In [ ]:
```
import pandas as pd
from pybaseball import statcast_pitcher_pitch_arsenal
```

```
/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/urllib3/__i
nit__.py:35: NotOpenSSLWarning: urllib3 v2 only supports OpenSSL 1.1.1+, cur
rently the 'ssl' module is compiled with 'LibreSSL 2.8.3'. See: https://gith
ub.com/urllib3/urllib3/issues/3020
  warnings.warn(
```

In [28]:
```
from pybaseball import statcast
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime, timedelta

def analyze_rest_impact(start_date, end_date, min_pitches=20):
    """
    Analyze how pitchers' arm angles and xwOBA change based on rest days.
    """
    try:
        # Get data
        data = get_statcast_data(start_date, end_date)

        # Clean and prepare data
        analysis_data = data[['pitcher', 'player_name', 'arm_angle',
                              'pitcher_days_since_prev_game', 'estimated_woba_
                              'home_team', 'away_team', 'p_throws']].dropna()

        # Categorize rest days
        analysis_data['rest_category'] = np.where(analysis_data['pitcher_day
                                                  'low_rest', 'normal_rest')

        # Calculate pitcher-level statistics
        pitcher_stats = []

        for pitcher in analysis_data['pitcher'].unique():
            pitcher_data = analysis_data[analysis_data['pitcher'] == pitcher

            # Get low rest and normal rest data
            low_rest = pitcher_data[pitcher_data['rest_category'] == 'low_re
            normal_rest = pitcher_data[pitcher_data['rest_category'] == 'nor

            # Only include pitchers with sufficient samples in both categori
            if len(low_rest) >= min_pitches and len(normal_rest) >= min_pitc
                # Calculate changes in arm angle and xwOBA
                arm_angle_change = low_rest['arm_angle'].mean() - normal_res
                xwoba_change = low_rest['estimated_woba_using_speedangle'].m

                pitcher_stats.append({
                    'pitcher': pitcher,
                    'player_name': pitcher_data['player_name'].iloc[0],
                    'team': pitcher_data['home_team'].iloc[0],  # Using home
                    'throws': pitcher_data['p_throws'].iloc[0],
```

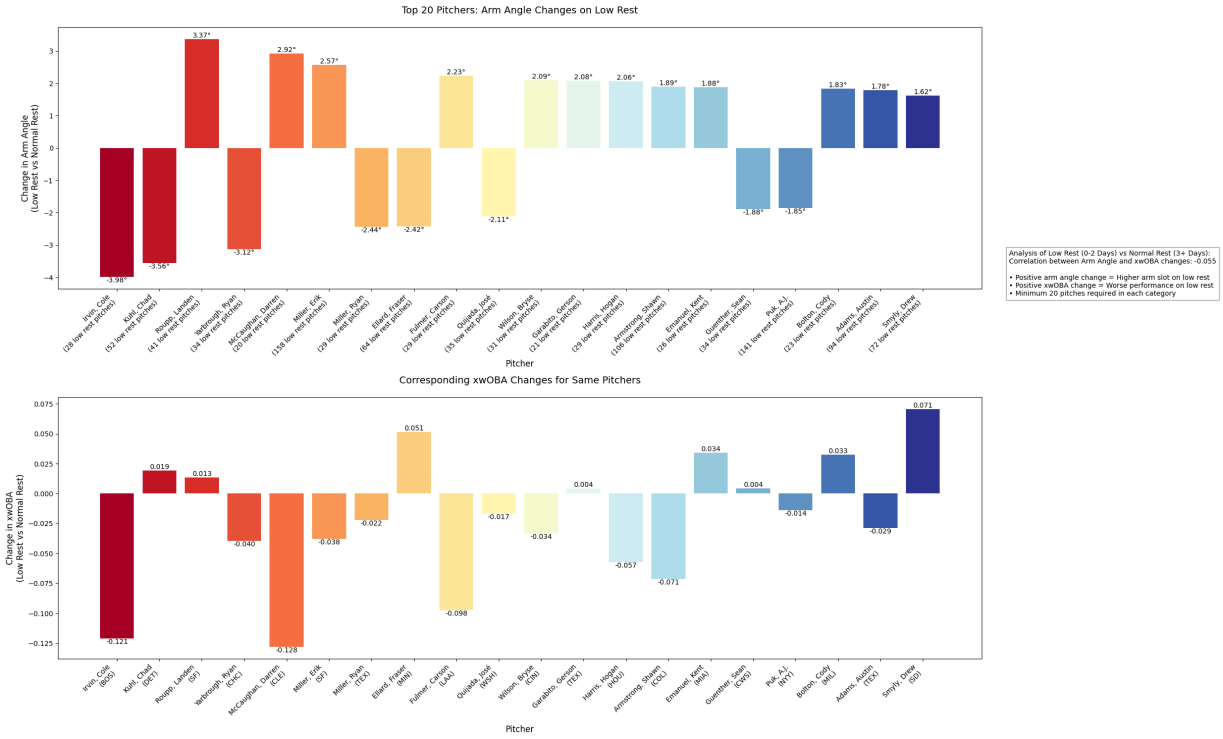
```
                'total_pitches': len(pitcher_data),
                'low_rest_pitches': len(low_rest),
                'normal_rest_pitches': len(normal_rest),
                'arm_angle_change': arm_angle_change,
                'xwoba_change': xwoba_change,
                'low_rest_arm_angle': low_rest['arm_angle'].mean(),
                'normal_rest_arm_angle': normal_rest['arm_angle'].mean()
                'low_rest_xwoba': low_rest['estimated_woba_using_speedar
                'normal_rest_xwoba': normal_rest['estimated_woba_using_s
            })

    # Create DataFrame
    analysis_df = pd.DataFrame(pitcher_stats)

    # Sort by absolute arm angle change
    analysis_df['abs_arm_angle_change'] = abs(analysis_df['arm_angle_cha
    top_changers = analysis_df.nlargest(20, 'abs_arm_angle_change')

    # Create visualization
    fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(20, 15))

    # Plot 1: Arm Angle Changes
    bars1 = ax1.bar(range(len(top_changers)),
                    top_changers['arm_angle_change'],
                    color=[plt.cm.RdYlBu(x) for x in np.linspace(0, 1, le

    ax1.set_title('Top 20 Pitchers: Arm Angle Changes on Low Rest', pad=
    ax1.set_xlabel('Pitcher', fontsize=12)
    ax1.set_ylabel('Change in Arm Angle\n(Low Rest vs Normal Rest)', for

    # Add pitcher labels
    ax1.set_xticks(range(len(top_changers)))
    ax1.set_xticklabels([f"{name}\n({pitches} low rest pitches)"
                         for name, pitches in zip(top_changers['player_na
                                                  top_changers['low_rest_pi
                        rotation=45, ha='right', fontsize=10)

    # Add value labels
    for i, bar in enumerate(bars1):
        height = bar.get_height()
        ax1.text(bar.get_x() + bar.get_width()/2., height,
                 f'{height:.2f}°',
                 ha='center', va='bottom' if height > 0 else 'top',
                 fontsize=10)

    # Plot 2: Corresponding xwOBA Changes
    bars2 = ax2.bar(range(len(top_changers)),
                    top_changers['xwoba_change'],
                    color=[plt.cm.RdYlBu(x) for x in np.linspace(0, 1, le

    ax2.set_title('Corresponding xwOBA Changes for Same Pitchers', pad=2
    ax2.set_xlabel('Pitcher', fontsize=12)
    ax2.set_ylabel('Change in xwOBA\n(Low Rest vs Normal Rest)', fontsiz

    # Add pitcher labels
    ax2.set_xticks(range(len(top_changers)))
```

```
        ax2.set_xticklabels([f"{name}\n({team})"
                             for name, team in zip(top_changers['player_name'
                                                   top_changers['team'])],
                            rotation=45, ha='right', fontsize=10)

        # Add value labels
        for i, bar in enumerate(bars2):
            height = bar.get_height()
            ax2.text(bar.get_x() + bar.get_width()/2., height,
                     f'{height:.3f}',
                     ha='center', va='bottom' if height > 0 else 'top',
                     fontsize=10)

        # Add correlation information
        correlation = np.corrcoef(top_changers['arm_angle_change'],
                                  top_changers['xwoba_change'])[0,1]

        explanation_text = (
            f"Analysis of Low Rest (0-2 Days) vs Normal Rest (3+ Days):\n"
            f"Correlation between Arm Angle and xwOBA changes: {correlation:
            f"• Positive arm angle change = Higher arm slot on low rest\n"
            f"• Positive xwOBA change = Worse performance on low rest\n"
            f"• Minimum {min_pitches} pitches required in each category"
        )
        plt.figtext(1.02, 0.6, explanation_text,
                    bbox=dict(facecolor='white', alpha=0.8, edgecolor='gray')
                    fontsize=10)

        plt.tight_layout()
        plt.show()

        # Print detailed statistics
        print("\nDetailed Statistics for Top Arm Angle Changers:")
        print(top_changers[['player_name', 'team', 'arm_angle_change', 'xwob
                            'low_rest_pitches', 'normal_rest_pitches']].to_st

        return analysis_df

    except Exception as e:
        print(f"Analysis failed: {str(e)}")
        raise

# Run analysis
if __name__ == "__main__":
    try:
        results = analyze_rest_impact("2024-03-28", "2024-11-01")
    except Exception as e:
        print(f"Analysis failed: {str(e)}")
```

```
Retrieving Statcast data in chunks...
Fetching data from 2024-03-28 to 2024-04-11...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 11.66it/s]
Retrieved 55595 rows
Fetching data from 2024-04-12 to 2024-04-26...
This is a large query, it may take a moment to complete
```

```
 40%|████      | 6/15 [00:01<00:01,  5.31it/s]
Error retrieving chunk: 'game_date'
Fetching data from 2024-04-27 to 2024-05-11...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 12.17it/s]
Retrieved 57843 rows
Fetching data from 2024-05-12 to 2024-05-26...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 13.03it/s]
Retrieved 58673 rows
Fetching data from 2024-05-27 to 2024-06-10...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 11.96it/s]
Retrieved 55561 rows
Fetching data from 2024-06-11 to 2024-06-25...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 12.86it/s]
Retrieved 58498 rows
Fetching data from 2024-06-26 to 2024-07-10...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 12.10it/s]
Retrieved 58538 rows
Fetching data from 2024-07-11 to 2024-07-25...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00,  7.88it/s]
/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pybaseball/
statcast.py:85: FutureWarning: The behavior of DataFrame concatenation with
empty or all-NA entries is deprecated. In a future version, this will no lon
ger exclude empty or all-NA columns when determining the result dtypes. To r
etain the old behavior, exclude the relevant entries before the concat opera
tion.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_stri
ng=False)
Retrieved 44302 rows
Fetching data from 2024-07-26 to 2024-08-09...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 12.83it/s]
Retrieved 59606 rows
Fetching data from 2024-08-10 to 2024-08-24...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 13.25it/s]
Retrieved 58832 rows
Fetching data from 2024-08-25 to 2024-09-08...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 14.32it/s]
Retrieved 60281 rows
Fetching data from 2024-09-09 to 2024-09-23...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 10.39it/s]
Retrieved 56459 rows
Fetching data from 2024-09-24 to 2024-10-08...
This is a large query, it may take a moment to complete
```

```
100%|██████████| 15/15 [00:01<00:00,  8.51it/s]
/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pybaseball/
statcast.py:85: FutureWarning: The behavior of DataFrame concatenation with
empty or all-NA entries is deprecated. In a future version, this will no lon
ger exclude empty or all-NA columns when determining the result dtypes. To r
etain the old behavior, exclude the relevant entries before the concat opera
tion.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_stri
ng=False)
Retrieved 30004 rows
Fetching data from 2024-10-09 to 2024-10-23...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00,  7.58it/s]
Retrieved 5795 rows
Fetching data from 2024-10-24 to 2024-11-01...
