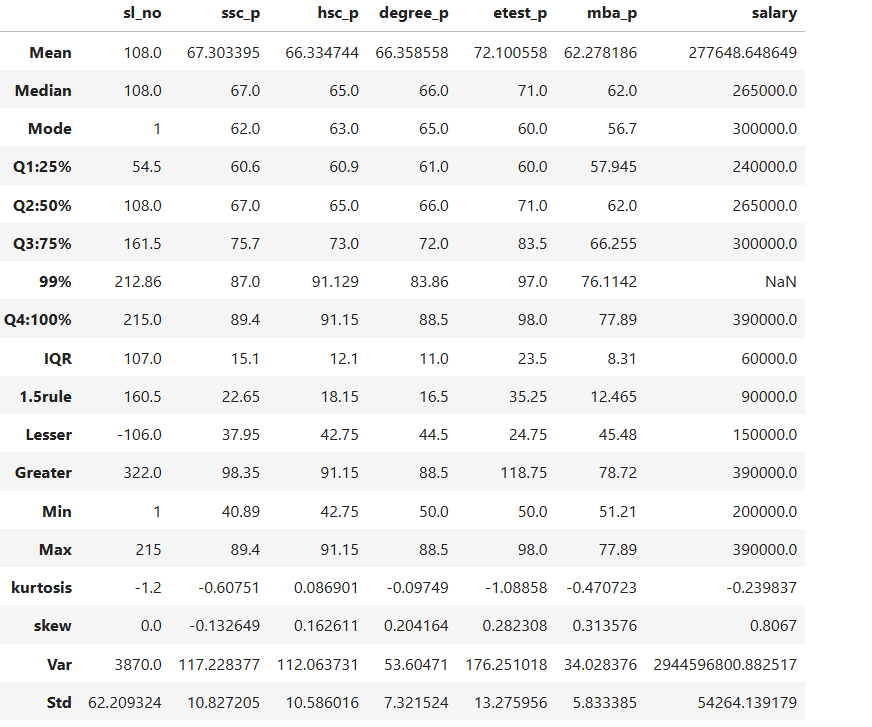
1. **Explain the below table.**

****

**Row Definitions**

|  |  |
| --- | --- |
| **Row** | **Description** |
| Mean | Average value of the column |
| Median | Middle value when sorted |
| Mode | Most frequently occurring value |
| Q1: 25% | First quartile (25% of values fall below this) |
| Q2: 50% | Median again (same as row above) |
| Q3: 75% | Third quartile (75% of values fall below this) |
| 99% | 99th percentile value |
| Q4: 100% | Maximum value (same as Max row) |
| IQR | Interquartile range = Q3 - Q1 |
| 1.5 rule | Used for outlier detection: 1.5 \* IQR |
| Lesser | Q1 - 1.5 \* IQR → lower outlier threshold |
| Greater | Q3 + 1.5 \* IQR → upper outlier threshold |
| Min | Minimum value |
| Max | Maximum value |
| Kurtosis | "Peakedness" of the distribution |
| Skew | Skewness (asymmetry) of the distribution |
| Var | Variance (measure of spread) |
| Std | Standard deviation (square root of variance) |

1. Outlier Detection:

* For etest\_p, values > 118.75 or < 24.75 might be outliers (using IQR method).
* For salary, the Lesser bound is ₹60,000 and Greater is ₹390,000 → values outside this range are considered outliers.

2. Distributions:

* Most features have skew close to 0, meaning nearly symmetric distributions.
* salary has positive skew (0.8067) → a few individuals have high salaries pulling the average up.
* etest\_p is also positively skewed (0.28), suggesting some very high-test scores.

3. Salary Insights:

* Mean salary: ₹277,648
* Median salary: ₹265,000 (less than mean → right-skewed)
* Std dev: ₹54,264 → fairly large spread
* IQR: ₹60,000
* Min salary: ₹200,000; Max: ₹390,000

