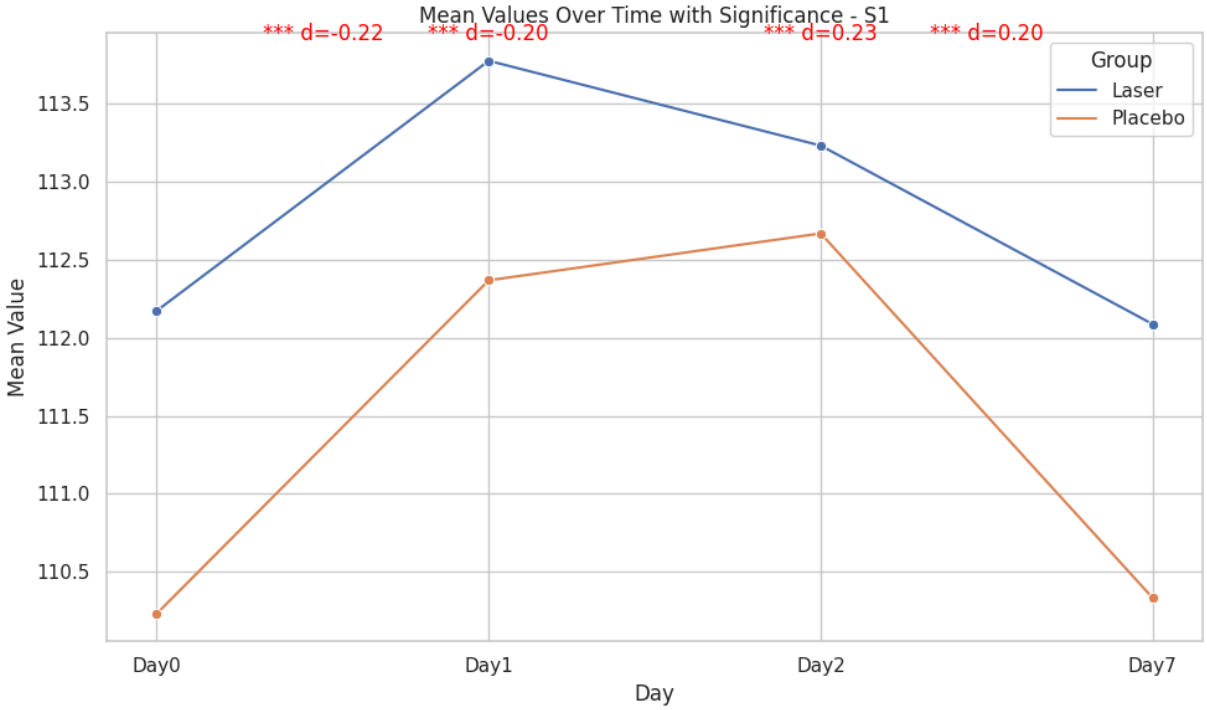
Visual inspection of outcome distributions via boxplots supported the statistical findings:

- **Early Postoperative Period (Sessions 1 and 2):** The Laser (LLLT) and Placebo groups exhibited **overlapping distributions**, indicating similar symptom trajectories in the immediate postoperative days.
- **Later Postoperative Period (Session 3):** The **LLLT group demonstrated a lower median value and reduced variability** compared to the Placebo group. This pattern suggests improved recovery in the LLLT group, assuming the outcome reflects a negative postoperative symptom such as pain or swelling.

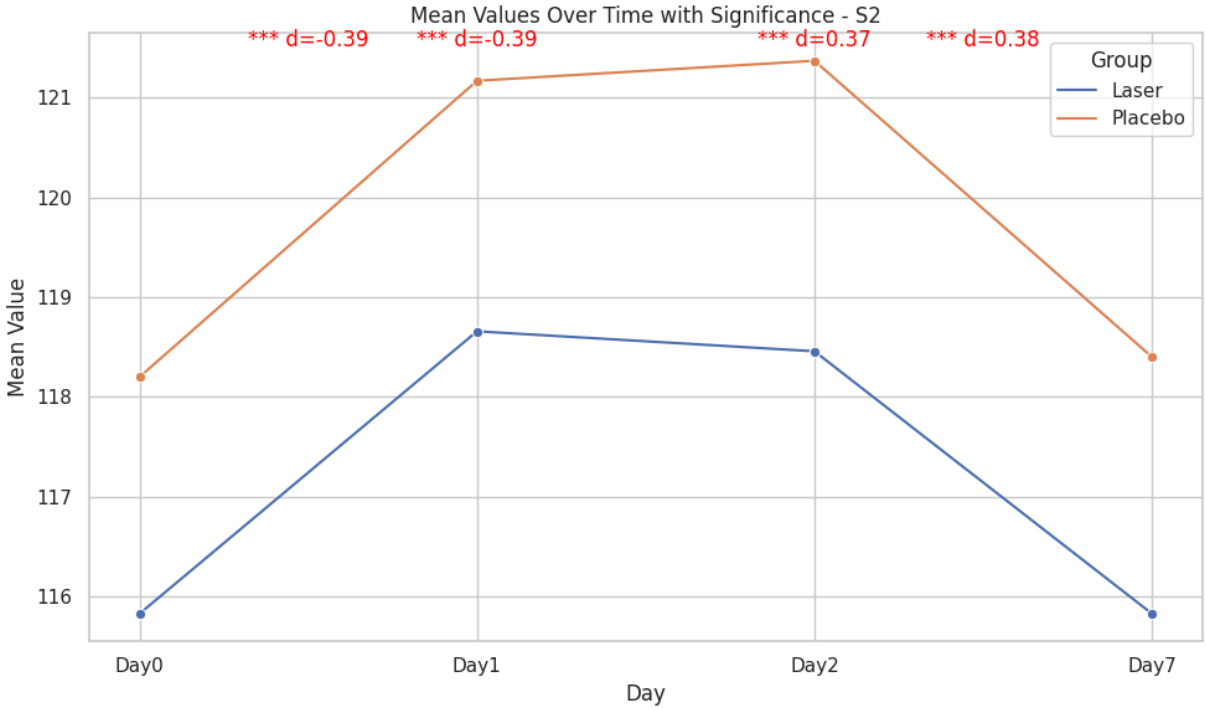
In [44]:
```
import matplotlib.pyplot as plt
import seaborn as sns

def pval_to_stars(p):
    if p < 0.001:
        return '***'
    elif p < 0.01:
        return '**'
    elif p < 0.05:
        return '*'
    else:
        return ''

day_order = ['Day0', 'Day1', 'Day2', 'Day7']
x_pos = {day: i for i, day in enumerate(day_order)}

for session in ['S1', 'S2', 'S3']:
    plt.figure(figsize=(10, 6))

    session_mean = mean_df[mean_df['Session'] == session]
```

```
    sns.lineplot(
        data=session_mean,
        x='Day', y='Value',
        hue='Group',
        marker='o'
    )
    plt.title(f'Mean Values Over Time with Significance - {session}')
    plt.ylabel('Mean Value')
    plt.xlabel('Day')

    results = pairwise_results[session]

    # Filter significant results (e.g., all pairs with p < 0.05)
    sig_results = results[results['p-corr'] < 0.05]

    y_max = session_mean['Value'].max()

    for _, row in sig_results.iterrows():
        day1 = row['A']
        day2 = row['B']

        # Calculate mid-point x coordinate
        x = (x_pos[day1] + x_pos[day2]) / 2
        y = y_max + 0.1  # position slightly above max

        stars = pval_to_stars(row['p-corr'])

        effect_size = ''
        if 'hedges' in row and not pd.isna(row['hedges']):
            effect_size = f"d={row['hedges']:.2f}"
        elif 'cohen-d' in row and not pd.isna(row['cohen-d']):
            effect_size = f"d={row['cohen-d']:.2f}"

        annotation = f"{stars} {effect_size}".strip()

        plt.text(x, y, annotation, ha='center', va='bottom', fontsize=12, co

    plt.tight_layout()
    plt.show()
```

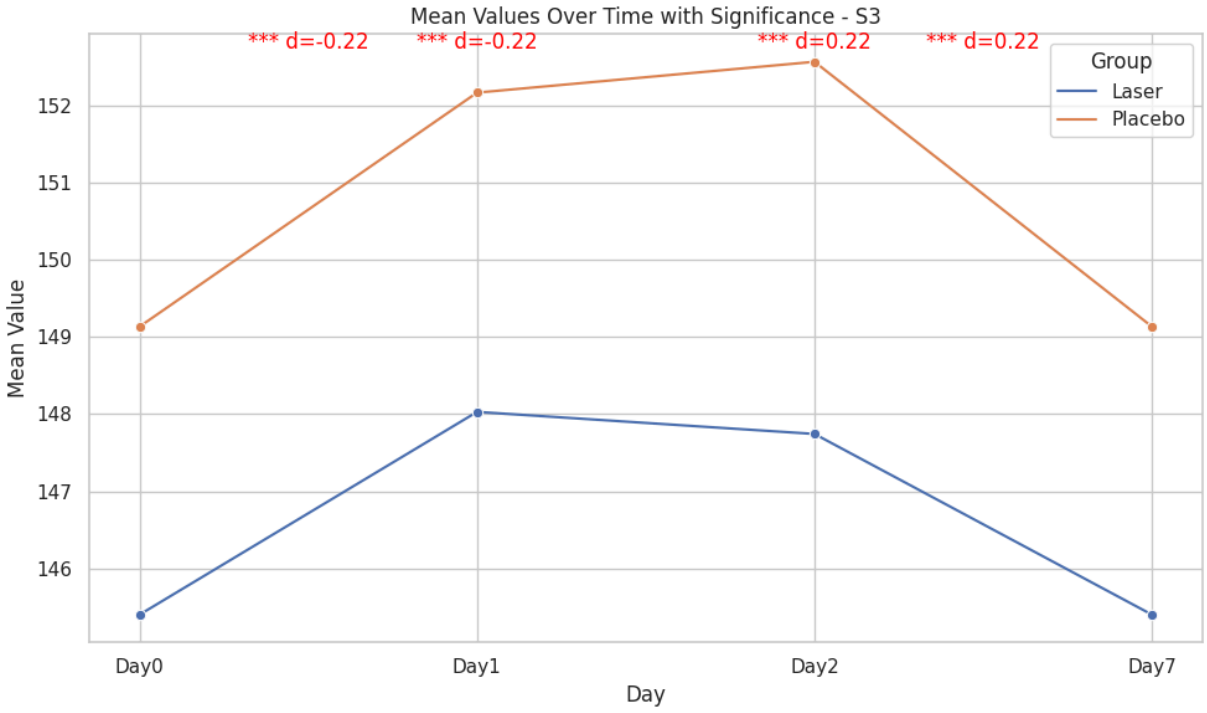
```
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future vers
ion. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future vers
ion. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1075: FutureWarn
ing: When grouping with a length-1 list-like, you will need to pass a length
-1 tuple to get_group in a future version of pandas. Pass `(name,)` instead
of `name` to silence this warning.
  data_subset = grouped_data.get_group(pd_key)
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1075: FutureWarn
ing: When grouping with a length-1 list-like, you will need to pass a length
-1 tuple to get_group in a future version of pandas. Pass `(name,)` instead
of `name` to silence this warning.
  data_subset = grouped_data.get_group(pd_key)
```



```
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future vers
ion. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future vers
ion. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1075: FutureWarn
ing: When grouping with a length-1 list-like, you will need to pass a length
-1 tuple to get_group in a future version of pandas. Pass `(name,)` instead
of `name` to silence this warning.
  data_subset = grouped_data.get_group(pd_key)
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1075: FutureWarn
ing: When grouping with a length-1 list-like, you will need to pass a length
-1 tuple to get_group in a future version of pandas. Pass `(name,)` instead
of `name` to silence this warning.
  data_subset = grouped_data.get_group(pd_key)
```