This is a large query, it may take a moment to complete
/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pybaseball/
statcast.py:85: FutureWarning: The behavior of DataFrame concatenation with
empty or all-NA entries is deprecated. In a future version, this will no lon
ger exclude empty or all-NA columns when determining the result dtypes. To r
etain the old behavior, exclude the relevant entries before the concat opera
tion.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_stri
ng=False)
100%|██████████| 9/9 [00:04<00:00,  2.08it/s]
/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pybaseball/
statcast.py:85: FutureWarning: The behavior of DataFrame concatenation with
empty or all-NA entries is deprecated. In a future version, this will no lon
ger exclude empty or all-NA columns when determining the result dtypes. To r
etain the old behavior, exclude the relevant entries before the concat opera
tion.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_stri
ng=False)
Retrieved 1576 rows
```


Top 20 Pitchers: Arm Angle Changes on Low Rest
Change in Arm Angle (Low Rest vs Normal Rest)
Pitcher
Analysis of Low Rest (0-2 Days) vs Normal Rest (3+ Days):
Correlation between Arm Angle and xwOBA changes: -0.055
• Positive arm angle change = Higher arm slot on low rest
• Positive xwOBA change = Worse performance on low rest
• Minimum 20 pitches required in each category


Corresponding xwOBA Changes for Same Pitchers
Change in xwOBA (Low Rest vs Normal Rest)
Pitcher

```
Detailed Statistics for Top Arm Angle Changers:
          player_name team  arm_angle_change  xwoba_change  low_rest_pitche
s  normal_rest_pitches
25         Irvin, Cole  BOS         -3.977399     -0.120869               2
8                  393
283         Kuhl, Chad  DET         -3.558097      0.019398               5
2                  171
69       Roupp, Landen   SF          3.366434      0.013346               4
1                  147
170    Yarbrough, Ryan  CHC         -3.119136     -0.039508               3
4                  316
282  McCaughan, Darren  CLE          2.918562     -0.127917               2
0                  146
67        Miller, Erik   SF          2.566947     -0.037792              15
8                  103
312       Miller, Ryan  TEX         -2.442633     -0.022029               2
9                   22
302     Ellard, Fraser  MIN         -2.421825      0.051300               6
4                   31
208     Fulmer, Carson  LAA          2.225693     -0.097534               2
9                  306
299      Quijada, José  WSH         -2.109524     -0.016872               3
5                   45
28       Wilson, Bryse  CIN          2.092246     -0.033504               3
1                  369
273   Garabito, Gerson  TEX          2.079887      0.004234               2
1                   76
267      Harris, Hogan  HOU          2.057995     -0.057397               2
9                  248
174   Armstrong, Shawn  COL          1.888870     -0.071314              10
6                  158
255      Emanuel, Kent  MIA          1.884240      0.034074               2
6                   41
306     Guenther, Sean  CWS         -1.883721      0.004144               3
4                   43
110          Puk, A.J.  NYY         -1.854701     -0.013920              14
1                  117
196       Bolton, Cody  MIL          1.831196      0.032571               2
3                   40
21       Adams, Austin  TEX          1.782871     -0.028626               9
4                   69
120        Smyly, Drew   SD          1.623186      0.070559               7
2                  147
```

In [30]:
```python
from pybaseball import statcast
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime, timedelta

def get_statcast_data(start_date, end_date, max_days=14):
    """
    Safely retrieve Statcast data in larger chunks for full season analysis.
    """
    start_dt = datetime.strptime(start_date, '%Y-%m-%d')
```

```
    end_dt = datetime.strptime(end_date, '%Y-%m-%d')

    all_data = []
    current_date = start_dt

    print("Retrieving Statcast data in chunks...")
    while current_date <= end_dt:
        chunk_end = min(current_date + timedelta(days=max_days), end_dt)

        try:
            print(f"Fetching data from {current_date.strftime('%Y-%m-%d')} t
            chunk_data = statcast(
                start_dt=current_date.strftime('%Y-%m-%d'),
                end_dt=chunk_end.strftime('%Y-%m-%d')
            )

            if chunk_data is not None and not chunk_data.empty:
                all_data.append(chunk_data)
                print(f"Retrieved {len(chunk_data)} rows")
            else:
                print("No data found for this date range")

        except Exception as e:
            print(f"Error retrieving data for {current_date.strftime('%Y-%m-

        current_date = chunk_end + timedelta(days=1)

    if not all_data:
        raise ValueError("No data could be retrieved from Statcast")

    return pd.concat(all_data, ignore_index=True)

def analyze_rest_vs_variance(start_date, end_date):
    """
    Analyze and visualize the relationship between days of rest and arm angl
    for the full season, including all available data points.
    """
    try:
        # Get and prepare data
        data = get_statcast_data(start_date, end_date)

        print(f"\nInitial data statistics:")
        print(f"Total pitches: {len(data)}")
        print(f"Unique pitchers: {data['pitcher'].nunique()}")

        # Select required columns and remove NaN values
        analysis_data = data[['pitcher', 'arm_angle', 'pitcher_days_since_pr
                              'player_name', 'home_team', 'away_team']].dropna

        # Calculate variance for each pitcher and rest day combination
        variance_by_rest = analysis_data.groupby(['pitcher', 'pitcher_days_s
            'arm_angle': ['var', 'count'],
            'player_name': 'first'
        }).reset_index()

        # Flatten column names
```

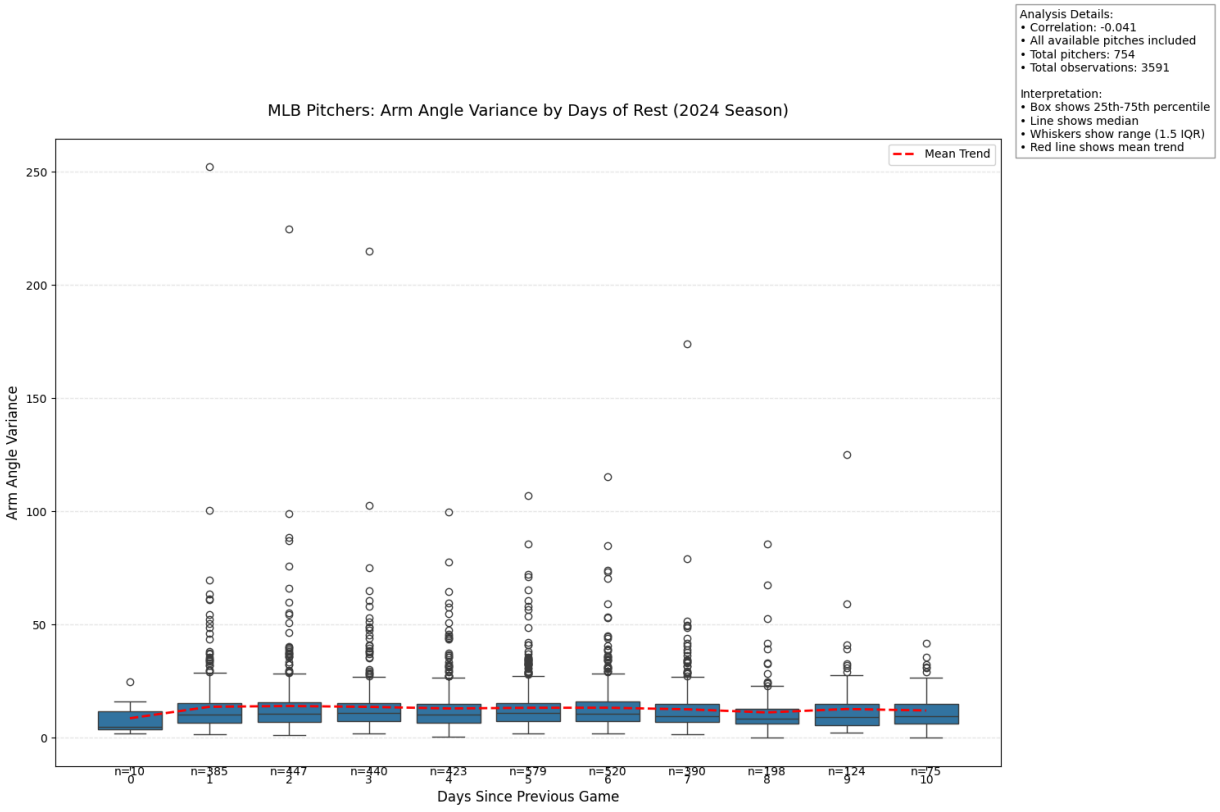
```
        variance_by_rest.columns = ['pitcher', 'days_rest', 'arm_angle_varia

        # Filter only for reasonable rest days and valid variances
        variance_by_rest = variance_by_rest[
            (variance_by_rest['days_rest'] <= 10) &     # Limit to reasonabl
            (variance_by_rest['days_rest'] >= 0) &      # Remove negative re
            (variance_by_rest['arm_angle_variance'].notna())    # Remove NaN
        ]

        # Create visualization
        plt.figure(figsize=(15, 10))

        # Create box plot
        ax = sns.boxplot(data=variance_by_rest, x='days_rest', y='arm_angle_
                         whis=1.5)  # 1.5 IQR for whiskers

        # Customize plot
        plt.title('MLB Pitchers: Arm Angle Variance by Days of Rest (2024 Se
                  pad=20, fontsize=14)
        plt.xlabel('Days Since Previous Game', fontsize=12)
        plt.ylabel('Arm Angle Variance', fontsize=12)

        # Add trend line
        means = variance_by_rest.groupby('days_rest')['arm_angle_variance'].
        plt.plot(range(len(means)), means.values, 'r--', linewidth=2, label=

        # Add sample sizes
        counts = variance_by_rest.groupby('days_rest')['pitcher'].count()
        for i, count in enumerate(counts):
            plt.text(i, ax.get_ylim()[0], f'n={count}', ha='center', va='top

        # Calculate correlations
        correlation = np.corrcoef(variance_by_rest['days_rest'],
                                  variance_by_rest['arm_angle_variance'])[0,1]

        # Add explanation text
        explanation_text = (
            f"Analysis Details:\n"
            f"• Correlation: {correlation:.3f}\n"
            f"• All available pitches included\n"
            f"• Total pitchers: {variance_by_rest['pitcher'].nunique()}\n"
            f"• Total observations: {len(variance_by_rest)}\n\n"
            f"Interpretation:\n"
            f"• Box shows 25th-75th percentile\n"
            f"• Line shows median\n"
            f"• Whiskers show range (1.5 IQR)\n"
            f"• Red line shows mean trend"
        )
        plt.text(1.02, 0.98, explanation_text,
                 transform=plt.gca().transAxes,
                 bbox=dict(facecolor='white', alpha=0.8, edgecolor='gray'),
                 fontsize=10)

        plt.grid(True, axis='y', linestyle='--', alpha=0.3)
        plt.legend()
        plt.tight_layout()
```

```
        plt.show()

        # Print summary statistics
        print("\nSummary Statistics by Days of Rest:")
        summary_stats = variance_by_rest.groupby('days_rest').agg({
            'arm_angle_variance': ['mean', 'median', 'std', 'count'],
            'pitcher': 'nunique'
        }).round(3)
        print(summary_stats)

        # Print additional statistics about sample sizes
        print("\nPitch Count Distribution:")
        pitch_count_stats = variance_by_rest.groupby('days_rest')['pitch_cou
        print(pitch_count_stats)

        return variance_by_rest

    except Exception as e:
        print(f"Analysis failed: {str(e)}")
        raise

# Run analysis for full season
if __name__ == "__main__":
    try:
        results = analyze_rest_vs_variance("2024-03-28", "2024-11-01")
    except Exception as e:
        print(f"Analysis failed: {str(e)}")
```

```
Retrieving Statcast data in chunks...
Fetching data from 2024-03-28 to 2024-04-11...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 11.13it/s]
Retrieved 55595 rows
Fetching data from 2024-04-12 to 2024-04-26...
This is a large query, it may take a moment to complete
  7%|▋         | 1/15 [00:01<00:17,  1.23s/it]
Error retrieving data for 2024-04-12: 'game_date'
Fetching data from 2024-04-27 to 2024-05-11...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 13.20it/s]
Retrieved 57843 rows
Fetching data from 2024-05-12 to 2024-05-26...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 12.07it/s]
Retrieved 58673 rows
Fetching data from 2024-05-27 to 2024-06-10...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 11.95it/s]
Retrieved 55561 rows
Fetching data from 2024-06-11 to 2024-06-25...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 12.22it/s]
Retrieved 58498 rows
Fetching data from 2024-06-26 to 2024-07-10...
This is a large query, it may take a moment to complete
```

```
100%|██████████| 15/15 [00:01<00:00, 12.48it/s]
Retrieved 58538 rows
Fetching data from 2024-07-11 to 2024-07-25...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 10.74it/s]
/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pybaseball/
statcast.py:85: FutureWarning: The behavior of DataFrame concatenation with
empty or all-NA entries is deprecated. In a future version, this will no lon
ger exclude empty or all-NA columns when determining the result dtypes. To r
etain the old behavior, exclude the relevant entries before the concat opera
tion.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_stri
ng=False)
Retrieved 44302 rows
Fetching data from 2024-07-26 to 2024-08-09...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 12.26it/s]
Retrieved 59606 rows
Fetching data from 2024-08-10 to 2024-08-24...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 11.88it/s]
Retrieved 58832 rows
Fetching data from 2024-08-25 to 2024-09-08...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 11.98it/s]
Retrieved 60281 rows
Fetching data from 2024-09-09 to 2024-09-23...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 11.18it/s]
Retrieved 56459 rows
Fetching data from 2024-09-24 to 2024-10-08...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 12.04it/s]
/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pybaseball/
statcast.py:85: FutureWarning: The behavior of DataFrame concatenation with
empty or all-NA entries is deprecated. In a future version, this will no lon
ger exclude empty or all-NA columns when determining the result dtypes. To r
etain the old behavior, exclude the relevant entries before the concat opera
tion.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_stri
ng=False)
Retrieved 30004 rows
Fetching data from 2024-10-09 to 2024-10-23...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 10.78it/s]
/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pybaseball/
statcast.py:85: FutureWarning: The behavior of DataFrame concatenation with
empty or all-NA entries is deprecated. In a future version, this will no lon
ger exclude empty or all-NA columns when determining the result dtypes. To r
etain the old behavior, exclude the relevant entries before the concat opera
tion.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_stri
ng=False)
```

```
Retrieved 5795 rows
Fetching data from 2024-10-24 to 2024-11-01...
