THE ART OF

DATA SCIENCE

Driving Growth and Success in the Digital Age

PARVEZ SHAH SHAIK

1. Difference between Data Mining and Data Profiling?

Data Mining	Data Profiling
Data mining is the process of discovering patterns, correlations, trends, and useful information from large sets of data. The aim is to extract knowledge from a dataset that was previously unknown and to predict future trends and behaviors.	Data profiling is the process of examining the data available in an existing database and collecting statistics and information about that data. The goal is to understand the quality, structure, and distribution of data.
It involves the use of sophisticated data analysis tools to discover patterns and relationships in data that may be used for decision making. Techniques include clustering, classification, regression, association rules, and anomaly detection.	It involves analyzing the data to identify inconsistencies, anomalies, and patterns in the data structure. This includes checking for data accuracy, completeness, uniqueness, and consistency.
Data mining is used in various fields such as marketing, finance, healthcare, and bioinformatics for applications like market basket analysis, fraud detection, customer segmentation, and gene expression analysis.	Data profiling is often used in the initial stages of data analysis to assess the quality of data, or in data integration and data migration projects to ensure the compatibility of data sources.
It is a complex process that requires a deep understanding of algorithms and statistical methods to interpret the data patterns correctly.	While it can be complex, data profiling is generally less technically intensive than data mining. It focuses more on the quality and structure of the data rather than discovering hidden patterns.

The outcome of data mining is the generation of new insights and predictive models that can be used to support decision-making and strategy development.	The outcome of data profiling is a detailed report on the condition of the data, including any issues that need to be addressed before further analysis or use of the data. This helps in improving data quality and ensuring that subsequent analyses are based on reliable data.

2. Define Data Wrangling in Data Analytics?

Definition:

- Data wrangling, also known as data munging, is a crucial step in the data analytics process that involves cleaning, structuring, and enriching raw data into a desired format for better decision making in less time. This process can map out large amounts of data, extracted from various sources into more useful format.
- Techniques such as *merging, grouping, concatenating, joining and sorting* are used to analyze the data.
- The goal of data wrangling is to ensure that the data feeding into an analytics or machine learning model is accurate, consistent, and ready for analysis, which in turn leads to more reliable outcomes and insights.

Explanation:

Data wrangling, sometimes called data munging, is a very important step when working with data. Imagine you have a messy room full of different things scattered everywhere. Data wrangling is like tidying up that room so you can find and use what you need more easily. Here's how it works in simple terms:

Cleaning the Data: This is like picking up trash and things that don't belong in the room. In data terms, it means fixing errors, filling in missing information, and removing useless data.
Organizing the Data: Just like sorting your things into where they belong, in data wrangling, you arrange the data in a way that makes sense. This could mean grouping similar things together or putting data in a certain order.
Making the Data Better: Sometimes, you might add shelves or labels in your room to make things even easier to find. With data, this means adding extra information that helps make the data more useful or easier to understand.
Using Techniques: Techniques like merging (combining two sets of data into one), grouping (organizing data based on common features), concatenating (linking data together in a sequence), joining (connecting related data from different sources), and sorting (arranging data in a certain order) are tools to help with this process.
The Goal: The aim of all this effort is to make sure the data is clean, well-organized, and ready to be used in data analysis or machine learning. Just like a tidy room helps you think clearer and make better decisions, well-prepared data leads to more accurate and reliable results from your analysis.

3. What are the steps involved in involved in Analytics Project?

Analytics projects, regardless of their specific domain or objectives, typically follow a structured process to transform raw data into actionable insights. Here are the various steps involved in any analytics project:

1. Problem Definition:
   - Understand and define the business problem or question that needs to be answered. This step involves discussions with stakeholders to clearly outline the project's goals and success criteria.

2. Data Collection:
- Gather the necessary data from various sources. This could involve extracting data from internal databases, external datasets, APIs, or even manual data collection methods.

3. Data Cleaning and Preparation (Data Wrangling):
- Clean and preprocess the data to ensure it is accurate, consistent, and usable for analysis. This involves handling missing values, removing duplicates, correcting errors, and possibly normalizing data formats.