4. Academic Scores:

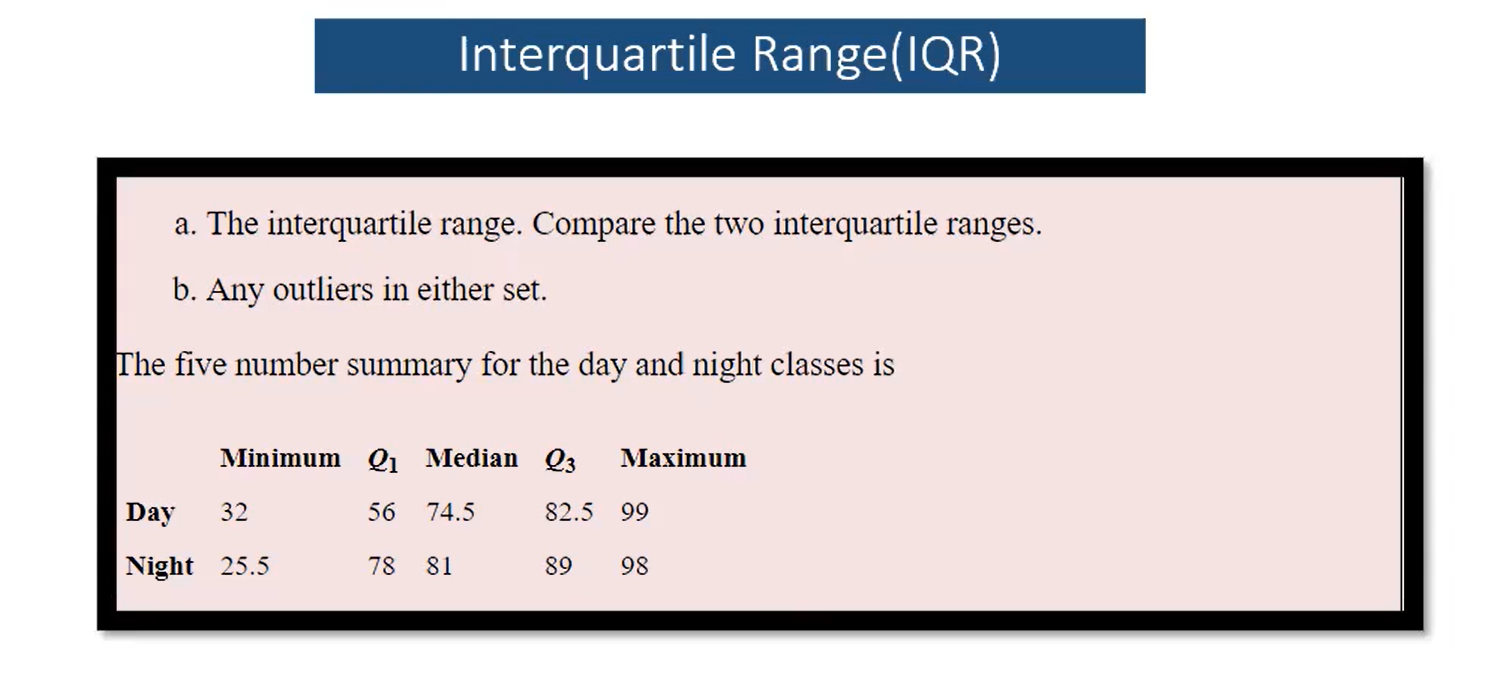
* Most academic scores (SSC, HSC, Degree, MBA) fall in the 60–70 range on average.
* Variance is highest in ssc\_p and hsc\_p, showing more diversity in those scores.