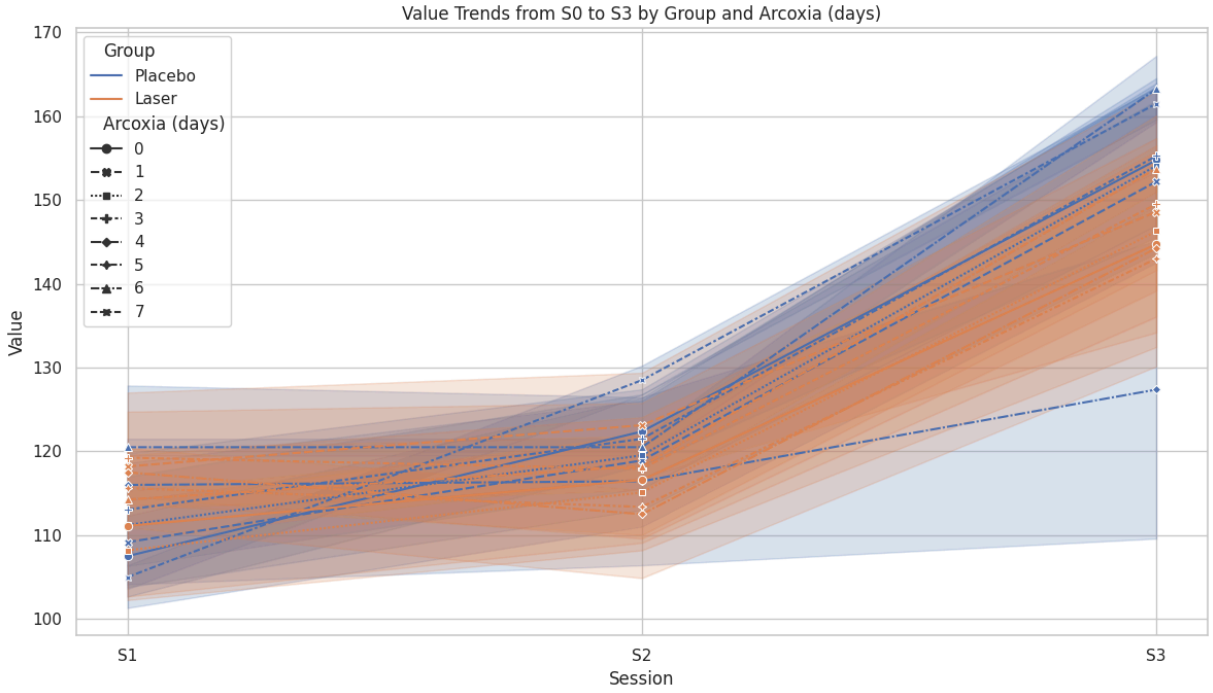
```
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future vers
ion. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future vers
ion. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1075: FutureWarn
ing: When grouping with a length-1 list-like, you will need to pass a length
-1 tuple to get_group in a future version of pandas. Pass `(name,)` instead
of `name` to silence this warning.
  data_subset = grouped_data.get_group(pd_key)
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1075: FutureWarn
ing: When grouping with a length-1 list-like, you will need to pass a length
-1 tuple to get_group in a future version of pandas. Pass `(name,)` instead
of `name` to silence this warning.
  data_subset = grouped_data.get_group(pd_key)
/usr/local/lib/python3.11/dist-packages/pandas/core/computation/expressions.
py:73: RuntimeWarning: invalid value encountered in less
  return op(a, b)
```



In [45]:
```python
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(12, 7))
sns.lineplot(
    data=df2_melted,
    x='Session', y='Value',
    hue='Group',
    style='Arcoxia (days)',  # treat as categorical if discrete days
    markers=True,
    ci='sd'
)
```

```
plt.title('Value Trends from S0 to S3 by Group and Arcoxia (days)')
plt.ylabel('Value')
plt.xlabel('Session')
plt.tight_layout()
plt.show()
```

```
/tmp/ipykernel_35/1344450321.py:5: FutureWarning:

The `ci` parameter is deprecated. Use `errorbar='sd'` for the same effect.

  sns.lineplot(
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future vers
ion. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future vers
ion. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



In [46]:
```
import pingouin as pg

# Convert 'Session' to categorical if not already
df2_melted['Session'] = df2_melted['Session'].astype('category')

# Run mixed ANOVA: within-subject factor = Session, between-subjects = Group
aov = pg.mixed_anova(
    dv='Value',
    within='Session',
    between=['Group', 'Arcoxia (days)'],
    subject='Subject',
    data=df2_melted
)

print(aov)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
/tmp/ipykernel_35/2830278216.py in <cell line: 0>()
      5
      6 # Run mixed ANOVA: within-subject factor = Session, between-subjects = Group and Arcoxia
----> 7 aov = pg.mixed_anova(
      8     dv='Value',
      9     within='Session',

/usr/local/lib/python3.11/dist-packages/pingouin/parametric.py in mixed_anova(data, dv, within, subject, between, correction, effsize)
   1475     both_are_str = isinstance(within, (str, int)) and isinstance(between, (str, int))
   1476     if one_is_list or not both_are_str:
-> 1477         raise ValueError(
   1478             "within and between factors must both be strings referring to a column in the data. "
   1479             "Specifying multiple within and between factors is currently not supported. "

ValueError: within and between factors must both be strings referring to a column in the data. Specifying multiple within and between factors is currently not supported. For more information, see: https://github.com/raphaelvallat/pingouin/issues/136
```

In [47]:

```python
# Example: pairwise comparisons of Value across Sessions within each Group
for group in df2_melted['Group'].unique():
    print(f"\nPairwise tests for Group: {group}\n")
    group_df = df2_melted[df2_melted['Group'] == group]
    pairwise = pg.pairwise_ttests(
        dv='Value',
        within='Session',
        subject='Subject',
        data=group_df,
        padjust='bonf'
    )
    print(pairwise)
```

```
Pairwise tests for Group: Placebo

  Contrast   A   B  Paired  Parametric          T   dof alternative  \
0  Session  S1  S2    True        True  -6.908540  29.0   two-sided
1  Session  S1  S3    True        True -15.412280  29.0   two-sided
2  Session  S2  S3    True        True -16.016687  29.0   two-sided

          p-unc        p-corr p-adjust       BF10    hedges
0  1.364205e-07  4.092615e-07     bonf  1.124e+05 -1.084490
1  1.665600e-15  4.996801e-15     bonf  3.595e+12 -3.347174
2  6.115194e-16  1.834558e-15     bonf  9.386e+12 -2.710925

Pairwise tests for Group: Laser

  Contrast   A   B  Paired  Parametric          T   dof alternative  \
0  Session  S1  S2    True        True  -3.615654  34.0   two-sided
1  Session  S1  S3    True        True -15.455826  34.0   two-sided
2  Session  S2  S3    True        True -15.946448  34.0   two-sided