This is a large query, it may take a moment to complete
100%|██████████| 9/9 [00:00<00:00, 11.68it/s]
/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pybaseball/statcast.py:85: FutureWarning: The behavior of DataFrame concatenation with empty or all-NA entries is deprecated. In a future version, this will no longer exclude empty or all-NA columns when determining the result dtypes. To retain the old behavior, exclude the relevant entries before the concat operation.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_string=False)
Retrieved 1576 rows

Initial data statistics:
Total pitches: 661563
Unique pitchers: 853
```



```
Summary Statistics by Days of Rest:
          arm_angle_variance                        pitcher
                        mean  median     std count nunique
days_rest
0                       8.51   4.615   7.398    10      10
1                     13.596   9.991  16.414   385     385
2                     13.934  10.651  15.158   447     447
3                     13.553   10.85   14.21   440     440
4                     12.843  10.177  10.736   423     423
5                     13.133  10.717  10.395   579     579
6                     13.171  10.447  10.896   520     520
7                     12.459   9.422  12.178   390     390
8                      11.09   8.354   9.895   198     198
9                     12.593   9.051  13.565   124     124
10                    11.949   9.586   8.774    75      75

Pitch Count Distribution:
            count        mean         std  min    25%    50%    75%     max
days_rest
0            10.0        13.7    6.147267  5.0    8.5   12.5   19.0    22.0
1           385.0   98.516883   78.152531  2.0   34.0   75.0  150.0   343.0
2           447.0  155.727069  118.680778  3.0   49.5  126.0  239.5   521.0
3           440.0     134.575   93.767409  3.0  54.75  119.0  198.0   567.0
4           423.0   92.579196   68.969138  2.0   38.0   79.0  129.0   394.0
5           579.0  253.481865  326.015609  3.0   42.0   89.0  356.5  1524.0
6           520.0  341.755769  445.391132  3.0  28.75   90.5  551.0  1762.0
7           390.0  115.884615  127.039048  2.0   22.0   75.5  174.0   680.0
8           198.0   55.358586   51.814171  2.0   17.0   35.0   85.0   379.0
9           124.0   54.846774   45.247655  4.0  17.75   37.5  86.25   198.0
10           75.0   58.906667   39.283162  2.0   22.0   66.0   91.0   186.0
```

In [42]:

```python
from pybaseball import statcast
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime, timedelta

def get_statcast_data(start_date, end_date, max_days=14):
    """
    Safely retrieve Statcast data in chunks, skipping problematic dates.
    """
    start_dt = datetime.strptime(start_date, '%Y-%m-%d')
    end_dt = datetime.strptime(end_date, '%Y-%m-%d')

    all_data = []
    current_date = start_dt

    # Known problematic date ranges to skip
    skip_ranges = [
        (datetime(2024, 4, 12), datetime(2024, 4, 17)),
    ]

    print("Retrieving Statcast data in chunks...")
    while current_date <= end_dt:
        chunk_end = min(current_date + timedelta(days=max_days), end_dt)
```

```
        # Check if current chunk overlaps with any skip ranges
        skip_this_chunk = False
        for skip_start, skip_end in skip_ranges:
            if (current_date <= skip_end and chunk_end >= skip_start):
                skip_this_chunk = True
                print(f"Skipping problematic date range: {current_date.strft
                current_date = skip_end + timedelta(days=1)
                break

        if skip_this_chunk:
            continue

        try:
            print(f"Fetching data from {current_date.strftime('%Y-%m-%d')} t
            chunk_data = statcast(
                start_dt=current_date.strftime('%Y-%m-%d'),
                end_dt=chunk_end.strftime('%Y-%m-%d')
            )

            if chunk_data is not None and not chunk_data.empty:
                all_data.append(chunk_data)
                print(f"Retrieved {len(chunk_data)} rows")
            else:
                print("No data found for this date range")

        except Exception as e:
            print(f"Error retrieving chunk: {str(e)}")

        current_date = chunk_end + timedelta(days=1)

    if not all_data:
        raise ValueError("No data could be retrieved from Statcast")

    return pd.concat(all_data, ignore_index=True)

def analyze_variance_vs_performance(start_date, end_date, min_pitches=500):
    """
    Analyze relationship between pitcher's arm angle variance and their perf
    """
    try:
        # Get data
        data = get_statcast_data(start_date, end_date)

        print(f"\nInitial data statistics:")
        print(f"Total pitches: {len(data)}")
        print(f"Unique pitchers: {data['pitcher'].nunique()}")

        # Clean and prepare data
        analysis_data = data[['pitcher', 'player_name', 'arm_angle',
                              'estimated_woba_using_speedangle', 'home_team',
                              'away_team', 'p_throws']].dropna()

        # Calculate pitcher-level statistics
        pitcher_stats = []
```

```
        for pitcher in analysis_data['pitcher'].unique():
            pitcher_data = analysis_data[analysis_data['pitcher'] == pitcher
            total_pitches = len(pitcher_data)

            if total_pitches >= min_pitches:  # Only include pitchers with 1
                # Get most frequent team
                team = (pd.concat([
                    pitcher_data['home_team'],
                    pitcher_data['away_team']
                ]).mode()[0])

                pitcher_stats.append({
                    'pitcher': pitcher,
                    'player_name': pitcher_data['player_name'].iloc[0],
                    'team': team,
                    'throws': pitcher_data['p_throws'].iloc[0],
                    'total_pitches': total_pitches,
                    'arm_angle_variance': pitcher_data['arm_angle'].var(),
                    'arm_angle_std': pitcher_data['arm_angle'].std(),
                    'arm_angle_mean': pitcher_data['arm_angle'].mean(),
                    'xwoba': pitcher_data['estimated_woba_using_speedangle']
                })

        # Create DataFrame
        analysis_df = pd.DataFrame(pitcher_stats)

        print(f"\nPitchers with {min_pitches}+ pitches: {len(analysis_df)}")

        # Calculate correlation
        correlation = analysis_df['arm_angle_variance'].corr(analysis_df['xw

        # Create visualization
        fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(20, 10))

        # Plot 1: Top 30 Most Variable Pitchers
        top_variable = analysis_df.nlargest(30, 'arm_angle_variance')

        bars = ax1.bar(range(len(top_variable)),
                       top_variable['arm_angle_variance'],
                       color=plt.cm.RdYlBu(np.linspace(0, 1, len(top_variable

        ax1.set_title(f'30 Pitchers with Highest Arm Angle Variance\n(Minimu
                      pad=20, fontsize=14)
        ax1.set_xlabel('Pitcher', fontsize=12)
        ax1.set_ylabel('Arm Angle Variance', fontsize=12)

        # Add pitcher labels
        ax1.set_xticks(range(len(top_variable)))
        ax1.set_xticklabels([f"{name}\n{team}\n({pitches} pitches)"
                             for name, team, pitches in zip(top_variable['pla
                                                            top_variable['team'
                                                            top_variable['total
                            rotation=45, ha='right', fontsize=10)

        # Add value labels
        for i, bar in enumerate(bars):
```

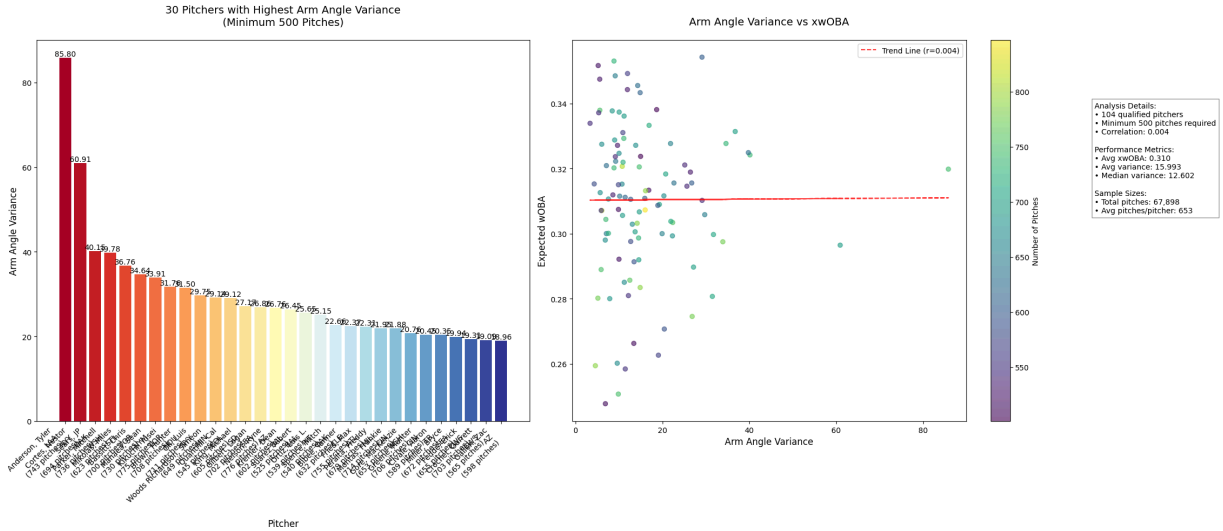
```
        height = bar.get_height()
        ax1.text(bar.get_x() + bar.get_width()/2., height,
                f'{height:.2f}',
                ha='center', va='bottom', fontsize=10)

    # Plot 2: Scatter plot of Variance vs xwOBA
    scatter = ax2.scatter(analysis_df['arm_angle_variance'],
                          analysis_df['xwoba'],
                          alpha=0.6,
                          c=analysis_df['total_pitches'],
                          cmap='viridis')

    # Add trend line
    z = np.polyfit(analysis_df['arm_angle_variance'],
                   analysis_df['xwoba'], 1)
    p = np.poly1d(z)
    ax2.plot(analysis_df['arm_angle_variance'],
            p(analysis_df['arm_angle_variance']),
            "r--", alpha=0.8, label=f'Trend Line (r={correlation:.3f})')

    ax2.set_title('Arm Angle Variance vs xwOBA', pad=20, fontsize=14)
    ax2.set_xlabel('Arm Angle Variance', fontsize=12)
    ax2.set_ylabel('Expected wOBA', fontsize=12)

    # Add colorbar
    plt.colorbar(scatter, ax=ax2, label='Number of Pitches')

    ax2.legend()

    # Add explanation text
    explanation_text = (
        f"Analysis Details:\n"
        f"• {len(analysis_df)} qualified pitchers\n"
        f"• Minimum {min_pitches} pitches required\n"
        f"• Correlation: {correlation:.3f}\n\n"
        f"Performance Metrics:\n"
        f"• Avg xwOBA: {analysis_df['xwoba'].mean():.3f}\n"
        f"• Avg variance: {analysis_df['arm_angle_variance'].mean():.3f}
        f"• Median variance: {analysis_df['arm_angle_variance'].median()
        f"Sample Sizes:\n"
        f"• Total pitches: {analysis_df['total_pitches'].sum():,}\n"
        f"• Avg pitches/pitcher: {analysis_df['total_pitches'].mean():.0
    )
    plt.figtext(1.02, 0.6, explanation_text,
               bbox=dict(facecolor='white', alpha=0.8, edgecolor='gray')
               fontsize=10)

    plt.tight_layout()
    plt.show()

    # Print detailed statistics
    print("\nTop 10 Most Variable Pitchers:")
    print(top_variable[['player_name', 'team', 'arm_angle_variance',
                        'xwoba', 'total_pitches']].head(10).to_string())

    # Performance analysis
```

```
        high_var = analysis_df[analysis_df['arm_angle_variance'] >
                               analysis_df['arm_angle_variance'].median()]
        low_var = analysis_df[analysis_df['arm_angle_variance'] <=
                              analysis_df['arm_angle_variance'].median()]

        print("\nPerformance Comparison:")
        print(f"High Variance Pitchers (n={len(high_var)}):")
        print(f"Average xwOBA: {high_var['xwoba'].mean():.3f}")
        print(f"Average pitches: {high_var['total_pitches'].mean():.0f}")
        print(f"\nLow Variance Pitchers (n={len(low_var)}):")
        print(f"Average xwOBA: {low_var['xwoba'].mean():.3f}")
        print(f"Average pitches: {low_var['total_pitches'].mean():.0f}")

        return analysis_df

    except Exception as e:
        print(f"Analysis failed: {str(e)}")
        raise

# Run analysis for full season
if __name__ == "__main__":
    try:
        results = analyze_variance_vs_performance("2024-03-28", "2024-11-01"
    except Exception as e:
        print(f"Analysis failed: {str(e)}")
```

```
Retrieving Statcast data in chunks...
