Here's an analysis of your code with potential viva questions and simplified answers:

---

### \*\*Code Analysis & Key Concepts\*\*

#### \*\*1. Data Cleaning\*\*

```python

# Fill missing math scores

df['math\_score'].fillna(df['math\_score'].mean())

# Drop missing reading scores

df.dropna(subset=['reading\_score'])

```

\*\*Why?\*\*

- Missing data can distort statistics and machine learning models

\*\*Q1: Why use different methods for math vs reading scores?\*\*

A1:

- Math: Filled with mean (preserves dataset size)

- Reading: Dropped rows (if few missing values)

\*Best Practice: Use domain knowledge to choose methods\*

\*\*Q2: When should you NOT use mean imputation?\*\*

A2: When data is skewed → mean ≠ typical value. Use median instead.

---

#### \*\*2. Skewness Analysis\*\*

```python

df['math\_score'].skew()

sns.histplot(df['math\_score'], kde=True)

```

\*\*Skewness Guide:\*\*

- 0 = Perfect normal

- (+) = Right skew (tail on right)

- (-) = Left skew

\*\*Q3: What if math\_score skewness is +0.8?\*\*

A3: Moderate right skew → most students scored below average, few high scorers.

\*\*Q4: Why visualize distributions?\*\*

A4: Histograms + KDE show:

- Data shape (normal/skewed)

- Outliers

- Need for transformations

---

#### \*\*3. Power Transformation\*\*

```python

df['math\_score power'] = df['math\_score']\*\*1.5

print(df['math\_score power'].skew())

```

\*\*Q5: Why use 1.5 power?\*\*

A5:

- Trial-and-error approach

- Common starting point for right-skewed data

- Higher exponents reduce skewness more

\*\*Q6: What's better - 1.5 power or log?\*\*

A6:

- \*\*Log\*\*: Stronger fix for heavy skew (e.g., +2)

- \*\*Power 1.5\*\*: Mild skew correction

\*Test both → choose what gives best results\*

\*\*Q7: What's Box-Cox transformation?\*\*

A7: Advanced method that automatically finds best exponent (λ).

\*Code: `from scipy.stats import boxcox`\*

---

### \*\*Viva Questions & Answers\*\*

#### \*\*Basic Concepts\*\*

\*\*Q1: What is skewness?\*\*

A1: Measure of data symmetry. Positive = right tail, Negative = left tail.

\*\*Q2: Why reduce skewness?\*\*

A2: Many statistical methods (like regression) assume normal data. Skewed data can:

- Reduce model accuracy

- Bias results

\*\*Q3: How to check if transformation worked?\*\*

A3:

1. Recalculate skewness (closer to 0 = better)

2. Replot histogram/KDE

#### \*\*Technical Questions\*\*

\*\*Q4: What does `kde=True` do?\*\*

A4: Adds Kernel Density Estimate line → smooth version of histogram.

\*\*Q5: Why not use `df.skew()` alone?\*\*

A5: Numbers can be misleading. Visualization shows:

- Multiple peaks

- Outliers

- Real-world meaning of skewness value

\*\*Q6: How would you handle left-skewed data?\*\*

A6: Try:

- Square/cube values (reverse power transform)

- Exponential transformation

#### \*\*Advanced Questions\*\*

\*\*Q7: What if transformations don't fix skewness?\*\*

A7: Use:

- Non-parametric methods (e.g., median instead of mean)

- Tree-based models (handle non-normal data well)

\*\*Q8: Can transformations create new issues?\*\*

A8: Yes!

- Over-transformation → reverse skew

- Lose interpretability (e.g., "math\_score^1.5" units)

\*\*Q9: When to prefer data binning over transformations?\*\*

A9: When:

- Preserving categories is important (e.g., letter grades)

- Data has extreme outliers

---

### \*\*Visualization Tips\*\*

Before Transformation | After Transformation

---|---

Right-skewed histogram | More bell-shaped curve

Long right tail | Symmetric distribution

---

### \*\*Key Takeaways\*\*

1. Always clean data first (handle missing values)

2. Check skewness → transform if needed

3. Visualize before/after changes

4. Power transformations are simple but effective

Would you like me to demonstrate any specific transformation technique in more detail? 😊