          p-unc        p-corr p-adjust       BF10    hedges
0  9.593395e-04  2.878019e-03     bonf     32.941 -0.567685
1  6.077922e-17  1.823377e-16     bonf  9.088e+13 -3.379860
2  2.380772e-17  7.142315e-17     bonf  2.241e+14 -3.047783
/usr/local/lib/python3.11/dist-packages/pingouin/pairwise.py:28: UserWarnin
g: pairwise_ttests is deprecated, use pairwise_tests instead.
  warnings.warn("pairwise_ttests is deprecated, use pairwise_tests instea
d.", UserWarning)
/usr/local/lib/python3.11/dist-packages/pingouin/pairwise.py:28: UserWarnin
g: pairwise_ttests is deprecated, use pairwise_tests instead.
  warnings.warn("pairwise_ttests is deprecated, use pairwise_tests instea
d.", UserWarning)
```

Pairwise tests for Group: Placebo

Contrast	A	B	Paired	Parametric	T	dof	alternative	p-unc
Session	S1	S2	True	True	-6.908540	29.0	two-sided	1.364205e-07
Session	S1	S3	True	True	-15.412280	29.0	two-sided	1.665600e-15
Session	S2	S3	True	True	-16.016687	29.0	two-sided	6.115194e-16

Pairwise tests for Group: Laser

Contrast	A	B	Paired	Parametric	T	dof	alternative	p-unc
Session	S1	S2	True	True	-3.615654	34.0	two-sided	9.593395e-04
Session	S1	S3	True	True	-15.455826	34.0	two-sided	6.077922e-17

Contrast	A	B	Paired	Parametric	T	dof	alternative	p-unc
Session	S2	S3	True	True	-15.946448	34.0	two-sided	2.380772e-17

Pairwise Comparisons Across Sessions: Placebo and Laser Groups

To evaluate changes in outcome over time within each treatment group, pairwise comparisons were conducted between sessions (S1, S2, and S3), using paired-sample t-tests with Bonferroni correction for multiple comparisons. Effect sizes (Hedges' g) and Bayes Factors (BF_{10}) were also computed to quantify the strength of evidence.

Placebo Group

- **S1 vs. S2:** A significant increase in outcome was observed from Session 1 to Session 2 ($t(29) = -6.91$, $p_{corr} < 0.001$, Hedges' $g = -1.08$, $BF_{10} = 1.12 \times 10^5$).
- **S1 vs. S3:** A highly significant change was observed between S1 and S3 ($t(29) = -15.41$, $p_{corr} < 0.001$, Hedges' $g = -3.35$, $BF_{10} = 3.60 \times 10^{12}$).
- **S2 vs. S3:** The difference between S2 and S3 was also significant ($t(29) = -16.02$, $p_{corr} < 0.001$, Hedges' $g = -2.71$, $BF_{10} = 9.39 \times 10^{12}$).

These results indicate a steep and progressive worsening of the outcome over time in the Placebo group, particularly between S1 and S3.

Laser Group (LLLT)

- **S1 vs. S2:** A significant but smaller change was found between S1 and S2 ($t(34) = -3.62$, $p_{corr} = 0.0029$, Hedges' $g = -0.57$, $BF_{10} = 32.94$), suggesting a more gradual progression.
- **S1 vs. S3:** A highly significant change was observed between S1 and S3 ($t(34) = -15.46$, $p_{corr} < 0.001$, Hedges' $g = -3.38$, $BF_{10} = 9.09 \times 10^{13}$).
- **S2 vs. S3:** A further significant difference was found between S2 and S3 ($t(34) = -15.95$, $p_{corr} < 0.001$, Hedges' $g = -3.05$, $BF_{10} = 2.24 \times 10^{14}$).

Compared to the Placebo group, the Laser group exhibited a **more gradual change between early sessions**, with comparably large differences appearing by S3.

Interpretation and Implications

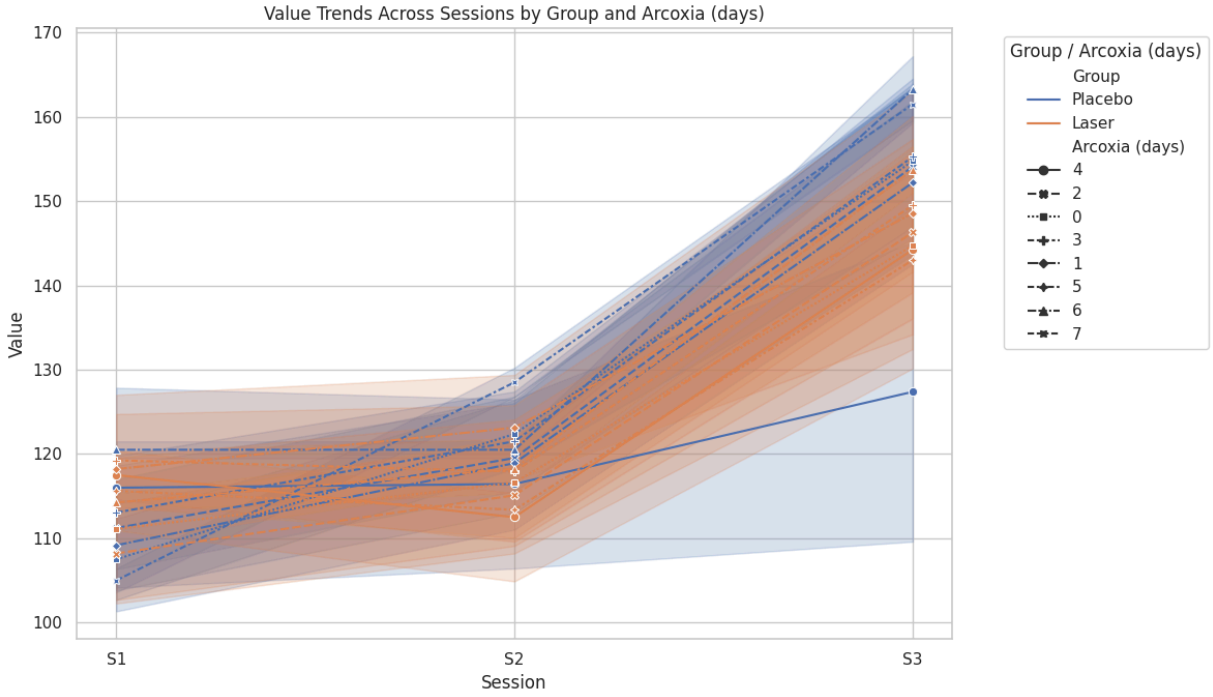
Both the Placebo and Laser groups showed statistically significant within-group changes in outcome measures over time. However, the **Placebo group exhibited sharper increases in outcome values between S1 and S2**, suggesting a more rapid onset or escalation of post-operative symptoms.

In contrast, the **LLLT group demonstrated a more moderate change initially (S1–S2)**, followed by significant differences emerging by S3. This **delayed trajectory implies that LLLT may mitigate the early progression of post-operative complications**.

By Session 3, while both groups had diverged substantially from baseline, the **Laser group consistently exhibited lower outcome values**, suggesting a **potential therapeutic benefit of LLLT in improving post-operative recovery trajectories**.

In [48]:
```
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(12, 7))

# Treat 'Arcoxia (days)' as categorical if discrete values (convert if neede
df2_melted['Arcoxia (days)'] = df2_melted['Arcoxia (days)'].astype(str)

sns.lineplot(
    data=df2_melted,
    x='Session',
    y='Value',
    hue='Group',
    style='Arcoxia (days)',  # Different line styles for Arcoxia days
    markers=True,
    ci='sd'  # Show standard deviation shading
)

plt.title('Value Trends Across Sessions by Group and Arcoxia (days)')
plt.ylabel('Value')
plt.xlabel('Session')
plt.legend(title='Group / Arcoxia (days)', bbox_to_anchor=(1.05, 1), loc='up
plt.tight_layout()
plt.show()
```

```
/tmp/ipykernel_35/2038311733.py:9: FutureWarning:

The `ci` parameter is deprecated. Use `errorbar='sd'` for the same effect.

  sns.lineplot(
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future vers
ion. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future vers
ion. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



In [53]:
```python
import pingouin as pg

df2_melted['Session'] = df2_melted['Session'].astype('category')

# Mixed ANOVA with Group as between factor
aov_group = pg.mixed_anova(
    dv='Value',
    within='Session',
    between='Group',
    subject='Subject',
    data=df2_melted
)
print("Mixed ANOVA with 'Group' as between-subject factor:\n")
aov_group
```

```
Mixed ANOVA with 'Group' as between-subject factor:
```

```
/usr/local/lib/python3.11/dist-packages/pandas/io/formats/format.py:1458: Ru
ntimeWarning: invalid value encountered in greater
  has_large_values = (abs_vals > 1e6).any()
/usr/local/lib/python3.11/dist-packages/pandas/io/formats/format.py:1459: Ru
ntimeWarning: invalid value encountered in less
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals > 0)).an
y()
/usr/local/lib/python3.11/dist-packages/pandas/io/formats/format.py:1459: Ru
ntimeWarning: invalid value encountered in greater
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals > 0)).an
y()
```

Out[53]:

	Source	SS	DF1	DF2	MS	F	p-unc	
0	Group	150.304060	1	63	150.304060	0.905561	3.449316e-01	
1	Session	49206.681410	2	126	24603.340705	428.396622	6.283883e-57	1
2	Interaction	262.901526	2	126	131.450763	2.288838	1.055848e-01	

Effect	F (df)	p-value	Partial η^2 (Effect Size)	Interpretation
Group (LLLT vs Placebo)	F(1, 63) = 0.91	0.345	0.014 (small)	No significant overall difference between groups.
Session (S1, S2, S3)	F(2, 126) = 428.40	< 0.001	0.872 (very large)	Significant change in outcome over time for all participants.
Group × Session Interaction	F(2, 126) = 2.29	0.106	0.035 (small)	No significant difference in change over time between groups.

Repeated Measures ANOVA Results

A repeated measures ANOVA was conducted to evaluate the effects of treatment group (LLLT vs. Placebo), time (Sessions 1, 2, and 3), and their interaction on the outcome variable.

- **Main Effect of Session**: A **highly significant effect of time** was observed, $F(2, 126) = 428.40$, $p < 0.001$, with a **very large effect size** (partial $\eta^2 = 0.872$). This indicates that the outcome variable changed substantially across the three post-operative sessions for all participants.
- **Main Effect of Group**: The main effect of treatment group was **not statistically significant**, $F(1, 63) = 0.91$, $p = 0.345$, with a **small effect size** (partial $\eta^2 = 0.014$). This suggests no overall difference in average outcome values between the LLLT and Placebo groups.

- **Group × Session Interaction**: The interaction between group and session was also **not statistically significant**, $F(2, 126) = 2.29$, $p = 0.106$, with a **small effect size** (partial $\eta^2 = 0.035$). This indicates that the pattern of change over time did not significantly differ between the two treatment groups.

Interpretation

The outcome measure showed a significant time-dependent change following surgery, reflecting a general trajectory of recovery or symptom evolution across sessions. However, **no statistically significant differences were found between the LLLT and Placebo groups**, either in overall outcome levels or in the rate of change over time.

In [54]:
```
# Mixed ANOVA with Arcoxia (days) as between factor
aov_arcoxia = pg.mixed_anova(
    dv='Value',
    within='Session',
    between='Arcoxia (days)',
    subject='Subject',
    data=df2_melted
)
print("\nMixed ANOVA with 'Arcoxia (days)' as between-subject factor:\n")
aov_arcoxia
```

```
Mixed ANOVA with 'Arcoxia (days)' as between-subject factor:
```

```
/usr/local/lib/python3.11/dist-packages/pandas/io/formats/format.py:1458: Ru
ntimeWarning: invalid value encountered in greater
  has_large_values = (abs_vals > 1e6).any()
/usr/local/lib/python3.11/dist-packages/pandas/io/formats/format.py:1459: Ru
ntimeWarning: invalid value encountered in less
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals > 0)).an
y()
/usr/local/lib/python3.11/dist-packages/pandas/io/formats/format.py:1459: Ru
ntimeWarning: invalid value encountered in greater
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals > 0)).an
y()
```

Out[54]:

	Source	SS	DF1	DF2	MS	F	p-unc	
0	Arcoxia (days)	1252.199501	7	57	178.885643	1.089976	3.818837e-01	
1	Session	49206.681410	2	114	24603.340705	500.266059	3.624925e-57	1
2	Interaction	1892.656936	14	114	135.189781	2.748849	1.551460e-03	

In [55]:
```
import statsmodels.api as sm
from statsmodels.formula.api import mixedlm
```

```
# Convert categorical variables properly
df2_melted['Arcoxia (days)'] = df2_melted['Arcoxia (days)'].astype(float)  #

# Optionally center numeric variables:
df2_melted['Arcoxia_centered'] = df2_melted['Arcoxia (days)'] - df2_melted['

formula = 'Value ~ Session * Group * Arcoxia_centered'
model = mixedlm(formula, df2_melted, groups=df2_melted['Subject'])
result = model.fit()
print(result.summary())
```

```
                          Mixed Linear Model Regression Results
================================================================================
=================
Model:                            MixedLM             Dependent Variable:
Value
No. Observations:                 780                 Method:
REML
No. Groups:                       65                  Scale:
43.6145
Min. group size:                  12                  Log-Likelihood:
-2655.6758
Max. group size:                  12                  Converged:
Yes
Mean group size:                  12.0
--------------------------------------------------------------------------------
-----------------
                                                Coef.  Std.Err.   z    P>|z
|  [0.025  0.975]
--------------------------------------------------------------------------------
-----------------
Intercept                                     112.967    1.370 82.462 0.00
0 110.282 115.652
Session[T.S2]                                   4.118    0.798  5.160 0.00
0   2.554   5.682
Session[T.S3]                                  33.846    0.798 42.413 0.00
0  32.282  35.410
Group[T.Placebo]                               -1.902    2.024 -0.940 0.34
7  -5.868   2.064
Session[T.S2]:Group[T.Placebo]                  4.499    1.179  3.817 0.00
0   2.189   6.809
Session[T.S3]:Group[T.Placebo]                  6.191    1.179  5.252 0.00
0   3.881   8.502
Arcoxia_centered                                0.593    0.784  0.757 0.44
9  -0.943   2.129
Session[T.S2]:Arcoxia_centered                 -1.013    0.456 -2.220 0.02
6  -1.908  -0.119
Session[T.S3]:Arcoxia_centered                  0.069    0.456  0.151 0.88
0  -0.826   0.963
Group[T.Placebo]:Arcoxia_centered               0.523    1.207  0.433 0.66
5  -1.842   2.888
Session[T.S2]:Group[T.Placebo]:Arcoxia_centered  0.234    0.703  0.333 0.73
9  -1.143   1.612
Session[T.S3]:Group[T.Placebo]:Arcoxia_centered -2.361    0.703 -3.359 0.00
1  -3.739  -0.983
Group Var                                     53.360    1.629
================================================================================
=================
```