Fetching data from 2024-03-28 to 2024-04-11...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00,  9.30it/s]
Retrieved 55595 rows
Skipping problematic date range: 2024-04-12 to 2024-04-26
Fetching data from 2024-04-18 to 2024-05-02...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00,  9.25it/s]
Retrieved 55630 rows
Fetching data from 2024-05-03 to 2024-05-17...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00,  9.28it/s]
Retrieved 58443 rows
Fetching data from 2024-05-18 to 2024-06-01...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 10.13it/s]
Retrieved 58312 rows
Fetching data from 2024-06-02 to 2024-06-16...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 10.10it/s]
Retrieved 57626 rows
Fetching data from 2024-06-17 to 2024-07-01...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00,  9.87it/s]
Retrieved 54877 rows
Fetching data from 2024-07-02 to 2024-07-16...
This is a large query, it may take a moment to complete
```

```
100%|██████████| 15/15 [00:02<00:00,  6.37it/s]
/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pybaseball/
statcast.py:85: FutureWarning: The behavior of DataFrame concatenation with
empty or all-NA entries is deprecated. In a future version, this will no lon
ger exclude empty or all-NA columns when determining the result dtypes. To r
etain the old behavior, exclude the relevant entries before the concat opera
tion.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_stri
ng=False)
Retrieved 54098 rows
Fetching data from 2024-07-17 to 2024-07-31...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:02<00:00,  6.54it/s]
/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pybaseball/
statcast.py:85: FutureWarning: The behavior of DataFrame concatenation with
empty or all-NA entries is deprecated. In a future version, this will no lon
ger exclude empty or all-NA columns when determining the result dtypes. To r
etain the old behavior, exclude the relevant entries before the concat opera
tion.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_stri
ng=False)
Retrieved 53338 rows
Fetching data from 2024-08-01 to 2024-08-15...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 10.09it/s]
Retrieved 56649 rows
Fetching data from 2024-08-16 to 2024-08-30...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 10.16it/s]
Retrieved 60147 rows
Fetching data from 2024-08-31 to 2024-09-14...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00,  9.42it/s]
Retrieved 58282 rows
Fetching data from 2024-09-15 to 2024-09-29...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00,  9.60it/s]
Retrieved 58272 rows
Fetching data from 2024-09-30 to 2024-10-14...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:06<00:00,  2.47it/s]
Retrieved 9176 rows
Fetching data from 2024-10-15 to 2024-10-29...
This is a large query, it may take a moment to complete
/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pybaseball/
statcast.py:85: FutureWarning: The behavior of DataFrame concatenation with
empty or all-NA entries is deprecated. In a future version, this will no lon
ger exclude empty or all-NA columns when determining the result dtypes. To r
etain the old behavior, exclude the relevant entries before the concat opera
tion.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_stri
ng=False)
100%|██████████| 15/15 [00:02<00:00,  6.11it/s]
```

```
Retrieved 3845 rows
Fetching data from 2024-10-30 to 2024-11-01...
This is a large query, it may take a moment to complete
/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pybaseball/statcast.py:85: FutureWarning: The behavior of DataFrame concatenation with empty or all-NA entries is deprecated. In a future version, this will no longer exclude empty or all-NA columns when determining the result dtypes. To retain the old behavior, exclude the relevant entries before the concat operation.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_string=False)
100%|██████████| 3/3 [00:00<00:00, 11.17it/s]
/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pybaseball/statcast.py:85: FutureWarning: The behavior of DataFrame concatenation with empty or all-NA entries is deprecated. In a future version, this will no longer exclude empty or all-NA columns when determining the result dtypes. To retain the old behavior, exclude the relevant entries before the concat operation.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_string=False)
Retrieved 342 rows

Initial data statistics:
Total pitches: 694632
Unique pitchers: 854

Pitchers with 500+ pitches: 104
```



```
Top 10 Most Variable Pitchers:
         player_name team  arm_angle_variance     xwoba  total_pitches
38   Anderson, Tyler  LAA           85.803973  0.319856            743
41    Cortes, Nestor  NYY           60.909429  0.296481            694
4          Sears, JP  OAK           40.146268  0.324218            736
90  Parker, Mitchell  WSH           39.781968  0.324939            623
44    Mikolas, Miles  STL           36.755174  0.331429            700
34    Bassitt, Chris  TOR           34.640822  0.327739            730
52      Manaea, Sean  NYM           33.906637  0.297521            775
19    Kikuchi, Yusei  TOR           31.756989  0.299753            708
2      Brown, Hunter  HOU           31.500731  0.280722            711
60         Gil, Luis  NYY           29.747163  0.305853            649

Performance Comparison:
High Variance Pitchers (n=52):
Average xwOBA: 0.309
Average pitches: 660

Low Variance Pitchers (n=52):
Average xwOBA: 0.312
Average pitches: 646
```

In [41]:

```python
from pybaseball import statcast
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime, timedelta

def get_statcast_data(start_date, end_date, max_days=14):
    """
    Safely retrieve Statcast data in chunks, skipping problematic dates.
    """
    start_dt = datetime.strptime(start_date, '%Y-%m-%d')
    end_dt = datetime.strptime(end_date, '%Y-%m-%d')

    all_data = []
    current_date = start_dt

    # Known problematic date ranges to skip
    skip_ranges = [
        (datetime(2024, 4, 12), datetime(2024, 4, 15))
    ]

    print("Retrieving Statcast data in chunks...")
    while current_date <= end_dt:
        chunk_end = min(current_date + timedelta(days=max_days), end_dt)

        # Check if current chunk overlaps with any skip ranges
        skip_this_chunk = False
        for skip_start, skip_end in skip_ranges:
            if (current_date <= skip_end and chunk_end >= skip_start):
                skip_this_chunk = True
                print(f"Skipping problematic date range: {current_date.strft
                current_date = skip_end + timedelta(days=1)
                break
```

```
        if skip_this_chunk:
            continue

        try:
            print(f"Fetching data from {current_date.strftime('%Y-%m-%d')} t
            chunk_data = statcast(
                start_dt=current_date.strftime('%Y-%m-%d'),
                end_dt=chunk_end.strftime('%Y-%m-%d')
            )

            if chunk_data is not None and not chunk_data.empty:
                all_data.append(chunk_data)
                print(f"Retrieved {len(chunk_data)} rows")
            else:
                print("No data found for this date range")

        except Exception as e:
            print(f"Error retrieving chunk: {str(e)}")

        current_date = chunk_end + timedelta(days=1)

    if not all_data:
        raise ValueError("No data could be retrieved from Statcast")

    return pd.concat(all_data, ignore_index=True)

def create_team_variance_plot(team_stats):
    """
    Create visualization of team-level arm angle variance.
    """
    plt.figure(figsize=(20, 10))

    # Create bars with gradient color
    bars = plt.bar(
        range(len(team_stats)),
        team_stats['mean_variance'],
        color=plt.cm.RdYlGn_r(np.linspace(0, 1, len(team_stats)))
    )

    # Customize plot
    plt.title('MLB Teams Ranked by Pitching Staff Arm Angle Variance (2024 S
              pad=20, fontsize=14)
    plt.xlabel('Team', fontsize=12)
    plt.ylabel('Mean Arm Angle Variance', fontsize=12)

    # Add team labels with pitcher counts
    plt.xticks(
        range(len(team_stats)),
        [f"{team}\n{pitchers} pitchers\n({pitches:,} pitches)"
         for team, pitchers, pitches in zip(team_stats['team'],
                                            team_stats['pitcher_count'],
                                            team_stats['total_pitches'])],
        rotation=45,
        ha='right',
        fontsize=10
```

```
    )

    # Add value labels on bars
    for i, bar in enumerate(bars):
        height = bar.get_height()
        plt.text(
            bar.get_x() + bar.get_width()/2.,
            height,
            f'{height:.2f}',
            ha='center',
            va='bottom',
            fontsize=10
        )

    # Add explanation text
    explanation_text = (
        f"Analysis Details:\n"
        f"• {team_stats['pitcher_count'].sum()} qualified pitchers\n"
        f"• {team_stats['total_pitches'].sum():,} total pitches\n"
        f"• Minimum 100 pitches per pitcher\n\n"
        f"Team-Level Metrics:\n"
        f"• League Avg Variance: {team_stats['mean_variance'].mean():.2f}\n"
        f"• League Median Variance: {team_stats['mean_variance'].median():.2
        f"• Avg Pitchers per Team: {team_stats['pitcher_count'].mean():.1f}\
        f"Color Scale:\n"
        f"Green = More Consistent\n"
        f"Red = Less Consistent"
    )
    plt.text(1.02, 0.98, explanation_text,
            transform=plt.gca().transAxes,
            bbox=dict(facecolor='white', alpha=0.8, edgecolor='gray'),
            fontsize=10)

    plt.grid(True, axis='y', linestyle='--', alpha=0.3)
    plt.tight_layout()

    return plt

def analyze_team_variance(start_date, end_date, min_pitches=30):
    """
    Analyze and visualize arm angle variance at the team level.
    """
    try:
        # Get data
        data = get_statcast_data(start_date, end_date)

        print(f"\nInitial data statistics:")
        print(f"Total pitches: {len(data)}")
        print(f"Unique pitchers: {data['pitcher'].nunique()}")

        # Clean and prepare data
        analysis_data = data[['pitcher', 'player_name', 'arm_angle',
                              'estimated_woba_using_speedangle', 'home_team',
                              'away_team', 'p_throws']].dropna()

        # Calculate pitcher-level statistics
```

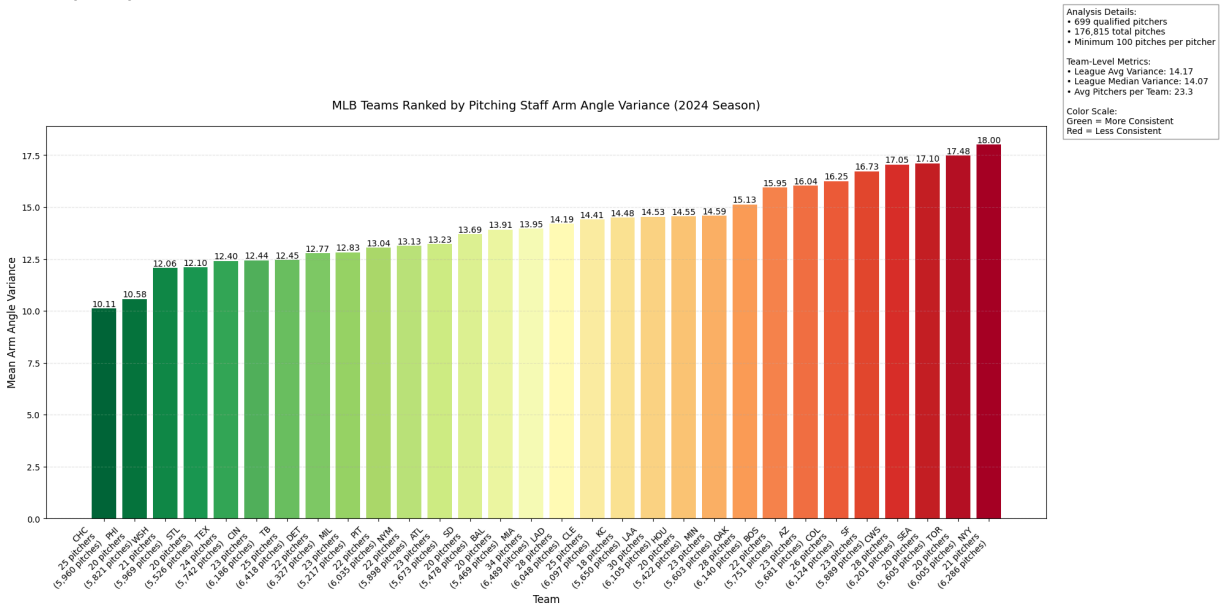
```
        pitcher_stats = []

        for pitcher in analysis_data['pitcher'].unique():
            pitcher_data = analysis_data[analysis_data['pitcher'] == pitcher
            total_pitches = len(pitcher_data)

            if total_pitches >= min_pitches:
                # Get most frequent team
                team = (pd.concat([
                    pitcher_data['home_team'],
                    pitcher_data['away_team']
                ]).mode()[0])

                # Calculate statistics
                pitcher_stats.append({
                    'pitcher': pitcher,
                    'player_name': pitcher_data['player_name'].iloc[0],
                    'team': team,
                    'throws': pitcher_data['p_throws'].iloc[0],
                    'total_pitches': total_pitches,
                    'arm_angle_variance': pitcher_data['arm_angle'].var(),
                    'arm_angle_std': pitcher_data['arm_angle'].std(),
                    'arm_angle_mean': pitcher_data['arm_angle'].mean(),
                    'xwoba': pitcher_data['estimated_woba_using_speedangle']
                })

        # Create DataFrames
        pitcher_df = pd.DataFrame(pitcher_stats)

        # Calculate team-level metrics
        team_stats = pitcher_df.groupby('team').agg({
            'arm_angle_variance': ['mean', 'median', 'std'],
            'xwoba': 'mean',
            'pitcher': 'count',
            'total_pitches': 'sum'
        }).reset_index()

        # Flatten column names
        team_stats.columns = ['team', 'mean_variance', 'median_variance', 'v
                              'mean_xwoba', 'pitcher_count', 'total_pitches']

        # Sort teams by mean variance
        team_stats = team_stats.sort_values('mean_variance')

        # Create and show visualization
        plt = create_team_variance_plot(team_stats)
        plt.show()

        # Print summary statistics
        print("\nTeam Rankings (Lowest to Highest Variance):")
        print(team_stats[['team', 'mean_variance', 'median_variance', 'pitch
                          'mean_xwoba']].to_string())

        # Calculate correlation between team variance and performance
        correlation = team_stats['mean_variance'].corr(team_stats['mean_xwob
        print(f"\nCorrelation between team variance and xwOBA: {correlation:
```

```
        # Print most/least consistent teams
        print("\nMost Consistent Teams (Lowest Variance):")
        print(team_stats.head(5)[['team', 'mean_variance', 'pitcher_count',
                                  'mean_xwoba']].to_string())

        print("\nLeast Consistent Teams (Highest Variance):")
        print(team_stats.tail(5)[['team', 'mean_variance', 'pitcher_count',
                                  'mean_xwoba']].to_string())

        # Add supplementary team analysis
        print("\nTeam Consistency Metrics:")
        print("Teams with most consistent individual pitchers:")
        consistent_teams = team_stats.nsmallest(5, 'variance_std')[
            ['team', 'variance_std', 'pitcher_count']]
        print(consistent_teams.to_string())

        return {
            'team_stats': team_stats,
            'pitcher_df': pitcher_df,
            'correlation': correlation
        }

    except Exception as e:
        print(f"Analysis failed: {str(e)}")
        raise

# Run analysis
if __name__ == "__main__":
    try:
        results = analyze_team_variance("2024-03-28", "2024-11-01")
    except Exception as e:
        print(f"Analysis failed: {str(e)}")
```

```
Retrieving Statcast data in chunks...
