**EXCEL Functions**

1. **VLOOKUP:** Searches for a value in the first column of a table array and returns a value in the same row from another column.
2. **HLOOKUP:** Similar to VLOOKUP but searches for a value un the first row of a table array and returns a value in the same column.
3. **SUMIF/SUMIFS:** Adds the cells specified by a given condition or criteria.
4. **COUNTIF/COUNTIFS:** **Counts the number of cells specified by a given condition or criteria.**
5. **AVERAGEIF/AVERAGEIFS:** **Calculates the average of cells specified by a given condition or criteria.**
6. **INDEX/MATCH:** **Returns the value in a cell at the intersection of a particular row and column, based on matching a criteria.**
7. **Pivot Tables:** Summarizes, sorts and filters data in excel.
8. **IF/IFERROR:** **Executes a specific action based on a condition or returns a value if an error occurs.**
9. **CONCATENATE/CONCAT:** **Joins two or more strings together.**
10. **TEXT/DATEVALUE:** **Converts text to date values.**
11. **INDEX/MATCH Combination:** **Provides more flexibility than VLOOKUP and HLOOKUP for searching values in a table.**
12. **Conditional Formatting:** **Allows formatting cells based on certain conditions, making data visualization more intuitive.**
13. **Data Validation:** **Restricts the type of data that users can enter into a cell, ensuring data integrity.**
14. **Array Formulas:** Perform multiple calculations on one or more items in an array.
15. **Solver:** **An Excel add-in used for optimization and what-if analysis.**
16. **Pivot Charts:** **Visual representations of Pivot Table data.**
17. **Goal Seek:** **Finds the input needed to achieve a desired result in a formula.**
18. **Advanced Filter:** **Allows filtering data by multiple criteria and copying the filtered results to another location.**
19. **Text Functions (e.**g.: LEFT(), RIGHT(), MID(), etc.): **Extract or manipulate text data in cells.**
20. **Data Tables:** **Allows performing sensitivity analysis by calculating multiple versions of a formula with different input values.**

**SQL Functions**

1. **SELECT: Used to retrieve data from a database.**
2. **WHERE: Filters data based on specified conditions.**
3. **GROUP BY: Groups rows that have the same values into summary rows.**
4. **HAVING: Filters records returned by a GROUP BY clause.**
5. **ORDER BY: Sort the results set in ascending or descending order.**
6. **JOIN: Combines rows from two or more tables based on a related column.**
7. **DISTINCT: Returns unique values in a specified column or expression.**
8. **COUNT: Returns the number of rows in a specified table or view.**
9. **SUM(): Calculates the sum of set of values.**
10. **AVG(): Calculates the average of a set of values.**
11. **CASE Statement: Allows conditional logic within SQL queries.**
12. **UNION: Combines the result sets of two or more SELECT statements.**
13. **CTE (Common Table Expression): Temporary result sets that can be referenced within a SELECT, INSERT, UPDATE or DELETE statement.**
14. **Window Functions: Perform calculations across a set of rows that are related to the current row.**
15. **Store Procedures: Precompiled SQL code that can be executed by calling the procedure name.**
16. **INDEX: Improves the speed of data retrieval operations on a database table at the cost of additional space and decreased performance for insert, update and delete operations.**
17. **TRIGGER: A database object that automatically performs an action in response to certain on a particular table or view.**
18. **EXISTS: Texts for the existence of any rows in a subquery and returns true if the subquery returns one or more rows.**
19. **ROLLUP: Generates subtotal values for the data, based on one or more columns.**
20. **EXPLAIN: Analyses the execution plan of a SELECT statement to help optimize the query performance.**

**PANDAS Functions**

1. **read\_csv(): Reads a CSV file into a DataFrame.**
2. **to\_csv(): Writes DataFrame to a CSV file.**
3. **pd.DatFrame.to\_excel(): Writes DataFrame to an Excel file.**
4. **head(): Returns the first n rows of a DataFrame.**
5. **tail(): Returns the last n rows of a DataFrame.**
6. **info(): Provides a concise summary of a DataFrame, including data types and not null values.**
7. **describe(): Generates descriptive statistics of the DaraFrame.**
8. **loc[]: Accesses a group of rows and columns by label(s) or a boolean array.**
9. **iloc[]: Accesses a group of rows and columns by integer positions(s).**
10. **groupby(): Combines two DataFrames using a mapper or by a Series of columns.**
11. **merge(): Combines two DataFrames by a common column.**
12. **pivot\_table(): Creates a spreadsheet-style pivot table as a DataFrame.**
13. **pd.DataFrame.groupby(): Groups DatFrame using a mapper or by a Series of columns.**
14. **pd.merge(): Combines DataFrame objects by performing a database-style join operation.**
15. **pd.concat(): Concatenates pandas objects along a particular axis with optional set logic the other axes.**
16. **pd.pivot\_table(): Creates a spreadsheet-style pivot table as a DataFrame.**
17. **pd.melt(): Unpivots DataFrame from wide to long format.**
18. **pd.cut(): Bin values into discrete intervals.**
19. **pd.qcut(): Quantile-based discretization function.**
20. **pd.DataFrame.apply(): Applies a function along an axis of the DataFrame**
21. **pd.DataFrame.drop\_duplicates(): Removes duplicate rows from the DataFrame.**