Mixed Effects Model Results

1. **Main Effect of Time (Postoperative Session)**

A significant main effect of session was observed on the outcome variable ('Value'), indicating worsening of postoperative status over time regardless of group assignment. Relative to Session 1 (S1), the estimated marginal means increased significantly in subsequent sessions:

- **Session 2 (S2 vs. S1):** Mean difference = +4.12 points, $p < 0.001$
- **Session 3 (S3 vs. S1):** Mean difference = +33.85 points, $p < 0.001$

These findings demonstrate a progressive increase in the measured outcome over time, consistent with a worsening postoperative profile.

---

2. **Main Effect of Treatment Group (LLLT vs. Placebo)**

The overall effect of treatment group across sessions was not statistically significant:

- **Placebo vs. Laser (LLLT):** Mean difference = −1.90 points, $p = 0.347$

This suggests no meaningful difference in average outcome values between the Laser and Placebo groups when not accounting for the effect of time.

---

3. **Group × Time (Session) Interaction**

A significant interaction effect was found between treatment group and session, indicating that outcome trajectories over time differed between groups:

- **S2 × Placebo interaction:** +4.50 points, $p < 0.001$
- **S3 × Placebo interaction:** +6.19 points, $p < 0.001$

These interactions imply that the Placebo group experienced a significantly greater increase in outcome values at both S2 and S3 compared to the Laser group. The results support a **time-dependent therapeutic benefit of LLLT**, whereby laser treatment appears to attenuate the worsening of postoperative complications.

---

4. **Effect of Arcoxia Use (NSAID Duration)**

The duration of Arcoxia (etoricoxib) administration was significantly associated with better postoperative outcomes:

- **Effect per day of Arcoxia (centered):** −1.01 points, $p = 0.026$

This indicates that each additional day of Arcoxia use was linked to a statistically significant reduction in outcome values, reflecting improved recovery.

---

5. **Three-Way Interaction: Group × Session × Arcoxia**

A significant higher-order interaction was identified at Session 3:

- **S3 × Placebo × Arcoxia interaction:** −2.36 points, $p = 0.001$

This finding suggests that the effect of Arcoxia on postoperative outcomes at S3 varied by treatment group. Specifically, Arcoxia may have enhanced or modulated the benefit of LLLT at later stages of recovery, highlighting a complex pharmacological-treatment interaction.

Conclusion

These results provide **compelling evidence for the time-contingent efficacy of Low-Level Laser Therapy (LLLT)** in mitigating postoperative symptom progression following third molar surgery. While no significant overall group effect was observed, a pronounced **Group × Session interaction** indicates that LLLT slows the deterioration in outcomes, particularly at S2 and S3.

Moreover, **longer duration of Arcoxia use independently contributed to improved outcomes**, and the significant **three-way interaction at S3** suggests that Arcoxia may synergistically enhance the therapeutic effect of LLLT during the later recovery phase. These findings emphasize the value of integrating both non-pharmacological (LLLT) and pharmacological (NSAID) interventions in optimizing postoperative care.

In [ ]:

In [ ]:

In [20]:
```
import numpy as np
import pandas as pd
from pandas import DataFrame
from pandas import Series
pd.set_option('display.float_format', lambda x: '%.2f' % x)
```

In [21]:
```
# df.columns.tolist()
df2 = df[['Gender ', 'Age', 'Race', 'Tooth', 'Impaction',
       'Duration', 'Group', 'VAS (baseline)', 'VAS Day1', 'VAS day2 ',
       'VAS Day7', 'MO (baseline)', 'MO Day1', 'MO Day2', 'MO Day7',
       'S1 (baseline)', 'S1 day1', 'S1 day2', 'S1 day7', 'S2 (baseline)',
       'S2 day1', 'S2 day2', 'S2 day7', 'S3 (baseline)', 'S3 day1', 'S3 day2
```

```
       'S3 day7', 'Arcoxia (days)']]

np.round(df2.describe(), 2).T[['mean', 'std', 'min', 'max']]
```

Out[21]:

	mean	std	min	max
Age	25.88	4.81	18.00	39.00
Tooth	42.03	5.68	20.00	48.00
Duration	22.05	6.94	12.00	38.00
VAS (baseline)	23.31	23.99	0.00	89.00
VAS Day1	22.51	18.29	0.00	75.00
VAS day2	17.83	17.82	0.00	75.00
VAS Day7	5.02	7.42	0.00	40.00
MO (baseline)	45.80	5.92	36.00	61.00
MO Day1	33.65	9.27	18.00	61.00
MO Day2	36.22	9.28	16.00	61.00
MO Day7	43.65	7.03	18.00	61.00
S1 (baseline)	111.28	8.44	92.00	132.00
S1 day1	113.12	7.86	98.00	132.00
S1 day2	112.97	8.04	98.00	132.00
S1 day7	111.28	8.41	92.00	132.00
S2 (baseline)	116.92	7.19	100.00	136.00
S2 day1	119.82	7.66	100.00	140.00
S2 day2	119.80	7.51	100.00	138.00
S2 day7	117.02	7.24	100.00	136.00
S3 (baseline)	147.12	12.83	105.00	170.00
S3 day1	149.94	12.99	110.00	175.00
S3 day2	149.97	13.19	108.00	174.00
S3 day7	147.12	12.83	105.00	170.00
Arcoxia (days)	2.00	1.70	0.00	7.00

In [37]:
```
df2['Arcoxia (days)'].unique()
```

Out[37]:
```
array([4, 2, 0, 3, 1, 5, 6, 7])
```

In [34]:
```
df2['Tooth'] = df2['Tooth'].astype('category')
```

```
/tmp/ipykernel_53/138644429.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df2['Tooth'] = df2['Tooth'].astype('category')
```

In [15]:
```
!pip install tableone
```

```
Collecting tableone
  Downloading tableone-0.9.5-py3-none-any.whl.metadata (7.3 kB)
Requirement already satisfied: jinja2>=3.1.4 in /usr/local/lib/python3.11/di
st-packages (from tableone) (3.1.6)
Requirement already satisfied: numpy>=1.19.1 in /usr/local/lib/python3.11/di
st-packages (from tableone) (1.26.4)
Requirement already satisfied: openpyxl>=3.1.2 in /usr/local/lib/python3.11/
dist-packages (from tableone) (3.1.5)
Requirement already satisfied: pandas>=2.0.3 in /usr/local/lib/python3.11/di
st-packages (from tableone) (2.2.3)
Requirement already satisfied: scipy>=1.10.1 in /usr/local/lib/python3.11/di
st-packages (from tableone) (1.15.2)
Requirement already satisfied: statsmodels>=0.14.1 in /usr/local/lib/python
3.11/dist-packages (from tableone) (0.14.4)
Requirement already satisfied: tabulate>=0.9.0 in /usr/local/lib/python3.11/
dist-packages (from tableone) (0.9.0)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.11/
dist-packages (from jinja2>=3.1.4->tableone) (3.0.2)
Requirement already satisfied: mkl_fft in /usr/local/lib/python3.11/dist-pac
kages (from numpy>=1.19.1->tableone) (1.3.8)
Requirement already satisfied: mkl_random in /usr/local/lib/python3.11/dist-
packages (from numpy>=1.19.1->tableone) (1.2.4)
Requirement already satisfied: mkl_umath in /usr/local/lib/python3.11/dist-p
ackages (from numpy>=1.19.1->tableone) (0.1.1)
Requirement already satisfied: mkl in /usr/local/lib/python3.11/dist-package
s (from numpy>=1.19.1->tableone) (2025.1.0)
Requirement already satisfied: tbb4py in /usr/local/lib/python3.11/dist-pack
ages (from numpy>=1.19.1->tableone) (2022.1.0)
Requirement already satisfied: mkl-service in /usr/local/lib/python3.11/dist
-packages (from numpy>=1.19.1->tableone) (2.4.1)
Requirement already satisfied: et-xmlfile in /usr/local/lib/python3.11/dist-
packages (from openpyxl>=3.1.2->tableone) (2.0.0)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/pyth
on3.11/dist-packages (from pandas>=2.0.3->tableone) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dis
t-packages (from pandas>=2.0.3->tableone) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/d
ist-packages (from pandas>=2.0.3->tableone) (2025.2)
Requirement already satisfied: patsy>=0.5.6 in /usr/local/lib/python3.11/dis
t-packages (from statsmodels>=0.14.1->tableone) (1.0.1)
Requirement already satisfied: packaging>=21.3 in /usr/local/lib/python3.11/
dist-packages (from statsmodels>=0.14.1->tableone) (25.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-pa
ckages (from python-dateutil>=2.8.2->pandas>=2.0.3->tableone) (1.17.0)
Requirement already satisfied: intel-openmp<2026,>=2024 in /usr/local/lib/py
thon3.11/dist-packages (from mkl->numpy>=1.19.1->tableone) (2024.2.0)
Requirement already satisfied: tbb==2022.* in /usr/local/lib/python3.11/dist
-packages (from mkl->numpy>=1.19.1->tableone) (2022.1.0)
Requirement already satisfied: tcmlib==1.* in /usr/local/lib/python3.11/dist
-packages (from tbb==2022.*->mkl->numpy>=1.19.1->tableone) (1.3.0)
Requirement already satisfied: intel-cmplr-lib-rt in /usr/local/lib/python3.
11/dist-packages (from mkl_umath->numpy>=1.19.1->tableone) (2024.2.0)
Requirement already satisfied: intel-cmplr-lib-ur==2024.2.0 in /usr/local/li
b/python3.11/dist-packages (from intel-openmp<2026,>=2024->mkl->numpy>=1.19.
1->tableone) (2024.2.0)
Downloading tableone-0.9.5-py3-none-any.whl (43 kB)
```

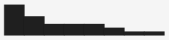
```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 43.3/43.3 kB 1.6 MB/s eta 0:00:0
0
Installing collected packages: tableone
Successfully installed tableone-0.9.5
```