Fetching data from 2024-03-28 to 2024-04-11...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00,  9.97it/s]
Retrieved 55595 rows
Skipping problematic date range: 2024-04-12 to 2024-04-26
Fetching data from 2024-04-16 to 2024-04-30...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 11.51it/s]
Retrieved 58566 rows
Fetching data from 2024-05-01 to 2024-05-15...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00,  9.68it/s]
Retrieved 58196 rows
Fetching data from 2024-05-16 to 2024-05-30...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 11.26it/s]
Retrieved 55791 rows
Fetching data from 2024-05-31 to 2024-06-14...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 12.13it/s]
```

```
Retrieved 57619 rows
Fetching data from 2024-06-15 to 2024-06-29...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 11.20it/s]
Retrieved 58354 rows
Fetching data from 2024-06-30 to 2024-07-14...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 11.43it/s]
Retrieved 59343 rows
Fetching data from 2024-07-15 to 2024-07-29...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00,  8.69it/s]
/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pybaseball/
statcast.py:85: FutureWarning: The behavior of DataFrame concatenation with
empty or all-NA entries is deprecated. In a future version, this will no lon
ger exclude empty or all-NA columns when determining the result dtypes. To r
etain the old behavior, exclude the relevant entries before the concat opera
tion.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_stri
ng=False)
Retrieved 44767 rows
Fetching data from 2024-07-30 to 2024-08-13...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00,  9.63it/s]
Retrieved 58604 rows
Fetching data from 2024-08-14 to 2024-08-28...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00,  9.76it/s]
Retrieved 58912 rows
Fetching data from 2024-08-29 to 2024-09-12...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:02<00:00,  6.99it/s]
Retrieved 57453 rows
Fetching data from 2024-09-13 to 2024-09-27...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00,  9.69it/s]
Retrieved 58506 rows
Fetching data from 2024-09-28 to 2024-10-12...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00, 10.18it/s]
/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pybaseball/
statcast.py:85: FutureWarning: The behavior of DataFrame concatenation with
empty or all-NA entries is deprecated. In a future version, this will no lon
ger exclude empty or all-NA columns when determining the result dtypes. To r
etain the old behavior, exclude the relevant entries before the concat opera
tion.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_stri
ng=False)
Retrieved 16693 rows
Fetching data from 2024-10-13 to 2024-10-27...
This is a large query, it may take a moment to complete
100%|██████████| 15/15 [00:01<00:00,  9.08it/s]
```

```
Retrieved 4127 rows
Fetching data from 2024-10-28 to 2024-11-01...
This is a large query, it may take a moment to complete
/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pybaseball/statcast.py:85: FutureWarning: The behavior of DataFrame concatenation with empty or all-NA entries is deprecated. In a future version, this will no longer exclude empty or all-NA columns when determining the result dtypes. To retain the old behavior, exclude the relevant entries before the concat operation.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_string=False)
100%|██████████| 5/5 [00:00<00:00, 12.83it/s]
/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pybaseball/statcast.py:85: FutureWarning: The behavior of DataFrame concatenation with empty or all-NA entries is deprecated. In a future version, this will no longer exclude empty or all-NA columns when determining the result dtypes. To retain the old behavior, exclude the relevant entries before the concat operation.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_string=False)
Retrieved 989 rows

Initial data statistics:
Total pitches: 703515
Unique pitchers: 854
```



```
Team Rankings (Lowest to Highest Variance):
   team  mean_variance  median_variance  pitcher_count  mean_xwoba
4   CHC      10.110287        10.180263             25    0.323324
20  PHI      10.578888         9.112760             20    0.323365
29  WSH      12.064950        10.895248             21    0.324356
25  STL      12.097402        11.225459             20    0.320188
27  TEX      12.404154        10.733608             24    0.320497
5   CIN      12.435431        11.774688             23    0.308776
26   TB      12.445681        10.831621             25    0.303253
9   DET      12.774514        13.260832             22    0.306578
15  MIL      12.826726        10.512867             23    0.317216
21  PIT      13.036217        12.398134             22    0.325160
17  NYM      13.127575        11.678712             22    0.315720
0   ATL      13.227392         8.028150             23    0.309922
22   SD      13.690499        12.518204             20    0.304134
2   BAL      13.910431        12.082670             20    0.306467
14  MIA      13.953187        12.667896             34    0.325541
13  LAD      14.194162        13.169551             28    0.323210
6   CLE      14.408934        12.201865             25    0.299359
11   KC      14.484783        12.684228             18    0.323202
12  LAA      14.532692        11.362277             30    0.331289
10  HOU      14.548762        12.300111             20    0.313403
16  MIN      14.588652        11.859928             23    0.307943
19  OAK      15.132042        13.154979             28    0.315397
3   BOS      15.947450        13.217732             22    0.314444
1    AZ      16.038259        11.323843             23    0.342797
7   COL      16.252863        11.449478             26    0.345433
24   SF      16.728852        11.731176             23    0.315772
8   CWS      17.046738        12.587578             28    0.331932
23  SEA      17.103119        12.140824             20    0.292012
28  TOR      17.478949        12.686010             20    0.326642
18  NYY      18.002928        14.086431             21    0.309376

Correlation between team variance and xwOBA: 0.071

Most Consistent Teams (Lowest Variance):
   team  mean_variance  pitcher_count  mean_xwoba
4   CHC      10.110287             25    0.323324
20  PHI      10.578888             20    0.323365
29  WSH      12.064950             21    0.324356
25  STL      12.097402             20    0.320188
27  TEX      12.404154             24    0.320497

Least Consistent Teams (Highest Variance):
   team  mean_variance  pitcher_count  mean_xwoba
24   SF      16.728852             23    0.315772
8   CWS      17.046738             28    0.331932
23  SEA      17.103119             20    0.292012
28  TOR      17.478949             20    0.326642
18  NYY      18.002928             21    0.309376

Team Consistency Metrics:
Teams with most consistent individual pitchers:
   team  variance_std  pitcher_count
4   CHC      4.559548             25
20  PHI      4.880326             20
```

```
9   DET      4.894009            22
26   TB      5.117908            25
5   CIN      5.261751            23
```

In [40]:

```
from pybaseball import statcast
import pandas as pd
import numpy as np
from datetime import datetime, timedelta
import traceback

def debug_data_retrieval(start_date, end_date, days_per_chunk=3):
    """
    Debug Statcast data retrieval by trying smaller chunks and printing deta
    """
    start_dt = datetime.strptime(start_date, '%Y-%m-%d')
    end_dt = datetime.strptime(end_date, '%Y-%m-%d')

    all_data = []
    current_date = start_dt

    print(f"\nDebug Analysis:")
    print(f"Testing date range from {start_date} to {end_date}")
    print(f"Using {days_per_chunk} days per chunk")

    while current_date <= end_dt:
        chunk_end = min(current_date + timedelta(days=days_per_chunk), end_d

        try:
            print(f"\nAttempting to fetch: {current_date.strftime('%Y-%m-%d'

            chunk_data = statcast(
                start_dt=current_date.strftime('%Y-%m-%d'),
                end_dt=chunk_end.strftime('%Y-%m-%d')
            )

            if chunk_data is not None and not chunk_data.empty:
                print(f"Success! Retrieved {len(chunk_data)} rows")
                print(f"Columns present: {chunk_data.columns.tolist()}")
                print(f"Sample of data shape: {chunk_data.shape}")
                all_data.append(chunk_data)
            else:
                print("No data returned for this date range")

        except Exception as e:
            print(f"\nError details for {current_date.strftime('%Y-%m-%d')}
            print(f"Error type: {type(e).__name__}")
            print(f"Error message: {str(e)}")
            print("\nFull traceback:")
            print(traceback.format_exc())

        current_date = chunk_end + timedelta(days=1)

    if all_data:
        combined_data = pd.concat(all_data, ignore_index=True)
        print(f"\nFinal combined dataset:")
        print(f"Total rows: {len(combined_data)}")
```

```
        print(f"Columns: {combined_data.columns.tolist()}")
        return combined_data
    else:
        print("\nNo data was successfully retrieved")
        return None

# Test the problematic date ranges specifically
if __name__ == "__main__":
    # Test first problematic range
    print("\nTesting first problematic range:")
    data1 = debug_data_retrieval("2024-04-12", "2024-04-26")

    # Test second problematic range
    print("\nTesting second problematic range:")
    data2 = debug_data_retrieval("2024-06-26", "2024-07-10")

    # Test a working range for comparison
    print("\nTesting known working range:")
    data3 = debug_data_retrieval("2024-05-01", "2024-05-15")
```

```
Testing first problematic range:

Debug Analysis:
Testing date range from 2024-04-12 to 2024-04-26
Using 3 days per chunk

Attempting to fetch: 2024-04-12 to 2024-04-15
This is a large query, it may take a moment to complete
  0%|          | 0/4 [00:00<?, ?it/s]
```

```
Error details for 2024-04-12 to 2024-04-15:
Error type: KeyError
Error message: 'game_date'

Full traceback:
Traceback (most recent call last):
  File "/var/folders/bk/465dzg3j4v9f90yvbgx02ly80000gn/T/ipykernel_62931/228
3342252.py", line 27, in debug_data_retrieval
    chunk_data = statcast(
  File "/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pyb
aseball/statcast.py", line 113, in statcast
    return _handle_request(start_dt_date, end_dt_date, 1, verbose=verbose,
  File "/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pyb
aseball/statcast.py", line 76, in _handle_request
    dataframe_list.append(future.result())
  File "/Library/Developer/CommandLineTools/Library/Frameworks/Python3.frame
work/Versions/3.9/lib/python3.9/concurrent/futures/_base.py", line 438, in r
esult
    return self.__get_result()
  File "/Library/Developer/CommandLineTools/Library/Frameworks/Python3.frame
work/Versions/3.9/lib/python3.9/concurrent/futures/_base.py", line 390, in _
_get_result
    raise self._exception
  File "/Library/Developer/CommandLineTools/Library/Frameworks/Python3.frame
work/Versions/3.9/lib/python3.9/concurrent/futures/thread.py", line 52, in r
un
    result = self.fn(*self.args, **self.kwargs)
  File "/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pyb
aseball/cache/cache.py", line 58, in _cached
    result = func(*args, **kwargs)
  File "/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pyb
aseball/statcast.py", line 31, in _small_request
    data = data.sort_values(
  File "/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pan
das/core/frame.py", line 7159, in sort_values
    keys = [self._get_label_or_level_values(x, axis=axis) for x in by]
  File "/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pan
das/core/frame.py", line 7159, in <listcomp>
    keys = [self._get_label_or_level_values(x, axis=axis) for x in by]
  File "/Users/rohitkrishnan/Library/Python/3.9/lib/python/site-packages/pan
das/core/generic.py", line 1910, in _get_label_or_level_values
    raise KeyError(key)
KeyError: 'game_date'


Attempting to fetch: 2024-04-16 to 2024-04-19
This is a large query, it may take a moment to complete
100%|██████████| 4/4 [00:00<00:00, 11.71it/s]
```

```
Success! Retrieved 14103 rows
Columns present: ['pitch_type', 'game_date', 'release_speed', 'release_pos_
x', 'release_pos_z', 'player_name', 'batter', 'pitcher', 'events', 'descript
ion', 'spin_dir', 'spin_rate_deprecated', 'break_angle_deprecated', 'break_l
ength_deprecated', 'zone', 'des', 'game_type', 'stand', 'p_throws', 'home_te
am', 'away_team', 'type', 'hit_location', 'bb_type', 'balls', 'strikes', 'ga
me_year', 'pfx_x', 'pfx_z', 'plate_x', 'plate_z', 'on_3b', 'on_2b', 'on_1b',
'outs_when_up', 'inning', 'inning_topbot', 'hc_x', 'hc_y', 'tfs_deprecated',
'tfs_zulu_deprecated', 'umpire', 'sv_id', 'vx0', 'vy0', 'vz0', 'ax', 'ay',
'az', 'sz_top', 'sz_bot', 'hit_distance_sc', 'launch_speed', 'launch_angle',
'effective_speed', 'release_spin_rate', 'release_extension', 'game_pk', 'fie
lder_2', 'fielder_3', 'fielder_4', 'fielder_5', 'fielder_6', 'fielder_7', 'f
ielder_8', 'fielder_9', 'release_pos_y', 'estimated_ba_using_speedangle', 'e
stimated_woba_using_speedangle', 'woba_value', 'woba_denom', 'babip_value',
'iso_value', 'launch_speed_angle', 'at_bat_number', 'pitch_number', 'pitch_n
ame', 'home_score', 'away_score', 'bat_score', 'fld_score', 'post_away_scor
e', 'post_home_score', 'post_bat_score', 'post_fld_score', 'if_fielding_alig
nment', 'of_fielding_alignment', 'spin_axis', 'delta_home_win_exp', 'delta_r
un_exp', 'bat_speed', 'swing_length', 'estimated_slg_using_speedangle', 'del
ta_pitcher_run_exp', 'hyper_speed', 'home_score_diff', 'bat_score_diff', 'ho
me_win_exp', 'bat_win_exp', 'age_pit_legacy', 'age_bat_legacy', 'age_pit',
'age_bat', 'n_thruorder_pitcher', 'n_priorpa_thisgame_player_at_bat', 'pitch
er_days_since_prev_game', 'batter_days_since_prev_game', 'pitcher_days_until
_next_game', 'batter_days_until_next_game', 'api_break_z_with_gravity', 'api
_break_x_arm', 'api_break_x_batter_in', 'arm_angle']
Sample of data shape: (14103, 113)

Attempting to fetch: 2024-04-20 to 2024-04-23
This is a large query, it may take a moment to complete
100%|██████████| 4/4 [00:00<00:00, 21.50it/s]
```

```
Success! Retrieved 16601 rows
Columns present: ['pitch_type', 'game_date', 'release_speed', 'release_pos_
x', 'release_pos_z', 'player_name', 'batter', 'pitcher', 'events', 'descript
ion', 'spin_dir', 'spin_rate_deprecated', 'break_angle_deprecated', 'break_l
ength_deprecated', 'zone', 'des', 'game_type', 'stand', 'p_throws', 'home_te
am', 'away_team', 'type', 'hit_location', 'bb_type', 'balls', 'strikes', 'ga
me_year', 'pfx_x', 'pfx_z', 'plate_x', 'plate_z', 'on_3b', 'on_2b', 'on_1b',
'outs_when_up', 'inning', 'inning_topbot', 'hc_x', 'hc_y', 'tfs_deprecated',
'tfs_zulu_deprecated', 'umpire', 'sv_id', 'vx0', 'vy0', 'vz0', 'ax', 'ay',
'az', 'sz_top', 'sz_bot', 'hit_distance_sc', 'launch_speed', 'launch_angle',
'effective_speed', 'release_spin_rate', 'release_extension', 'game_pk', 'fie
lder_2', 'fielder_3', 'fielder_4', 'fielder_5', 'fielder_6', 'fielder_7', 'f
ielder_8', 'fielder_9', 'release_pos_y', 'estimated_ba_using_speedangle', 'e
stimated_woba_using_speedangle', 'woba_value', 'woba_denom', 'babip_value',
'iso_value', 'launch_speed_angle', 'at_bat_number', 'pitch_number', 'pitch_n
ame', 'home_score', 'away_score', 'bat_score', 'fld_score', 'post_away_scor
e', 'post_home_score', 'post_bat_score', 'post_fld_score', 'if_fielding_alig
nment', 'of_fielding_alignment', 'spin_axis', 'delta_home_win_exp', 'delta_r
un_exp', 'bat_speed', 'swing_length', 'estimated_slg_using_speedangle', 'del
ta_pitcher_run_exp', 'hyper_speed', 'home_score_diff', 'bat_score_diff', 'ho
me_win_exp', 'bat_win_exp', 'age_pit_legacy', 'age_bat_legacy', 'age_pit',
'age_bat', 'n_thruorder_pitcher', 'n_priorpa_thisgame_player_at_bat', 'pitch
er_days_since_prev_game', 'batter_days_since_prev_game', 'pitcher_days_until
_next_game', 'batter_days_until_next_game', 'api_break_z_with_gravity', 'api
_break_x_arm', 'api_break_x_batter_in', 'arm_angle']
Sample of data shape: (16601, 113)

Attempting to fetch: 2024-04-24 to 2024-04-26
This is a large query, it may take a moment to complete
100%|██████████| 3/3 [00:00<00:00, 27.46it/s]
Success! Retrieved 11248 rows
Columns present: ['pitch_type', 'game_date', 'release_speed', 'release_pos_
x', 'release_pos_z', 'player_name', 'batter', 'pitcher', 'events', 'descript
ion', 'spin_dir', 'spin_rate_deprecated', 'break_angle_deprecated', 'break_l
ength_deprecated', 'zone', 'des', 'game_type', 'stand', 'p_throws', 'home_te
am', 'away_team', 'type', 'hit_location', 'bb_type', 'balls', 'strikes', 'ga
me_year', 'pfx_x', 'pfx_z', 'plate_x', 'plate_z', 'on_3b', 'on_2b', 'on_1b',
'outs_when_up', 'inning', 'inning_topbot', 'hc_x', 'hc_y', 'tfs_deprecated',
'tfs_zulu_deprecated', 'umpire', 'sv_id', 'vx0', 'vy0', 'vz0', 'ax', 'ay',
'az', 'sz_top', 'sz_bot', 'hit_distance_sc', 'launch_speed', 'launch_angle',
'effective_speed', 'release_spin_rate', 'release_extension', 'game_pk', 'fie
lder_2', 'fielder_3', 'fielder_4', 'fielder_5', 'fielder_6', 'fielder_7', 'f
ielder_8', 'fielder_9', 'release_pos_y', 'estimated_ba_using_speedangle', 'e
stimated_woba_using_speedangle', 'woba_value', 'woba_denom', 'babip_value',
'iso_value', 'launch_speed_angle', 'at_bat_number', 'pitch_number', 'pitch_n
ame', 'home_score', 'away_score', 'bat_score', 'fld_score', 'post_away_scor
e', 'post_home_score', 'post_bat_score', 'post_fld_score', 'if_fielding_alig
nment', 'of_fielding_alignment', 'spin_axis', 'delta_home_win_exp', 'delta_r
un_exp', 'bat_speed', 'swing_length', 'estimated_slg_using_speedangle', 'del
ta_pitcher_run_exp', 'hyper_speed', 'home_score_diff', 'bat_score_diff', 'ho
me_win_exp', 'bat_win_exp', 'age_pit_legacy', 'age_bat_legacy', 'age_pit',
'age_bat', 'n_thruorder_pitcher', 'n_priorpa_thisgame_player_at_bat', 'pitch
er_days_since_prev_game', 'batter_days_since_prev_game', 'pitcher_days_until
_next_game', 'batter_days_until_next_game', 'api_break_z_with_gravity', 'api
_break_x_arm', 'api_break_x_batter_in', 'arm_angle']
Sample of data shape: (11248, 113)
```

```
Final combined dataset:
Total rows: 41952
Columns: ['pitch_type', 'game_date', 'release_speed', 'release_pos_x', 'rele
ase_pos_z', 'player_name', 'batter', 'pitcher', 'events', 'description', 'sp
in_dir', 'spin_rate_deprecated', 'break_angle_deprecated', 'break_length_dep
recated', 'zone', 'des', 'game_type', 'stand', 'p_throws', 'home_team', 'awa
y_team', 'type', 'hit_location', 'bb_type', 'balls', 'strikes', 'game_year',
'pfx_x', 'pfx_z', 'plate_x', 'plate_z', 'on_3b', 'on_2b', 'on_1b', 'outs_whe
n_up', 'inning', 'inning_topbot', 'hc_x', 'hc_y', 'tfs_deprecated', 'tfs_zul
u_deprecated', 'umpire', 'sv_id', 'vx0', 'vy0', 'vz0', 'ax', 'ay', 'az', 'sz
_top', 'sz_bot', 'hit_distance_sc', 'launch_speed', 'launch_angle', 'effecti
ve_speed', 'release_spin_rate', 'release_extension', 'game_pk', 'fielder_2',
'fielder_3', 'fielder_4', 'fielder_5', 'fielder_6', 'fielder_7', 'fielder_
8', 'fielder_9', 'release_pos_y', 'estimated_ba_using_speedangle', 'estimate
d_woba_using_speedangle', 'woba_value', 'woba_denom', 'babip_value', 'iso_va
lue', 'launch_speed_angle', 'at_bat_number', 'pitch_number', 'pitch_name',
'home_score', 'away_score', 'bat_score', 'fld_score', 'post_away_score', 'po
st_home_score', 'post_bat_score', 'post_fld_score', 'if_fielding_alignment',
'of_fielding_alignment', 'spin_axis', 'delta_home_win_exp', 'delta_run_exp',
'bat_speed', 'swing_length', 'estimated_slg_using_speedangle', 'delta_pitche
r_run_exp', 'hyper_speed', 'home_score_diff', 'bat_score_diff', 'home_win_ex
p', 'bat_win_exp', 'age_pit_legacy', 'age_bat_legacy', 'age_pit', 'age_bat',
'n_thruorder_pitcher', 'n_priorpa_thisgame_player_at_bat', 'pitcher_days_sin
ce_prev_game', 'batter_days_since_prev_game', 'pitcher_days_until_next_gam
e', 'batter_days_until_next_game', 'api_break_z_with_gravity', 'api_break_x_
arm', 'api_break_x_batter_in', 'arm_angle']

Testing second problematic range:

Debug Analysis:
Testing date range from 2024-06-26 to 2024-07-10