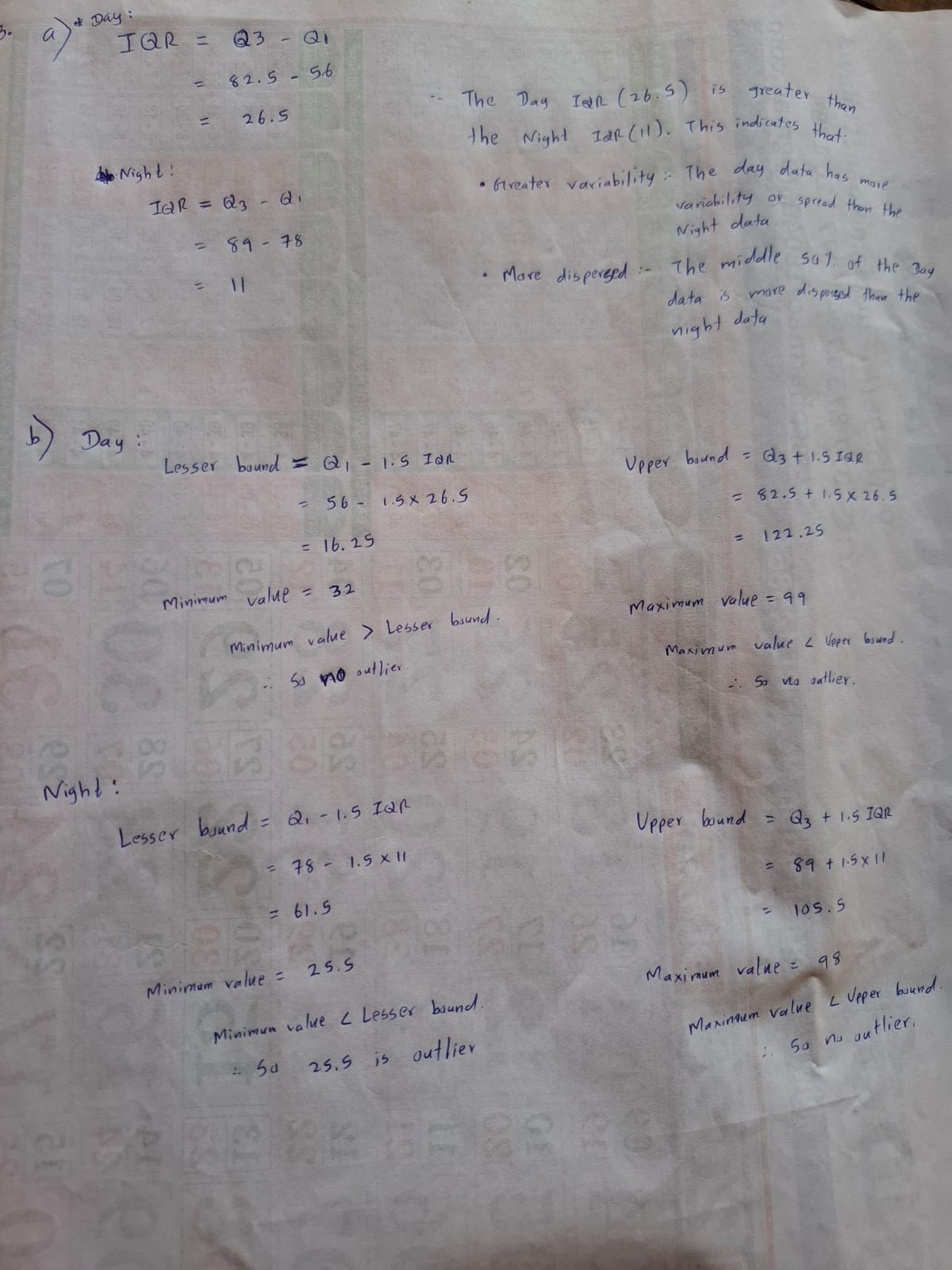
1. **Q1 - 1.5 \* IQR → lower outlier threshold and Q3 + 1.5 \* IQR → upper outlier threshold.**

**Why we are multiplying with 1.5?**

* The multiplier 1.5 comes from Tukey’s method, who introduced it in his work on exploratory data analysis. He empirically determined that multiplying the IQR by 1.5 is a good rule of thumb to catch mild outliers in most datasets without being too aggressive.
* Mathematically, if the data is normally distributed: The range from Q1 − 1.5 × IQR to Q3 + 1.5 × IQR includes **about 99.3%** of the data (for normal distribution), so anything outside that is **statistically rare**.

1. **Solve the below problem.**





1. **Explain the below function.**

****

This function used to Calculate and visualize the total probability (area under the curve) of a normal distribution between two specified values (start range and end range), using the Probability Density Function (PDF).

**from matplotlib import pyplot**

**from scipy.stats import norm**

**import seaborn as sns**

* Import libraries for plotting and statistical computation.

**ax = sns.distplot(dataset, kde=True, kde\_kws={'color':'blue'}, color='Green')**

**pyplot.axvline(startrange, color='Red')**

**pyplot.axvline(endrange, color='Red')**

* Histogram + KDE (Kernel Density Estimate) of the dataset.
* Vertical red lines at start range and end range to mark bounds on the plot.

**sample = dataset**

**sample\_mean = sample.mean()**

**sample\_std = sample.std()**

* Calculates mean and standard deviation of the dataset.
* These are parameters for the normal distribution assumed from the data.

**dist = norm(sample\_mean, sample\_std)**

* Creates a normal distribution object using scipy.stats.norm with the computed mean and std.

**values = [value for value in range(startrange, endrange)]**

**probabilities = [dist.pdf(value) for value in values]**

* Creates a list of values in the range.
* For each value, calculates the PDF value — this is the height of the curve at that point (not cumulative probability).

**prob = sum(probabilities)**

**print("The area between range({},{}):{}".format(startrange, endrange, sum(probabilities)))**

* Approximates the total area under the curve (probability) by summing PDF values.

**return prob**

* Returns the total approximated probability for the given range.