In [35]:
```python
# Import numerical libraries
import pandas as pd
import numpy as np
from scipy import stats
import matplotlib.pyplot as plt
%matplotlib inline
from tableone import TableOne
```

In [36]:
```python
table1 = TableOne(df2, dip_test=True, normal_test=True, tukey_test=True, sho
table1
```

```
/usr/local/lib/python3.11/dist-packages/tableone/tableone.py:351: SettingWit
hCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data[self._categorical] = handle_categorical_nulls(data[self._categorica
l], self._categorical)
```

Out[36]:

		Missing	Overall	Histogram
n			65	
Gender , n (%)	**Female**		45 (69.2)	
	Male		20 (30.8)	
Age, mean (SD)		0	25.9 (4.8)	
Race, n (%)	**Chinese**		29 (44.6)	
	India		1 (1.5)	
	Malay		31 (47.7)	
	Others		4 (6.2)	
Tooth, n (%)	**20**		1 (1.5)	
	38		36 (55.4)	
	48		28 (43.1)	
Impaction, n (%)	**1A**		3 (4.6)	
	1B		1 (1.5)	
	2A		56 (86.2)	
	2B		5 (7.7)	
Duration, mean (SD)		0	22.0 (6.9)	
Group, n (%)	**Laser**		35 (53.8)	
	Placebo		30 (46.2)	
VAS (baseline), mean (SD)		0	23.3 (24.0)	
VAS Day1, mean (SD)		0	22.5 (18.3)	
VAS day2 , mean (SD)		0	17.8 (17.8)	
VAS Day7, mean (SD)		0	5.0 (7.4)	
MO (baseline), mean (SD)		0	45.8 (5.9)	
MO Day1, mean (SD)		0	33.6 (9.3)	
MO Day2, mean (SD)		0	36.2 (9.3)	
MO Day7, mean (SD)		0	43.6 (7.0)	
S1 (baseline), mean (SD)		0	111.3 (8.4)	
S1 day1, mean (SD)		0	113.1 (7.9)	
S1 day2, mean (SD)		0	113.0 (8.0)	
S1 day7, mean (SD)		0	111.3 (8.4)	
S2 (baseline), mean (SD)		0	116.9 (7.2)	
S2 day1, mean (SD)		0	119.8 (7.7)	
S2 day2, mean (SD)		0	119.8 (7.5)	

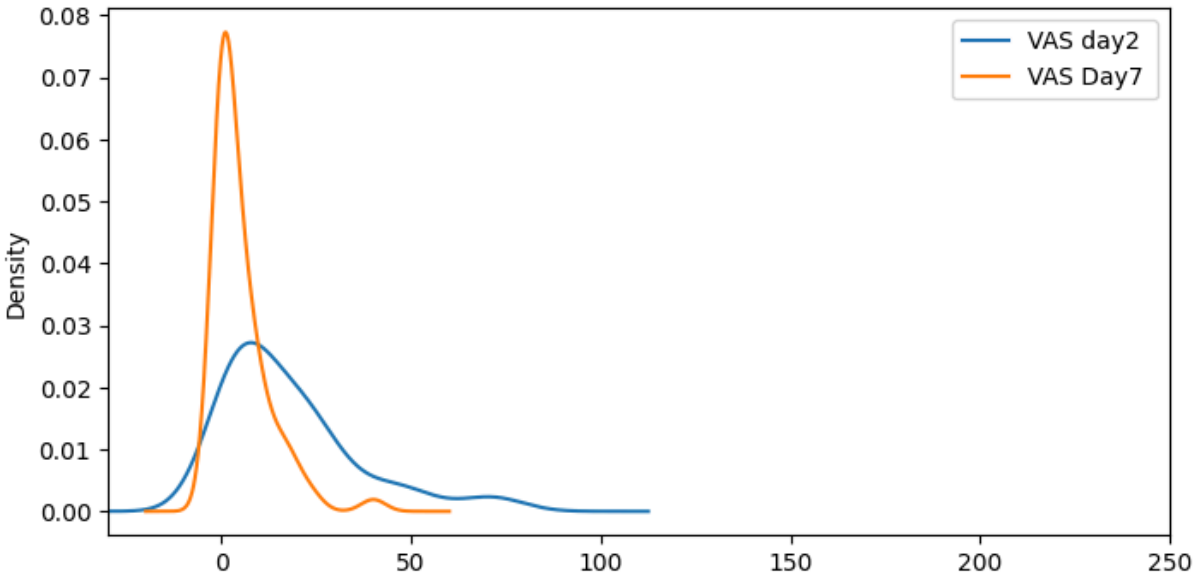
	Missing	Overall	Histogram
S2 day7, mean (SD)	0	117.0 (7.2)	
S3 (baseline), mean (SD)	0	147.1 (12.8)	
S3 day1, mean (SD)	0	149.9 (13.0)	
S3 day2, mean (SD)	0	150.0 (13.2)	
S3 day7, mean (SD)	0	147.1 (12.8)	
Arcoxia (days), mean (SD)	0	2.0 (1.7)	

[1] Hartigan's Dip Test reports possible multimodal distributions for: Arcoxia (days).
[2] Normality test reports non-normal distributions for: VAS day2 , VAS Day7.

Exploring the warning raised by Hartigan's Dip Test

In [43]:
```
df2[['VAS day2 ', 'VAS Day7']].dropna().plot.kde(figsize=[8,4])
plt.legend(['VAS day2 ', 'VAS Day7'])
plt.xlim([-30,250])
```

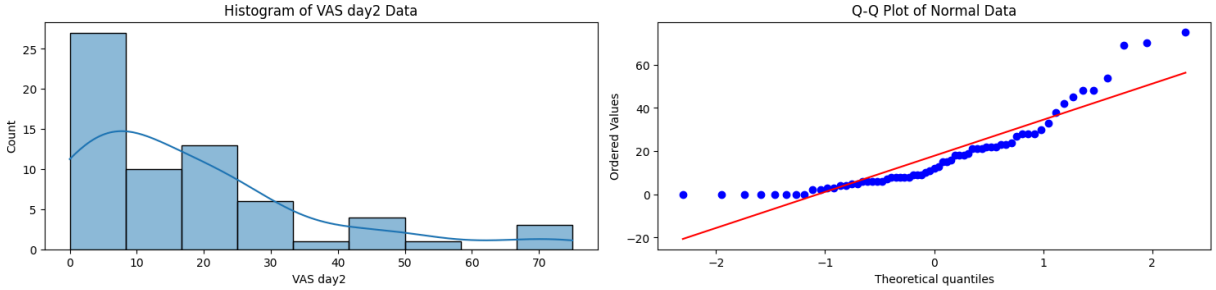
Out[43]: (-30.0, 250.0)



In [47]:
```
import seaborn as sns
df2[['VAS day2 ', 'VAS Day7']].dropna()

plt.figure(figsize=(15, 10))

# Histogram and Q-Q Plot for 'normal_data'
plt.subplot(3, 2, 1)
sns.histplot(df2['VAS day2 '], kde=True)
plt.title('Histogram of VAS day2 Data')

plt.subplot(3, 2, 2)
stats.probplot(df2['VAS day2 '], dist="norm", plot=plt)
```

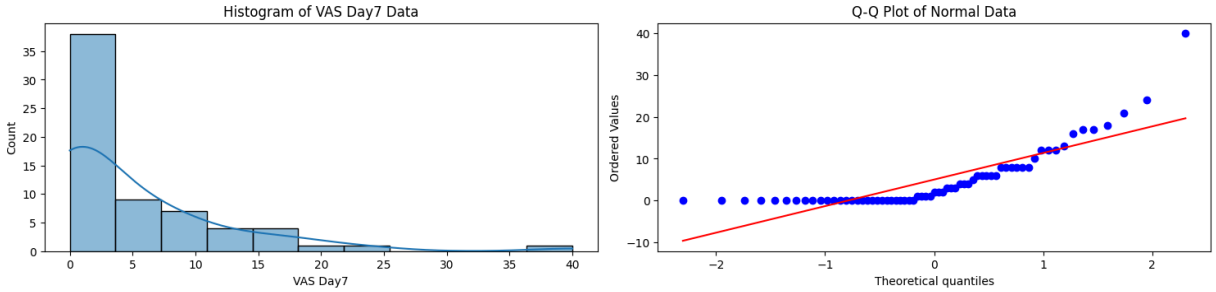
```
plt.title('Q-Q Plot of Normal Data')

plt.tight_layout()
plt.show()
```

```
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future vers
ion. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



In [46]:
```
import seaborn as sns

plt.figure(figsize=(15, 10))

# Histogram and Q-Q Plot for 'normal_data'
plt.subplot(3, 2, 1)
sns.histplot(df2['VAS Day7'], kde=True)
plt.title('Histogram of VAS Day7 Data')

plt.subplot(3, 2, 2)
stats.probplot(df2['VAS Day7'], dist="norm", plot=plt)
plt.title('Q-Q Plot of Normal Data')

plt.tight_layout()
plt.show()
```

```
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future vers
ion. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



In [52]:
```
def check_normality(data, name):
    print(f"\n--- Normality Tests for '{name}' ---")

    # Shapiro-Wilk Test (good for n <= 5000)
    stat, p = stats.shapiro(data)
    print(f"Shapiro-Wilk Test: Statistic={stat:.4f}, p-value={p:.4f}")
    if p > 0.05:
```

```
        print("  -> Sample looks Gaussian (fail to reject H0)")
    else:
        print("  -> Sample does not look Gaussian (reject H0)")

    # D'Agostino's K-squared Test (omnibus test, good for larger samples)
    stat, p = stats.normaltest(data)
    print(f"D'Agostino's K^2 Test: Statistic={stat:.4f}, p-value={p:.4f}")
    if p > 0.05:
        print("  -> Sample looks Gaussian (fail to reject H0)")
    else:
        print("  -> Sample does not look Gaussian (reject H0)")

    # Anderson-Darling Test
    result = stats.anderson(data, dist='norm')
    print(f"Anderson-Darling Test: Statistic={result.statistic:.4f}")
    for i in range(len(result.critical_values)):
        sl, cv = result.significance_level[i], result.critical_values[i]
        if result.statistic < cv:
            print(f"  -> At {sl}% significance, data looks normal (critical
        else:
            print(f"  -> At {sl}% significance, data does not look normal (c

check_normality(df2['VAS day2 '], 'VAS day2 ')
```

```
--- Normality Tests for 'VAS day2 ' ---
Shapiro-Wilk Test: Statistic=0.8397, p-value=0.0000
  -> Sample does not look Gaussian (reject H0)
D'Agostino's K^2 Test: Statistic=23.4856, p-value=0.0000
  -> Sample does not look Gaussian (reject H0)
Anderson-Darling Test: Statistic=3.0198
  -> At 15.0% significance, data does not look normal (critical value = 0.54
60)
  -> At 10.0% significance, data does not look normal (critical value = 0.62
10)
  -> At 5.0% significance, data does not look normal (critical value = 0.746
0)
  -> At 2.5% significance, data does not look normal (critical value = 0.870
0)
  -> At 1.0% significance, data does not look normal (critical value = 1.034
0)
```