Using 3 days per chunk

Attempting to fetch: 2024-06-26 to 2024-06-29
This is a large query, it may take a moment to complete
100%|██████████| 4/4 [00:00<00:00, 22.08it/s]
```

```
Success! Retrieved 15999 rows
Columns present: ['pitch_type', 'game_date', 'release_speed', 'release_pos_
x', 'release_pos_z', 'player_name', 'batter', 'pitcher', 'events', 'descript
ion', 'spin_dir', 'spin_rate_deprecated', 'break_angle_deprecated', 'break_l
ength_deprecated', 'zone', 'des', 'game_type', 'stand', 'p_throws', 'home_te
am', 'away_team', 'type', 'hit_location', 'bb_type', 'balls', 'strikes', 'ga
me_year', 'pfx_x', 'pfx_z', 'plate_x', 'plate_z', 'on_3b', 'on_2b', 'on_1b',
'outs_when_up', 'inning', 'inning_topbot', 'hc_x', 'hc_y', 'tfs_deprecated',
'tfs_zulu_deprecated', 'umpire', 'sv_id', 'vx0', 'vy0', 'vz0', 'ax', 'ay',
'az', 'sz_top', 'sz_bot', 'hit_distance_sc', 'launch_speed', 'launch_angle',
'effective_speed', 'release_spin_rate', 'release_extension', 'game_pk', 'fie
lder_2', 'fielder_3', 'fielder_4', 'fielder_5', 'fielder_6', 'fielder_7', 'f
ielder_8', 'fielder_9', 'release_pos_y', 'estimated_ba_using_speedangle', 'e
stimated_woba_using_speedangle', 'woba_value', 'woba_denom', 'babip_value',
'iso_value', 'launch_speed_angle', 'at_bat_number', 'pitch_number', 'pitch_n
ame', 'home_score', 'away_score', 'bat_score', 'fld_score', 'post_away_scor
e', 'post_home_score', 'post_bat_score', 'post_fld_score', 'if_fielding_alig
nment', 'of_fielding_alignment', 'spin_axis', 'delta_home_win_exp', 'delta_r
un_exp', 'bat_speed', 'swing_length', 'estimated_slg_using_speedangle', 'del
ta_pitcher_run_exp', 'hyper_speed', 'home_score_diff', 'bat_score_diff', 'ho
me_win_exp', 'bat_win_exp', 'age_pit_legacy', 'age_bat_legacy', 'age_pit',
'age_bat', 'n_thruorder_pitcher', 'n_priorpa_thisgame_player_at_bat', 'pitch
er_days_since_prev_game', 'batter_days_since_prev_game', 'pitcher_days_until
_next_game', 'batter_days_until_next_game', 'api_break_z_with_gravity', 'api
_break_x_arm', 'api_break_x_batter_in', 'arm_angle']
Sample of data shape: (15999, 113)

Attempting to fetch: 2024-06-30 to 2024-07-03
This is a large query, it may take a moment to complete
100%|██████████| 4/4 [00:00<00:00, 24.60it/s]
```

```
Success! Retrieved 14034 rows
Columns present: ['pitch_type', 'game_date', 'release_speed', 'release_pos_
x', 'release_pos_z', 'player_name', 'batter', 'pitcher', 'events', 'descript
ion', 'spin_dir', 'spin_rate_deprecated', 'break_angle_deprecated', 'break_l
ength_deprecated', 'zone', 'des', 'game_type', 'stand', 'p_throws', 'home_te
am', 'away_team', 'type', 'hit_location', 'bb_type', 'balls', 'strikes', 'ga
me_year', 'pfx_x', 'pfx_z', 'plate_x', 'plate_z', 'on_3b', 'on_2b', 'on_1b',
'outs_when_up', 'inning', 'inning_topbot', 'hc_x', 'hc_y', 'tfs_deprecated',
'tfs_zulu_deprecated', 'umpire', 'sv_id', 'vx0', 'vy0', 'vz0', 'ax', 'ay',
'az', 'sz_top', 'sz_bot', 'hit_distance_sc', 'launch_speed', 'launch_angle',
'effective_speed', 'release_spin_rate', 'release_extension', 'game_pk', 'fie
lder_2', 'fielder_3', 'fielder_4', 'fielder_5', 'fielder_6', 'fielder_7', 'f
ielder_8', 'fielder_9', 'release_pos_y', 'estimated_ba_using_speedangle', 'e
stimated_woba_using_speedangle', 'woba_value', 'woba_denom', 'babip_value',
'iso_value', 'launch_speed_angle', 'at_bat_number', 'pitch_number', 'pitch_n
ame', 'home_score', 'away_score', 'bat_score', 'fld_score', 'post_away_scor
e', 'post_home_score', 'post_bat_score', 'post_fld_score', 'if_fielding_alig
nment', 'of_fielding_alignment', 'spin_axis', 'delta_home_win_exp', 'delta_r
un_exp', 'bat_speed', 'swing_length', 'estimated_slg_using_speedangle', 'del
ta_pitcher_run_exp', 'hyper_speed', 'home_score_diff', 'bat_score_diff', 'ho
me_win_exp', 'bat_win_exp', 'age_pit_legacy', 'age_bat_legacy', 'age_pit',
'age_bat', 'n_thruorder_pitcher', 'n_priorpa_thisgame_player_at_bat', 'pitch
er_days_since_prev_game', 'batter_days_since_prev_game', 'pitcher_days_until
_next_game', 'batter_days_until_next_game', 'api_break_z_with_gravity', 'api
_break_x_arm', 'api_break_x_batter_in', 'arm_angle']
Sample of data shape: (14034, 113)

Attempting to fetch: 2024-07-04 to 2024-07-07
This is a large query, it may take a moment to complete
100%|██████████| 4/4 [00:00<00:00, 24.79it/s]
```

```
Success! Retrieved 17883 rows
Columns present: ['pitch_type', 'game_date', 'release_speed', 'release_pos_
x', 'release_pos_z', 'player_name', 'batter', 'pitcher', 'events', 'descript
ion', 'spin_dir', 'spin_rate_deprecated', 'break_angle_deprecated', 'break_l
ength_deprecated', 'zone', 'des', 'game_type', 'stand', 'p_throws', 'home_te
am', 'away_team', 'type', 'hit_location', 'bb_type', 'balls', 'strikes', 'ga
me_year', 'pfx_x', 'pfx_z', 'plate_x', 'plate_z', 'on_3b', 'on_2b', 'on_1b',
'outs_when_up', 'inning', 'inning_topbot', 'hc_x', 'hc_y', 'tfs_deprecated',
'tfs_zulu_deprecated', 'umpire', 'sv_id', 'vx0', 'vy0', 'vz0', 'ax', 'ay',
'az', 'sz_top', 'sz_bot', 'hit_distance_sc', 'launch_speed', 'launch_angle',
'effective_speed', 'release_spin_rate', 'release_extension', 'game_pk', 'fie
lder_2', 'fielder_3', 'fielder_4', 'fielder_5', 'fielder_6', 'fielder_7', 'f
ielder_8', 'fielder_9', 'release_pos_y', 'estimated_ba_using_speedangle', 'e
stimated_woba_using_speedangle', 'woba_value', 'woba_denom', 'babip_value',
'iso_value', 'launch_speed_angle', 'at_bat_number', 'pitch_number', 'pitch_n
ame', 'home_score', 'away_score', 'bat_score', 'fld_score', 'post_away_scor
e', 'post_home_score', 'post_bat_score', 'post_fld_score', 'if_fielding_alig
nment', 'of_fielding_alignment', 'spin_axis', 'delta_home_win_exp', 'delta_r
un_exp', 'bat_speed', 'swing_length', 'estimated_slg_using_speedangle', 'del
ta_pitcher_run_exp', 'hyper_speed', 'home_score_diff', 'bat_score_diff', 'ho
me_win_exp', 'bat_win_exp', 'age_pit_legacy', 'age_bat_legacy', 'age_pit',
'age_bat', 'n_thruorder_pitcher', 'n_priorpa_thisgame_player_at_bat', 'pitch
er_days_since_prev_game', 'batter_days_since_prev_game', 'pitcher_days_until
_next_game', 'batter_days_until_next_game', 'api_break_z_with_gravity', 'api
_break_x_arm', 'api_break_x_batter_in', 'arm_angle']
Sample of data shape: (17883, 113)

Attempting to fetch: 2024-07-08 to 2024-07-10
This is a large query, it may take a moment to complete
100%|██████████| 3/3 [00:00<00:00, 27.98it/s]
Success! Retrieved 10622 rows
Columns present: ['pitch_type', 'game_date', 'release_speed', 'release_pos_
x', 'release_pos_z', 'player_name', 'batter', 'pitcher', 'events', 'descript
ion', 'spin_dir', 'spin_rate_deprecated', 'break_angle_deprecated', 'break_l
ength_deprecated', 'zone', 'des', 'game_type', 'stand', 'p_throws', 'home_te
am', 'away_team', 'type', 'hit_location', 'bb_type', 'balls', 'strikes', 'ga
me_year', 'pfx_x', 'pfx_z', 'plate_x', 'plate_z', 'on_3b', 'on_2b', 'on_1b',
'outs_when_up', 'inning', 'inning_topbot', 'hc_x', 'hc_y', 'tfs_deprecated',
'tfs_zulu_deprecated', 'umpire', 'sv_id', 'vx0', 'vy0', 'vz0', 'ax', 'ay',
'az', 'sz_top', 'sz_bot', 'hit_distance_sc', 'launch_speed', 'launch_angle',
'effective_speed', 'release_spin_rate', 'release_extension', 'game_pk', 'fie
lder_2', 'fielder_3', 'fielder_4', 'fielder_5', 'fielder_6', 'fielder_7', 'f
ielder_8', 'fielder_9', 'release_pos_y', 'estimated_ba_using_speedangle', 'e
stimated_woba_using_speedangle', 'woba_value', 'woba_denom', 'babip_value',
'iso_value', 'launch_speed_angle', 'at_bat_number', 'pitch_number', 'pitch_n
ame', 'home_score', 'away_score', 'bat_score', 'fld_score', 'post_away_scor
e', 'post_home_score', 'post_bat_score', 'post_fld_score', 'if_fielding_alig
nment', 'of_fielding_alignment', 'spin_axis', 'delta_home_win_exp', 'delta_r
un_exp', 'bat_speed', 'swing_length', 'estimated_slg_using_speedangle', 'del
ta_pitcher_run_exp', 'hyper_speed', 'home_score_diff', 'bat_score_diff', 'ho
me_win_exp', 'bat_win_exp', 'age_pit_legacy', 'age_bat_legacy', 'age_pit',
'age_bat', 'n_thruorder_pitcher', 'n_priorpa_thisgame_player_at_bat', 'pitch
er_days_since_prev_game', 'batter_days_since_prev_game', 'pitcher_days_until
_next_game', 'batter_days_until_next_game', 'api_break_z_with_gravity', 'api
_break_x_arm', 'api_break_x_batter_in', 'arm_angle']
Sample of data shape: (10622, 113)
```

```
Final combined dataset:
Total rows: 58538
Columns: ['pitch_type', 'game_date', 'release_speed', 'release_pos_x', 'rele
ase_pos_z', 'player_name', 'batter', 'pitcher', 'events', 'description', 'sp
in_dir', 'spin_rate_deprecated', 'break_angle_deprecated', 'break_length_dep
recated', 'zone', 'des', 'game_type', 'stand', 'p_throws', 'home_team', 'awa
y_team', 'type', 'hit_location', 'bb_type', 'balls', 'strikes', 'game_year',
'pfx_x', 'pfx_z', 'plate_x', 'plate_z', 'on_3b', 'on_2b', 'on_1b', 'outs_whe
n_up', 'inning', 'inning_topbot', 'hc_x', 'hc_y', 'tfs_deprecated', 'tfs_zul
u_deprecated', 'umpire', 'sv_id', 'vx0', 'vy0', 'vz0', 'ax', 'ay', 'az', 'sz
_top', 'sz_bot', 'hit_distance_sc', 'launch_speed', 'launch_angle', 'effecti
ve_speed', 'release_spin_rate', 'release_extension', 'game_pk', 'fielder_2',
'fielder_3', 'fielder_4', 'fielder_5', 'fielder_6', 'fielder_7', 'fielder_
8', 'fielder_9', 'release_pos_y', 'estimated_ba_using_speedangle', 'estimate
d_woba_using_speedangle', 'woba_value', 'woba_denom', 'babip_value', 'iso_va
lue', 'launch_speed_angle', 'at_bat_number', 'pitch_number', 'pitch_name',
'home_score', 'away_score', 'bat_score', 'fld_score', 'post_away_score', 'po
st_home_score', 'post_bat_score', 'post_fld_score', 'if_fielding_alignment',
'of_fielding_alignment', 'spin_axis', 'delta_home_win_exp', 'delta_run_exp',
'bat_speed', 'swing_length', 'estimated_slg_using_speedangle', 'delta_pitche
r_run_exp', 'hyper_speed', 'home_score_diff', 'bat_score_diff', 'home_win_ex
p', 'bat_win_exp', 'age_pit_legacy', 'age_bat_legacy', 'age_pit', 'age_bat',
'n_thruorder_pitcher', 'n_priorpa_thisgame_player_at_bat', 'pitcher_days_sin
ce_prev_game', 'batter_days_since_prev_game', 'pitcher_days_until_next_gam
e', 'batter_days_until_next_game', 'api_break_z_with_gravity', 'api_break_x_
arm', 'api_break_x_batter_in', 'arm_angle']

Testing known working range:

Debug Analysis:
Testing date range from 2024-05-01 to 2024-05-15
Using 3 days per chunk

Attempting to fetch: 2024-05-01 to 2024-05-04
This is a large query, it may take a moment to complete
100%|██████████| 4/4 [00:00<00:00, 25.22it/s]
```

```
Success! Retrieved 14787 rows
Columns present: ['pitch_type', 'game_date', 'release_speed', 'release_pos_
x', 'release_pos_z', 'player_name', 'batter', 'pitcher', 'events', 'descript
ion', 'spin_dir', 'spin_rate_deprecated', 'break_angle_deprecated', 'break_l
ength_deprecated', 'zone', 'des', 'game_type', 'stand', 'p_throws', 'home_te
am', 'away_team', 'type', 'hit_location', 'bb_type', 'balls', 'strikes', 'ga
me_year', 'pfx_x', 'pfx_z', 'plate_x', 'plate_z', 'on_3b', 'on_2b', 'on_1b',
'outs_when_up', 'inning', 'inning_topbot', 'hc_x', 'hc_y', 'tfs_deprecated',
'tfs_zulu_deprecated', 'umpire', 'sv_id', 'vx0', 'vy0', 'vz0', 'ax', 'ay',
'az', 'sz_top', 'sz_bot', 'hit_distance_sc', 'launch_speed', 'launch_angle',
'effective_speed', 'release_spin_rate', 'release_extension', 'game_pk', 'fie
lder_2', 'fielder_3', 'fielder_4', 'fielder_5', 'fielder_6', 'fielder_7', 'f
ielder_8', 'fielder_9', 'release_pos_y', 'estimated_ba_using_speedangle', 'e
stimated_woba_using_speedangle', 'woba_value', 'woba_denom', 'babip_value',
'iso_value', 'launch_speed_angle', 'at_bat_number', 'pitch_number', 'pitch_n
ame', 'home_score', 'away_score', 'bat_score', 'fld_score', 'post_away_scor
e', 'post_home_score', 'post_bat_score', 'post_fld_score', 'if_fielding_alig
nment', 'of_fielding_alignment', 'spin_axis', 'delta_home_win_exp', 'delta_r
un_exp', 'bat_speed', 'swing_length', 'estimated_slg_using_speedangle', 'del
ta_pitcher_run_exp', 'hyper_speed', 'home_score_diff', 'bat_score_diff', 'ho
me_win_exp', 'bat_win_exp', 'age_pit_legacy', 'age_bat_legacy', 'age_pit',
'age_bat', 'n_thruorder_pitcher', 'n_priorpa_thisgame_player_at_bat', 'pitch
er_days_since_prev_game', 'batter_days_since_prev_game', 'pitcher_days_until
_next_game', 'batter_days_until_next_game', 'api_break_z_with_gravity', 'api
_break_x_arm', 'api_break_x_batter_in', 'arm_angle']
Sample of data shape: (14787, 113)

Attempting to fetch: 2024-05-05 to 2024-05-08
This is a large query, it may take a moment to complete
100%|██████████| 4/4 [00:00<00:00, 20.68it/s]
```

```
Success! Retrieved 15832 rows
Columns present: ['pitch_type', 'game_date', 'release_speed', 'release_pos_
x', 'release_pos_z', 'player_name', 'batter', 'pitcher', 'events', 'descript
ion', 'spin_dir', 'spin_rate_deprecated', 'break_angle_deprecated', 'break_l
ength_deprecated', 'zone', 'des', 'game_type', 'stand', 'p_throws', 'home_te
am', 'away_team', 'type', 'hit_location', 'bb_type', 'balls', 'strikes', 'ga
me_year', 'pfx_x', 'pfx_z', 'plate_x', 'plate_z', 'on_3b', 'on_2b', 'on_1b',
'outs_when_up', 'inning', 'inning_topbot', 'hc_x', 'hc_y', 'tfs_deprecated',
'tfs_zulu_deprecated', 'umpire', 'sv_id', 'vx0', 'vy0', 'vz0', 'ax', 'ay',
'az', 'sz_top', 'sz_bot', 'hit_distance_sc', 'launch_speed', 'launch_angle',
'effective_speed', 'release_spin_rate', 'release_extension', 'game_pk', 'fie
lder_2', 'fielder_3', 'fielder_4', 'fielder_5', 'fielder_6', 'fielder_7', 'f
ielder_8', 'fielder_9', 'release_pos_y', 'estimated_ba_using_speedangle', 'e
stimated_woba_using_speedangle', 'woba_value', 'woba_denom', 'babip_value',
'iso_value', 'launch_speed_angle', 'at_bat_number', 'pitch_number', 'pitch_n
ame', 'home_score', 'away_score', 'bat_score', 'fld_score', 'post_away_scor
e', 'post_home_score', 'post_bat_score', 'post_fld_score', 'if_fielding_alig
nment', 'of_fielding_alignment', 'spin_axis', 'delta_home_win_exp', 'delta_r
un_exp', 'bat_speed', 'swing_length', 'estimated_slg_using_speedangle', 'del
ta_pitcher_run_exp', 'hyper_speed', 'home_score_diff', 'bat_score_diff', 'ho
me_win_exp', 'bat_win_exp', 'age_pit_legacy', 'age_bat_legacy', 'age_pit',
'age_bat', 'n_thruorder_pitcher', 'n_priorpa_thisgame_player_at_bat', 'pitch
er_days_since_prev_game', 'batter_days_since_prev_game', 'pitcher_days_until
_next_game', 'batter_days_until_next_game', 'api_break_z_with_gravity', 'api
_break_x_arm', 'api_break_x_batter_in', 'arm_angle']
Sample of data shape: (15832, 113)

Attempting to fetch: 2024-05-09 to 2024-05-12
This is a large query, it may take a moment to complete
100%|██████████| 4/4 [00:00<00:00, 24.71it/s]
```

```
Success! Retrieved 15036 rows
Columns present: ['pitch_type', 'game_date', 'release_speed', 'release_pos_
x', 'release_pos_z', 'player_name', 'batter', 'pitcher', 'events', 'descript
ion', 'spin_dir', 'spin_rate_deprecated', 'break_angle_deprecated', 'break_l
ength_deprecated', 'zone', 'des', 'game_type', 'stand', 'p_throws', 'home_te
am', 'away_team', 'type', 'hit_location', 'bb_type', 'balls', 'strikes', 'ga
me_year', 'pfx_x', 'pfx_z', 'plate_x', 'plate_z', 'on_3b', 'on_2b', 'on_1b',
'outs_when_up', 'inning', 'inning_topbot', 'hc_x', 'hc_y', 'tfs_deprecated',
'tfs_zulu_deprecated', 'umpire', 'sv_id', 'vx0', 'vy0', 'vz0', 'ax', 'ay',
'az', 'sz_top', 'sz_bot', 'hit_distance_sc', 'launch_speed', 'launch_angle',
'effective_speed', 'release_spin_rate', 'release_extension', 'game_pk', 'fie
lder_2', 'fielder_3', 'fielder_4', 'fielder_5', 'fielder_6', 'fielder_7', 'f
ielder_8', 'fielder_9', 'release_pos_y', 'estimated_ba_using_speedangle', 'e
stimated_woba_using_speedangle', 'woba_value', 'woba_denom', 'babip_value',
'iso_value', 'launch_speed_angle', 'at_bat_number', 'pitch_number', 'pitch_n
ame', 'home_score', 'away_score', 'bat_score', 'fld_score', 'post_away_scor
e', 'post_home_score', 'post_bat_score', 'post_fld_score', 'if_fielding_alig
nment', 'of_fielding_alignment', 'spin_axis', 'delta_home_win_exp', 'delta_r
un_exp', 'bat_speed', 'swing_length', 'estimated_slg_using_speedangle', 'del
ta_pitcher_run_exp', 'hyper_speed', 'home_score_diff', 'bat_score_diff', 'ho
me_win_exp', 'bat_win_exp', 'age_pit_legacy', 'age_bat_legacy', 'age_pit',
'age_bat', 'n_thruorder_pitcher', 'n_priorpa_thisgame_player_at_bat', 'pitch
er_days_since_prev_game', 'batter_days_since_prev_game', 'pitcher_days_until
_next_game', 'batter_days_until_next_game', 'api_break_z_with_gravity', 'api
_break_x_arm', 'api_break_x_batter_in', 'arm_angle']
Sample of data shape: (15036, 113)

Attempting to fetch: 2024-05-13 to 2024-05-15
This is a large query, it may take a moment to complete
100%|██████████| 3/3 [00:00<00:00, 23.94it/s]
```

```
Success! Retrieved 12541 rows
Columns present: ['pitch_type', 'game_date', 'release_speed', 'release_pos_
x', 'release_pos_z', 'player_name', 'batter', 'pitcher', 'events', 'descript
ion', 'spin_dir', 'spin_rate_deprecated', 'break_angle_deprecated', 'break_l
ength_deprecated', 'zone', 'des', 'game_type', 'stand', 'p_throws', 'home_te
am', 'away_team', 'type', 'hit_location', 'bb_type', 'balls', 'strikes', 'ga
me_year', 'pfx_x', 'pfx_z', 'plate_x', 'plate_z', 'on_3b', 'on_2b', 'on_1b',
'outs_when_up', 'inning', 'inning_topbot', 'hc_x', 'hc_y', 'tfs_deprecated',
'tfs_zulu_deprecated', 'umpire', 'sv_id', 'vx0', 'vy0', 'vz0', 'ax', 'ay',
'az', 'sz_top', 'sz_bot', 'hit_distance_sc', 'launch_speed', 'launch_angle',
'effective_speed', 'release_spin_rate', 'release_extension', 'game_pk', 'fie
lder_2', 'fielder_3', 'fielder_4', 'fielder_5', 'fielder_6', 'fielder_7', 'f
ielder_8', 'fielder_9', 'release_pos_y', 'estimated_ba_using_speedangle', 'e
stimated_woba_using_speedangle', 'woba_value', 'woba_denom', 'babip_value',
'iso_value', 'launch_speed_angle', 'at_bat_number', 'pitch_number', 'pitch_n
ame', 'home_score', 'away_score', 'bat_score', 'fld_score', 'post_away_scor
e', 'post_home_score', 'post_bat_score', 'post_fld_score', 'if_fielding_alig
nment', 'of_fielding_alignment', 'spin_axis', 'delta_home_win_exp', 'delta_r
un_exp', 'bat_speed', 'swing_length', 'estimated_slg_using_speedangle', 'del
ta_pitcher_run_exp', 'hyper_speed', 'home_score_diff', 'bat_score_diff', 'ho
me_win_exp', 'bat_win_exp', 'age_pit_legacy', 'age_bat_legacy', 'age_pit',
'age_bat', 'n_thruorder_pitcher', 'n_priorpa_thisgame_player_at_bat', 'pitch
er_days_since_prev_game', 'batter_days_since_prev_game', 'pitcher_days_until
_next_game', 'batter_days_until_next_game', 'api_break_z_with_gravity', 'api
_break_x_arm', 'api_break_x_batter_in', 'arm_angle']
Sample of data shape: (12541, 113)

Final combined dataset:
Total rows: 58196
Columns: ['pitch_type', 'game_date', 'release_speed', 'release_pos_x', 'rele
ase_pos_z', 'player_name', 'batter', 'pitcher', 'events', 'description', 'sp
in_dir', 'spin_rate_deprecated', 'break_angle_deprecated', 'break_length_dep
recated', 'zone', 'des', 'game_type', 'stand', 'p_throws', 'home_team', 'awa
y_team', 'type', 'hit_location', 'bb_type', 'balls', 'strikes', 'game_year',
'pfx_x', 'pfx_z', 'plate_x', 'plate_z', 'on_3b', 'on_2b', 'on_1b', 'outs_whe
n_up', 'inning', 'inning_topbot', 'hc_x', 'hc_y', 'tfs_deprecated', 'tfs_zul
u_deprecated', 'umpire', 'sv_id', 'vx0', 'vy0', 'vz0', 'ax', 'ay', 'az', 'sz
_top', 'sz_bot', 'hit_distance_sc', 'launch_speed', 'launch_angle', 'effecti
ve_speed', 'release_spin_rate', 'release_extension', 'game_pk', 'fielder_2',
'fielder_3', 'fielder_4', 'fielder_5', 'fielder_6', 'fielder_7', 'fielder_
8', 'fielder_9', 'release_pos_y', 'estimated_ba_using_speedangle', 'estimate
d_woba_using_speedangle', 'woba_value', 'woba_denom', 'babip_value', 'iso_va
lue', 'launch_speed_angle', 'at_bat_number', 'pitch_number', 'pitch_name',
'home_score', 'away_score', 'bat_score', 'fld_score', 'post_away_score', 'po
st_home_score', 'post_bat_score', 'post_fld_score', 'if_fielding_alignment',
'of_fielding_alignment', 'spin_axis', 'delta_home_win_exp', 'delta_run_exp',
'bat_speed', 'swing_length', 'estimated_slg_using_speedangle', 'delta_pitche
r_run_exp', 'hyper_speed', 'home_score_diff', 'bat_score_diff', 'home_win_ex
p', 'bat_win_exp', 'age_pit_legacy', 'age_bat_legacy', 'age_pit', 'age_bat',
'n_thruorder_pitcher', 'n_priorpa_thisgame_player_at_bat', 'pitcher_days_sin
ce_prev_game', 'batter_days_since_prev_game', 'pitcher_days_until_next_gam
e', 'batter_days_until_next_game', 'api_break_z_with_gravity', 'api_break_x_
arm', 'api_break_x_batter_in', 'arm_angle']
```