NUMPY Function

1. **np.array()**: Creates a NumPy array from a list or tuple, enabling efficient numerical operations.
2. **np.zeros()**: Creates an array filled with zeros, useful for initializing datasets.
3. **np.ones()**: Creates an array filled with ones, often used for creating masks or as initial values.
4. **np.arange()**: Generates an array with evenly spaced values within a specified range.
5. **np.linspace()**: Creates an array of evenly spaced values over a specified interval, useful for generating datasets.
6. **np.reshape()**: Changes the shape of an array without changing its data, facilitating data manipulation.
7. **np.flatten()**: Flattens a multi-dimensional array into a one-dimensional array.
8. **np.concatenate()**: Joins two or more arrays along a specified axis.
9. **np.vstack()**: Vertically stacks arrays (row-wise).
10. **np.hstack()**: Horizontally stacks arrays (column-wise).
11. **np.mean()**: Computes the average of array elements along a specified axis.
12. **np.median()**: Calculates the median of array elements along a specified axis.
13. **np.std()**: Computes the standard deviation of array elements, providing insights into variability.
14. **np.var()**: Calculates the variance of array elements.
15. **np.sum()**: Computes the sum of array elements, useful for aggregating data.
16. **np.min()**: Finds the minimum value in an array.
17. **np.max()**: Finds the maximum value in an array.
18. **np.unique()**: Identifies unique values in an array, useful for categorical data analysis.
19. **np.corrcoef()**: Computes the correlation coefficients between two or more arrays, helpful in analyzing relationships.
20. **np.dot()**: Performs dot product operations, commonly used in linear algebra and matrix multiplication.
21. **np.where()**: Returns indices of elements that satisfy a specified condition, useful for conditional selections.
22. **np.isin()**: Checks whether elements of one array are present in another, facilitating filtering and subsetting.
23. **np.random.rand()**: Generates an array of random numbers uniformly distributed over [0, 1), useful for simulations.
24. **np.random.randn()**: Generates an array of random numbers from the standard normal distribution.
25. **np.random.seed()**: Sets the random seed for reproducibility of random number generation.
26. **np.concatenate()**: Combines two or more arrays along an existing axis.
27. **np.clip()**: Limits the values in an array to a specified range, useful for handling outliers.
28. **np.split()**: Splits an array into multiple sub-arrays along a specified axis.
29. **np.transpose()**: Transposes an array, flipping it over its diagonal.
30. **np.apply\_along\_axis()**: Applies a function to 1-D slices of an array along the specified axis.

**MATPLOTLIB Functions**

1. **plt.plot(): Creates a line plot.**
2. **plt.**scatter(): Creates a scatter plot.
3. **plt.bar(): Creates a bar plot.**
4. **plt.hist(): Creates a histogram.**
5. **Plt.**boxplot(): Creates a boxplot.
6. **Plt.**xlable(): Sets the lable for the x-axis.
7. **Plt.**ylable(): Sets the lable for the y-axis.
8. **Plt.title(): Sets the title of the plot.**
9. **Plt.legends(): Add a legend to the plot.**
10. **Plt.**show(): Display the plot.
11. **Plt.**savefig(): Save the plot to a file.
12. **Plt.**subplots(): Create a figure and set of subplots.
13. **Plt.**figure(): Creates a new figure.
14. **Plt.**xticks(): Sets the tick labels on the x-axis.
15. **Plt.**yticks(): Sets the tick labels on the y-axis.
16. **Plt.**grid(): Adds grid lines to the plot.
17. **Plt.**xlim(): Sets the limits for the x-axis.
18. **Plt.**ylim(): Sets the limits for the y-axis.
19. **Plt.**annotate(): Adds annotations to the plot.
20. **Plt.**subplots\_adjust(): Adjusts the spacing between subplots.

**SEABORN Functions**

1. **sns.lineplot(): Creates a line plot with confidence intervals.**
2. **sns.scatterplot(): Creates a scatter plot with optional size and hue mappings.**
3. **sns.barplot(): Creates a bar plot with mean estimates and confidence intervals.**
4. **sns.histplot(): Creates a histogram or kernel density estimate plot.**
5. **sns.boxplot(): Creates a boxplot to show distributions and outliers.**
6. **sns.violinplot(): Combines boxplot and density information.**
7. **sns.heatmap(): Creates a heatmap from a matrix of values.**
8. **sns.pairplot(): Plots pairwise relationships in a dataset.**
9. **sns.catplot(): Creates a categorical plot that can be customized.**
10. **sns.FacetGrid(): Creates a grid of subplots based on different subsets of the data.**
11. **sns.despine(): Removes the top and right spines from a plot.**
12. **sns.set\_style(): Sets the aesthetic style of the plots (e.g., "darkgrid", "white").**
13. **sns.set\_palette(): Sets the color palette for the plots.**
14. **sns.set\_context(): Adjusts the scaling of plot elements for different contexts.**
15. **sns.regplot(): Creates a scatter plot with a regression line.**
16. **sns.kdeplot(): Creates a kernel density estimate plot.**
17. **sns.countplot(): Creates a bar plot for categorical data counts.**
18. **sns.jointplot(): Creates a multi-plot that shows the relationship between two variables along with their distributions.**
19. **sns.boxenplot(): Similar to boxplot but useful for large datasets with more quantiles.**
20. **sns.lmplot(): Combines scatter plots with regression lines for more complex data.**