In [53]:
```
check_normality(df2['VAS Day7'], 'VAS Day7')
```

```
--- Normality Tests for 'VAS Day7' ---
Shapiro-Wilk Test: Statistic=0.7143, p-value=0.0000
  -> Sample does not look Gaussian (reject H0)
D'Agostino's K^2 Test: Statistic=48.2044, p-value=0.0000
  -> Sample does not look Gaussian (reject H0)
Anderson-Darling Test: Statistic=5.7352
  -> At 15.0% significance, data does not look normal (critical value = 0.54
60)
  -> At 10.0% significance, data does not look normal (critical value = 0.62
10)
  -> At 5.0% significance, data does not look normal (critical value = 0.746
0)
  -> At 2.5% significance, data does not look normal (critical value = 0.870
0)
  -> At 1.0% significance, data does not look normal (critical value = 1.034
0)
```

In [54]:
```
df2.columns
```

Out[54]:
```
Index(['Gender ', 'Age', 'Race', 'Tooth', 'Impaction', 'Duration', 'Group',
       'VAS (baseline)', 'VAS Day1', 'VAS day2 ', 'VAS Day7', 'MO (baselin
e)',
       'MO Day1', 'MO Day2', 'MO Day7', 'S1 (baseline)', 'S1 day1', 'S1 day
2',
       'S1 day7', 'S2 (baseline)', 'S2 day1', 'S2 day2', 'S2 day7',
       'S3 (baseline)', 'S3 day1', 'S3 day2', 'S3 day7', 'Arcoxia (days)'],
      dtype='object')
```

In [61]:
```
df2.Group.unique()
```

Out[61]:
```
array(['Placebo', 'Laser'], dtype=object)
```

In [66]:
```
# columns to summarize
columns = ['Gender ', 'Race', 'Tooth', 'Impaction','Duration', 'VAS (baselir
       'MO Day1', 'MO Day2', 'MO Day7', 'S1 (baseline)', 'S1 day1', 'S1 day2
       'S1 day7', 'S2 (baseline)', 'S2 day1', 'S2 day2', 'S2 day7',
       'S3 (baseline)', 'S3 day1', 'S3 day2', 'S3 day7', 'Arcoxia (days)','G

# columns containing categorical variables
categorical = ['Group','Gender ', 'Race', 'Tooth', 'Impaction']

# columns containing categorical variables
continuous = ['Duration', 'VAS (baseline)', 'VAS Day1', 'VAS day2 ', 'VAS Da
       'MO Day1', 'MO Day2', 'MO Day7', 'S1 (baseline)', 'S1 day1', 'S1 day2
       'S1 day7', 'S2 (baseline)', 'S2 day1', 'S2 day2', 'S2 day7',
       'S3 (baseline)', 'S3 day1', 'S3 day2', 'S3 day7', 'Arcoxia (days)']

# non-normal variables
nonnormal = ['VAS (baseline)','VAS day2 ', 'VAS Day7']

# limit the binary variable "death" to a single row
# limit = {"death": 1}

# set the order of the categorical variables
# order = {"Group": ["Laser", "Placebo"]}
```

```
# alternative labels
# labels={'death': 'Mortality'}

# set decimal places for age to 0
decimals = {"Age": 0}

# optionally, a categorical variable for stratification
groupby = 'Group'

# rename the death column
# labels={'death': 'Mortality'}

# display minimum and maximum for listed variables
# min_max = ['Height']

table2 = TableOne(df2, columns=columns, categorical=categorical, continuous=
                  groupby=groupby, nonnormal=nonnormal, #rename=labels, #lab
                  decimals=decimals, #limit=limit, min_max=min_max,
                  dip_test=True, normal_test=True, tukey_test=True)

table2
```

```
/usr/local/lib/python3.11/dist-packages/tableone/tableone.py:351: SettingWit
hCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data[self._categorical] = handle_categorical_nulls(data[self._categorica
l], self._categorical)
```

Out[66]:

			Grouped by Group		
		Missing	Overall	Laser	Placebo
n			65	35	30
Gender , n (%)	**Female**		45 (69.2)	24 (68.6)	21 (70.0)
	Male		20 (30.8)	11 (31.4)	9 (30.0)
Race, n (%)	**Chinese**		29 (44.6)	15 (42.9)	14 (46.7)
	India		1 (1.5)	1 (2.9)	0 (0.0)
	Malay		31 (47.7)	16 (45.7)	15 (50.0)
	Others		4 (6.2)	3 (8.6)	1 (3.3)
Tooth, n (%)	**20**		1 (1.5)	1 (2.9)	0 (0.0)
	38		36 (55.4)	19 (54.3)	17 (56.7)
	48		28 (43.1)	15 (42.9)	13 (43.3)
Impaction, n (%)	**1A**		3 (4.6)	1 (2.9)	2 (6.7)
	1B		1 (1.5)	1 (2.9)	0 (0.0)
	2A		56 (86.2)	30 (85.7)	26 (86.7)
	2B		5 (7.7)	3 (8.6)	2 (6.7)
Duration, mean (SD)		0	22.0 (6.9)	23.3 (7.3)	20.6 (6.2)
VAS (baseline), median [Q1,Q3]		0	13.0 [5.0,35.0]	25.0 [4.5,38.0]	13.0 [5.2,28.8]
VAS Day1, mean (SD)		0	22.5 (18.3)	22.0 (19.8)	23.1 (16.7)
VAS day2 , median [Q1,Q3]		0	12.0 [6.0,23.0]	8.0 [3.5,22.0]	15.0 [8.0,23.8]
VAS Day7, median [Q1,Q3]		0	2.0 [0.0,8.0]	1.0 [0.0,6.0]	3.5 [0.0,8.0]
MO (baseline), mean (SD)		0	45.8 (5.9)	45.4 (5.4)	46.2 (6.5)
MO Day1, mean (SD)		0	33.6 (9.3)	34.2 (8.5)	33.0 (10.2)
MO Day2, mean (SD)		0	36.2 (9.3)	36.7 (8.2)	35.7 (10.5)
MO Day7, mean (SD)		0	43.6 (7.0)	44.0 (5.2)	43.2 (8.7)
S1 (baseline), mean (SD)		0	111.3 (8.4)	112.2 (8.3)	110.2 (8.6)
S1 day1, mean (SD)		0	113.1 (7.9)	113.8 (7.9)	112.4 (7.9)
S1 day2, mean (SD)		0	113.0 (8.0)	113.2 (8.0)	112.7 (8.2)

				Grouped by Group	
		Missing	Overall	Laser	Placebo
S1 day7, mean (SD)		0	111.3 (8.4)	112.1 (8.3)	110.3 (8.6)
S2 (baseline), mean (SD)		0	116.9 (7.2)	115.8 (7.3)	118.2 (6.9)
S2 day1, mean (SD)		0	119.8 (7.7)	118.7 (7.7)	121.2 (7.5)
S2 day2, mean (SD)		0	119.8 (7.5)	118.5 (7.2)	121.4 (7.7)
S2 day7, mean (SD)		0	117.0 (7.2)	115.8 (7.3)	118.4 (7.0)
S3 (baseline), mean (SD)		0	147.1 (12.8)	145.4 (11.3)	149.1 (14.3)
S3 day1, mean (SD)		0	149.9 (13.0)	148.0 (11.6)	152.2 (14.3)
S3 day2, mean (SD)		0	150.0 (13.2)	147.7 (11.8)	152.6 (14.4)
S3 day7, mean (SD)		0	147.1 (12.8)	145.4 (11.3)	149.1 (14.3)
Arcoxia (days), mean (SD)		0	2.0 (1.7)	1.7 (1.8)	2.3 (1.6)
Group, n (%)	**Laser**		35 (53.8)	35 (100.0)	0 (0.0)
	Placebo		30 (46.2)	0 (0.0)	30 (100.0)

[1] Hartigan's Dip Test reports possible multimodal distributions for: Arcoxia (days).
[2] Normality test reports non-normal distributions for: VAS (baseline), VAS Day7, VAS day2 .
[3] Tukey test indicates far outliers in: MO Day7.

In [69]:
```
df3 = df2.copy()
```

In [71]:
```
# create grouped_table with p values
table3 = TableOne(df3, columns, categorical, continuous, groupby, nonnormal,
                  htest_name=True, dip_test=True, normal_test=True, tukey_te
table3
```

Out[71]:

Gr

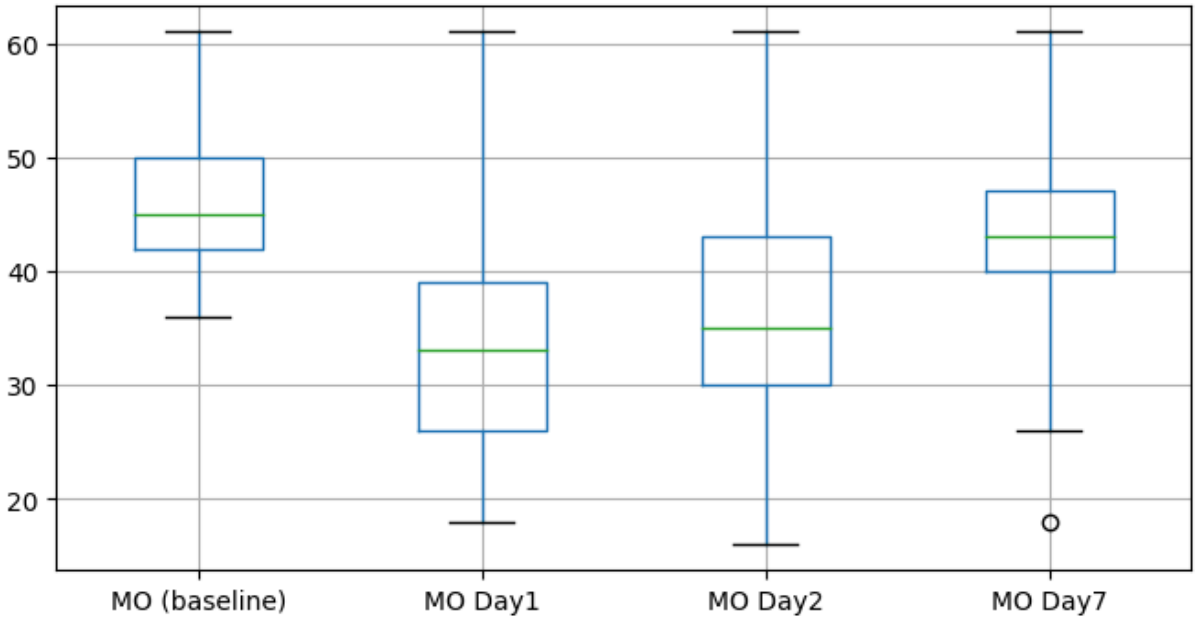
		Missing	Overall	Laser	Placebo	SMD (Laser,Placebo)
n			65	35	30	
Gender , n (%)	**Female**		45 (69.2)	24 (68.6)	21 (70.0)	0.031
	Male		20 (30.8)	11 (31.4)	9 (30.0)	
Race, n (%)	**Chinese**		29 (44.6)	15 (42.9)	14 (46.7)	0.337
	India		1 (1.5)	1 (2.9)	0 (0.0)	
	Malay		31 (47.7)	16 (45.7)	15 (50.0)	
	Others		4 (6.2)	3 (8.6)	1 (3.3)	
Tooth, n (%)	**20**		1 (1.5)	1 (2.9)	0 (0.0)	0.243
	38		36 (55.4)	19 (54.3)	17 (56.7)	
	48		28 (43.1)	15 (42.9)	13 (43.3)	
Impaction, n (%)	**1A**		3 (4.6)	1 (2.9)	2 (6.7)	0.308
	1B		1 (1.5)	1 (2.9)	0 (0.0)	
	2A		56 (86.2)	30 (85.7)	26 (86.7)	
	2B		5 (7.7)	3 (8.6)	2 (6.7)	
Duration, mean (SD)		0	22.0 (6.9)	23.3 (7.3)	20.6 (6.2)	-0.404
VAS (baseline), median [Q1,Q3]		0	13.0 [5.0,35.0]	25.0 [4.5,38.0]	13.0 [5.2,28.8]	-0.152
VAS Day1, mean (SD)		0	22.5 (18.3)	22.0 (19.8)	23.1 (16.7)	0.063
VAS day2 , median [Q1,Q3]		0	12.0 [6.0,23.0]	8.0 [3.5,22.0]	15.0 [8.0,23.8]	0.069

	Missing	Overall	Laser	Placebo	SMD (Laser,Placebo)
VAS Day7, median [Q1,Q3]	0	2.0 [0.0,8.0]	1.0 [0.0,6.0]	3.5 [0.0,8.0]	0.202
MO (baseline), mean (SD)	0	45.8 (5.9)	45.4 (5.4)	46.2 (6.5)	0.134
MO Day1, mean (SD)	0	33.6 (9.3)	34.2 (8.5)	33.0 (10.2)	-0.121
MO Day2, mean (SD)	0	36.2 (9.3)	36.7 (8.2)	35.7 (10.5)	-0.108
MO Day7, mean (SD)	0	43.6 (7.0)	44.0 (5.2)	43.2 (8.7)	-0.106
S1 (baseline), mean (SD)	0	111.3 (8.4)	112.2 (8.3)	110.2 (8.6)	-0.229
S1 day1, mean (SD)	0	113.1 (7.9)	113.8 (7.9)	112.4 (7.9)	-0.178
S1 day2, mean (SD)	0	113.0 (8.0)	113.2 (8.0)	112.7 (8.2)	-0.069
S1 day7, mean (SD)	0	111.3 (8.4)	112.1 (8.3)	110.3 (8.6)	-0.208
S2 (baseline), mean (SD)	0	116.9 (7.2)	115.8 (7.3)	118.2 (6.9)	0.333
S2 day1, mean (SD)	0	119.8 (7.7)	118.7 (7.7)	121.2 (7.5)	0.330
S2 day2, mean (SD)	0	119.8 (7.5)	118.5 (7.2)	121.4 (7.7)	0.391
S2 day7, mean (SD)	0	117.0 (7.2)	115.8 (7.3)	118.4 (7.0)	0.359
S3 (baseline), mean (SD)	0	147.1 (12.8)	145.4 (11.3)	149.1 (14.3)	0.289
S3 day1, mean (SD)	0	149.9 (13.0)	148.0 (11.6)	152.2 (14.3)	0.318
S3 day2, mean (SD)	0	150.0 (13.2)	147.7 (11.8)	152.6 (14.4)	0.366
S3 day7, mean (SD)	0	147.1 (12.8)	145.4 (11.3)	149.1 (14.3)	0.289
Arcoxia (days), mean (SD)	0	2.0 (1.7)	1.7 (1.8)	2.3 (1.6)	0.330

		Missing	Overall	Laser	Placebo	SMD (Laser,Placebo)
Group, n (%)	**Laser**		35 (53.8)	35 (100.0)	0 (0.0)	nan
	Placebo		30 (46.2)	0 (0.0)	30 (100.0)	

[1] Hartigan's Dip Test reports possible multimodal distributions for: Arcoxia (days).
[2] Normality test reports non-normal distributions for: VAS (baseline), VAS Day7, VAS day2 .
[3] Tukey test indicates far outliers in: MO Day7.

In [75]:
```
# Exploring the warning raised by Tukey's rule
df2[['MO (baseline)','MO Day1','MO Day2','MO Day7']].boxplot(whis=3, figsize
plt.show()
```



In [77]:
```
df2[['MO (baseline)','MO Day1','MO Day2','MO Day7']].describe().T
```

Out[77]:

	count	mean	std	min	25%	50%	75%	max
MO (baseline)	65.00	45.80	5.92	36.00	42.00	45.00	50.00	61.00
MO Day1	65.00	33.65	9.27	18.00	26.00	33.00	39.00	61.00
MO Day2	65.00	36.22	9.28	16.00	30.00	35.00	43.00	61.00
MO Day7	65.00	43.65	7.03	18.00	40.00	43.00	47.00	61.00

In [72]:
```
print(table3.tabulate(tablefmt = "github"))
```

| | | Missing | Overall | L
aser | Placebo | SMD (Laser,Placebo) | P-Value | Test
|
|---|---|---|---|---|---|---|---|---|
| n | | | 65 | 3
5 | 30 | | | |
|
| Gender , n (%) | Female | | 45 (69.2) | 2
4 (68.6) | 21 (70.0) | 0.031 | 1.000 | Chi-s
quared |
| | Male | | 20 (30.8) | 1
1 (31.4) | 9 (30.0) | | | |
|
| Race, n (%) | Chinese | | 29 (44.6) | 1
5 (42.9) | 14 (46.7) | 0.337 | 0.639 | Chi-s
quared (warning: expected count < 5) |
| | India | | 1 (1.5) | 1
(2.9) | 0 (0.0) | | | |
|
| | Malay | | 31 (47.7) | 1
6 (45.7) | 15 (50.0) | | | |
|
| | Others | | 4 (6.2) | 3
(8.6) | 1 (3.3) | | | |
|
| Tooth, n (%) | 20 | | 1 (1.5) | 1
(2.9) | 0 (0.0) | 0.243 | 0.646 | Chi-sq
uared (warning: expected count < 5) |
| | 38 | | 36 (55.4) | 1
9 (54.3) | 17 (56.7) | | | |
|
| | 48 | | 28 (43.1) | 1
5 (42.9) | 13 (43.3) | | | |
|
| Impaction, n (%) | 1A | | 3 (4.6) | 1
(2.9) | 2 (6.7) | 0.308 | 0.695 | Chi-sq
uared (warning: expected count < 5) |
| | 1B | | 1 (1.5) | 1
(2.9) | 0 (0.0) | | | |
|
| | 2A | | 56 (86.2) | 3
0 (85.7) | 26 (86.7) | | | |
|
| | 2B | | 5 (7.7) | 3
(8.6) | 2 (6.7) | | | |
|
| Duration, mean (SD) | | 0 | 22.0 (6.9) | 2
3.3 (7.3) | 20.6 (6.2) | -0.404 | 0.108 | Welc
h’s T-test |
| VAS (baseline), median [Q1,Q3] | | 0 | 13.0 [5.0,35.0] | 2
5.0 [4.5,38.0] | 13.0 [5.2,28.8] | -0.152 | 0.697 | Krusk
al-Wallis |
| VAS Day1, mean (SD) | | 0 | 22.5 (18.3) | 2
2.0 (19.8) | 23.1 (16.7) | 0.063 | 0.798 | Welc

h’s T-test |
VAS day2 , median [Q1,Q3]		0	12.0 [6.0,23.0]	8.0 [3.5,22.0]	15.0 [8.0,23.8]	0.069	0.370	Kruskal-Wallis
VAS Day7, median [Q1,Q3]		0	2.0 [0.0,8.0]	1.0 [0.0,6.0]	3.5 [0.0,8.0]	0.202	0.316	Kruskal-Wallis
MO (baseline), mean (SD)		0	45.8 (5.9)	45.4 (5.4)	46.2 (6.5)	0.134	0.594	Welch’s T-test
MO Day1, mean (SD)		0	33.6 (9.3)	34.2 (8.5)	33.0 (10.2)	-0.121	0.631	Welch’s T-test
MO Day2, mean (SD)		0	36.2 (9.3)	36.7 (8.2)	35.7 (10.5)	-0.108	0.669	Welch’s T-test
MO Day7, mean (SD)		0	43.6 (7.0)	44.0 (5.2)	43.2 (8.7)	-0.106	0.677	Welch’s T-test
S1 (baseline), mean (SD)		0	111.3 (8.4)	112.2 (8.3)	110.2 (8.6)	-0.229	0.362	Welch’s T-test
S1 day1, mean (SD)		0	113.1 (7.9)	113.8 (7.9)	112.4 (7.9)	-0.178	0.477	Welch’s T-test
S1 day2, mean (SD)		0	113.0 (8.0)	113.2 (8.0)	112.7 (8.2)	-0.069	0.782	Welch’s T-test
S1 day7, mean (SD)		0	111.3 (8.4)	112.1 (8.3)	110.3 (8.6)	-0.208	0.408	Welch’s T-test
S2 (baseline), mean (SD)		0	116.9 (7.2)	115.8 (7.3)	118.2 (6.9)	0.333	0.185	Welch’s T-test
S2 day1, mean (SD)		0	119.8 (7.7)	118.7 (7.7)	121.2 (7.5)	0.330	0.189	Welch’s T-test
S2 day2, mean (SD)		0	119.8 (7.5)	118.5 (7.2)	121.4 (7.7)	0.391	0.122	Welch’s T-test
S2 day7, mean (SD)		0	117.0 (7.2)	115.8 (7.3)	118.4 (7.0)	0.359	0.154	Welch’s T-test
S3 (baseline), mean (SD)		0	147.1 (12.8)	145.4 (11.3)	149.1 (14.3)	0.289	0.254	Welch’s T-test
S3 day1, mean (SD)		0	149.9 (13.0)	148.0 (11.6)	152.2 (14.3)	0.318	0.211	Welch’s T-test
S3 day2, mean (SD)		0	150.0 (13.2)	147.7 (11.8)	152.6 (14.4)	0.366	0.150	Welch’s T-test
S3 day7, mean (SD)		0	147.1 (12.8)	145.4 (11.3)	149.1 (14.3)	0.289	0.254	Welch’s T-test
Arcoxia (days), mean (SD)		0	2.0 (1.7)	

```
1.7 (1.8)       | 2.3 (1.6)       | 0.330                | 0.189     | Welc
h’s T-test                          |
| Group, n (%)                    | Laser   |           | 35 (53.8)       | 3
5 (100.0)      | 0 (0.0)         | nan                  | <0.001    | Chi-s
quared                              |
|                                 | Placebo |           | 30 (46.2)       | 0
(0.0)          | 30 (100.0)      |                      |           |
|
```

In [73]:
```
# Save to Excel
fn1 = 'tableone.xlsx'
table3.to_excel(fn1)
```

In [4]:
```
import pandas as pd
import numpy as np
from scipy import stats
from statsmodels.stats.anova import AnovaRM
from statsmodels.stats.multicomp import MultiComparison
import matplotlib.pyplot as plt
import seaborn as sns

# Load Excel file
# df = pd.read_excel("lllt_data_template.xlsx")
df['Group'] = df['Group'].str.lower()

def to_long_format(df, id_var, group_col, measure_vars, var_name, value_name
    long_df = pd.melt(df, id_vars=[id_var, group_col], value_vars=measure_va
                      var_name=var_name, value_name=value_name)
    long_df[var_name] = long_df[var_name].str.extract(day_regex, expand=Fals
    long_df[var_name] = long_df[var_name].astype(int)
    return long_df

def ttest_by_day(df, group_col, measure_cols, label):
    for col in measure_cols:
        lllt = df[df[group_col] == 'laser'][col]
        placebo = df[df[group_col] == 'placebo'][col]
        stat, p = stats.ttest_ind(lllt, placebo, equal_var=False)
        print(f"{label} | {col} | T-test p = {p:.4f}")

def run_rmanova(df_long, subject_col, within_col, depvar, between_col):
    return AnovaRM(df_long, depvar=depvar, subject=subject_col,
                   within=[within_col], between=between_col).fit()

def plot_line(data, x_col, y_col, title, y_label):
    plt.figure(figsize=(8,5))
    sns.lineplot(data=data, x=x_col, y=y_col, hue='Group', marker='o')
    plt.title(title)
    plt.ylabel(y_label)
    plt.xlabel('Day')
    plt.tight_layout()
    plt.show()
```

In [5]:
```
df.columns
```

Out[5]:
```
Index(['Subject', 'Name', 'Gender ', 'Age', 'Race', 'Tooth', 'Impaction',
       'Duration', 'Group', 'VAS (baseline)', 'VAS Day1', 'VAS day2 ',
       'VAS Day7', 'MO (baseline)', 'MO Day1', 'MO Day2', 'MO Day7',
       'S1 (baseline)', 'S1 day1', 'S1 day2', 'S1 day7', 'S2 (baseline)',
       'S2 day1', 'S2 day2', 'S2 day7', 'S3 (baseline)', 'S3 day1', 'S3 day
2',
       'S3 day7', 'Arcoxia (days)'],
      dtype='object')
```

In [8]:
```
import statsmodels.formula.api as smf

# Convert 'Day' to categorical for interaction
pain_long['Day'] = pain_long['Day'].astype(str)

# Run mixed effects model
model = smf.mixedlm("VAS ~ Day * Group", pain_long, groups=pain_long["Subjec
result = model.fit()
print("\nMixed Effects Model: Pain Intensity")
print(result.summary())
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
/tmp/ipykernel_35/2629519611.py in <cell line: 0>()
      2
      3 # Convert 'Day' to categorical for interaction
----> 4 pain_long['Day'] = pain_long['Day'].astype(str)
      5
      6 # Run mixed effects model

NameError: name 'pain_long' is not defined
```

In [7]:
```
def run_rmanova_per_group(df_long, group_name, subject_col, within_col, depv
    group_df = df_long[df_long['Group'] == group_name]
    return AnovaRM(group_df, depvar=depvar, subject=subject_col,
                   within=[within_col]).fit()

# Pain RM-ANOVA by group
for group in ['laser', 'placebo']:
    result = run_rmanova_per_group(pain_long, group, 'Subject', 'Day', 'VAS'
    print(f"\nRepeated Measures ANOVA (Pain) - Group: {group.capitalize()}")
    print(result)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
/tmp/ipykernel_35/444261352.py in <cell line: 0>()
      6 # Pain RM-ANOVA by group
      7 for group in ['laser', 'placebo']:
----> 8     result = run_rmanova_per_group(df_long, group, 'Subject', 'Day',
'VAS')
      9     print(f"\nRepeated Measures ANOVA (Pain) - Group: {group.capital
ize()}")
     10     print(result)

NameError: name 'df_long' is not defined
```

In [ ]:
```
# Only if RM-ANOVA is significant
mc = MultiComparison(pain_long['VAS'], pain_long['Group'])
posthoc = mc.allpairtest(stats.ttest_ind, method='bonf')
print("\nPost-Hoc Pairwise Comparison (Pain):")
print(posthoc[0])
```

In [ ]:
```
# Analgesic use: Arcoxia
arc_laser = df[df['Group'] == 'laser']['Arcoxia (days)']
arc_placebo = df[df['Group'] == 'placebo']['Arcoxia (days)']
stat, p = stats.ttest_ind(arc_laser, arc_placebo, equal_var=False)
print(f"\nAnalgesic Consumption T-test: p = {p:.4f}")
```

In [ ]:
```
import pandas as pd
import numpy as np
from scipy.stats import ttest_ind
from statsmodels.stats.anova import AnovaRM
import statsmodels.formula.api as smf
import warnings

# --------------------------
# 1. Helper: Wide → Long
# --------------------------
def to_long_format(df, subject_col, group_col, value_cols, time_name, value_
    day_numbers = [col.replace(value_name + '_', '') for col in value_cols]
    df_long = df[[subject_col, group_col] + value_cols].melt(
        id_vars=[subject_col, group_col],
        value_vars=value_cols,
        var_name=time_name,
        value_name=value_name
    )
    df_long[time_name] = df_long[time_name].str.extract(r'(\d+)').astype(int
    return df_long.dropna()

# --------------------------
# 2. Helper: Run RM-ANOVA Separately by Group
# --------------------------
def run_rmanova_per_group(df_long, group_name, subject_col, within_col, depv
    group_df = df_long[df_long['Group'] == group_name].copy()

    # Aggregate to ensure one value per subject/day
    group_df = group_df.groupby([subject_col, within_col], as_index=False)[d

    if group_df[subject_col].nunique() < 2 or group_df[within_col].nunique()
        return "⚠️ Not enough data for RM-ANOVA"

    try:
        return AnovaRM(group_df, depvar=depvar, subject=subject_col, within=
    except Exception as e:
        return f"❌ RM-ANOVA failed: {e}"

# --------------------------
# 3. Helper: Run Mixed Effects Model
# --------------------------
def run_mixed_effects(df_long, depvar):
    df_long = df_long.dropna(subset=[depvar, 'Day', 'Group', 'Subject'])
```

```
    df_long['Day'] = df_long['Day'].astype(str)

    try:
        model = smf.mixedlm(f"{depvar} ~ Day * Group", df_long, groups=df_lo
        return model.fit()
    except Exception as e:
        return f"❌ Mixed Effects Model failed: {e}"

# --------------------------
# 4. Helper: Safe t-tests
# --------------------------
def safe_ttest(group1, group2):
    if len(group1) < 2 or len(group2) < 2:
        return np.nan, np.nan
    with warnings.catch_warnings():
        warnings.simplefilter("ignore")
        return ttest_ind(group1, group2, nan_policy='omit')

# --------------------------
# Section Template
# --------------------------
def analyze_measure(df, cols, value_name, label):
    print(f"\n\n=== {label.upper()} ===")
    long_df = to_long_format(df, 'Subject', 'Group', cols, 'Day', value_name

    # --- T-tests ---
    print("\nIndependent t-tests:")
    for day in [1, 2, 7]:
        sub = long_df[long_df['Day'] == day]
        t, p = safe_ttest(sub[sub.Group == 'laser'][value_name], sub[sub.Gro
        print(f"Day {day}: t={t:.3f}, p={p:.3f}")

    # --- RM-ANOVA by Group ---
    for grp in ['laser', 'placebo']:
        result = run_rmanova_per_group(long_df, grp, 'Subject', 'Day', value
        print(f"\nRepeated Measures ANOVA - {label} (Group: {grp})")
        print(result if isinstance(result, str) else result.summary())

    # --- Mixed Model ---
    mixed = run_mixed_effects(long_df, value_name)
    print(f"\nMixed Effects Model - {label}")
    print(mixed if isinstance(mixed, str) else mixed.summary())


# --------------------------
# Load your DataFrame here
# --------------------------
# df = pd.read_csv("your_clean_data.csv")  # Uncomment and replace as needed

# --------------------------
# ANALYSIS SECTIONS
# --------------------------
analyze_measure(df, ['VAS Day1', 'VAS day2 ', 'VAS Day7'], 'VAS', 'Pain Inte
analyze_measure(df, ['S1 day1', 'S1 day2', 'S1 day7'], 'S1', 'Swelling S1')
analyze_measure(df, ['MO Day1', 'MO Day2', 'MO Day7'], 'Mouth', 'Trismus')
```

In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Visualization
def plot_line(data, y_col, title):
    # Clean data: convert inf to NaN and drop missing
    plot_df = data.copy()
    plot_df = plot_df.replace([np.inf, -np.inf], np.nan).dropna(subset=['Day

    # Ensure 'Day' is categorical and ordered
    day_order = sorted(plot_df['Day'].astype(int).unique())
    cat_type = pd.api.types.CategoricalDtype(categories=[str(d) for d in day
    plot_df['Day'] = plot_df['Day'].astype(int).astype(str).astype(cat_type)

    # Plot
    plt.figure(figsize=(8, 5))
    sns.lineplot(
        data=plot_df,
        x='Day',
        y=y_col,
        hue='Group',
        marker='o',
        errorbar='sd'
    )
    plt.title(title)
    plt.ylabel(y_col)
    plt.xlabel('Day')
    plt.tight_layout()
    plt.show()

import warnings

with warnings.catch_warnings():
    warnings.simplefilter("ignore", FutureWarning)
    plot_line(pain_long, 'VAS', 'Pain Intensity Over Time')
    plot_line(swell_long, 'S1', 'Swelling (S1) Over Time')
    plot_line(mo_long, 'MO', 'Mouth Opening Over Time')
```

In [ ]:
```
!pip install ydata-profiling
```

In [ ]:
```
import pandas as pd
from ydata_profiling import ProfileReport

# Generate profile report
profile = ProfileReport(df, title="Initial Report Data - LLLT", explorative=

# Save it directly to the root (Kaggle File Explorer reads root-level files
output_path = "/kaggle/working/pLLLT_report.html"
profile.to_file(output_path)

print(f"Profile report saved to: {output_path}")
```

In [ ]:

This notebook was converted with convert.ploomber.io