4. Data Exploration and Analysis:
- Perform exploratory data analysis (EDA) to understand the data's underlying patterns, relationships, and anomalies. This step often involves statistical analysis, data visualization, and the use of descriptive analytics techniques.

5. Feature Engineering and Selection:
- Create new features (variables) from the existing data that could improve the model's performance and select the most relevant features for the analysis. This step is crucial for machine learning projects.

6. Modeling:
- Develop and train statistical or machine learning models using the prepared dataset. This could involve choosing the right algorithms, tuning parameters, and using techniques like cross-validation to avoid overfitting.

7. Evaluation:
- Assess the model's performance using appropriate metrics. The choice of metrics depends on the type of problem (e.g., classification, regression) and the business objectives.

8. Deployment:
- Implement the model in a production environment where it can provide insights or predictions based on new data. This step may involve integrating the model into existing systems or processes.

9. Monitoring and Maintenance:
- Continuously monitor the model's performance over time to ensure it remains accurate and relevant. This may involve periodic retraining of the model with new data or adjusting the model to reflect changes in the underlying data patterns.

10. Communication and Reporting:
- Communicate the findings, insights, and recommendations to stakeholders in a clear and understandable manner. This often involves creating reports, dashboards, or presentations that summarize the project's outcomes and business impact.

Each of these steps is crucial for the success of an analytics project, ensuring that the final insights are reliable, relevant, and actionable for decision-making purposes.

4. What are the common problems that data analysts encounter during analysis?

Data analysts often run into several challenges while working with data, which can affect their work's outcome and efficiency. Here are some of the common issues they face, explained in simple terms:

1. Bad Quality Data: Sometimes, the data has mistakes, missing pieces, or duplicate information. Fixing these issues is crucial but can be tough.

2. Big and Complicated Data: Dealing with a lot of data or data that's hard to understand can be overwhelming. Analysts need special tools and skills to manage it.

3. Mixing Data from Different Places: Combining data from various sources can be tricky, especially when the formats or quality vary.

4. Keeping Data Safe and Private: It's important to follow rules that protect people's personal information and keep the data safe, which can be challenging.

5. Unclear Goals: If analysts aren't sure what the goal of their analysis is, it can lead to irrelevant findings.

6. Choosing the Right Tools: There are many tools and methods available for data analysis. Picking the most suitable ones is crucial but can be difficult.

7. Understanding the Data Correctly: Interpreting the data accurately is key. This requires not just technical skills but also knowledge about the subject and careful thinking.

8. Data Being Kept in Isolation: When data is kept separate in different parts of an organization, it can be hard to get a full picture.

9. Limited Time: Often, there's not a lot of time to do the analysis, which can make it hard to do a thorough job.

10. Staying Up-to-date: The tools and methods for data analysis change quickly. Keeping up with these changes requires constant learning.

11. Sharing Findings Clearly: It can be challenging to explain complex data findings in a way that everyone can understand, especially to people who don't work with data.

Dealing with these challenges usually means using a mix of skills, careful planning, and clear communication with others involved in the project to make sure the analysis is useful and accurate.

5. Which are the technical tools that you have used for analysis and presentation purposes?

As a data analyst you are expected to know the tools mentioned below for analysis and presentation purposes.
Some of the popular tools you should know are:

1. MS SQL Server, MySQL
For working with data stored in relational databases
2. MS Excel, Tableau
For creating reports and dashboards
3. Python, R, SPSS
For statistical analysis, data modeling, and exploratory data analysis
4. MS PowerPoint
For presentation, displaying the final results and important conclusions

6. What are the best methods for data cleaning?

1. Make a Plan for Cleaning Data: First, figure out where mistakes usually happen. Keep talking to your team to solve problems together.
2. Remove Copies Before Starting: Find and take out any repeated data. This makes analyzing the data easier and more effective.
3. Check for Mistakes Carefully: Make sure data matches up correctly, keep data types the same, and follow the rules for what data should look like. This keeps your data accurate.
4. Fix Data Right When It Comes In: Make your data consistent from the start. This helps avoid mistakes and keeps everything uniform.

7. What is the importance of EDA?

Exploratory Data Analysis (EDA) is like getting to know your data by exploring it. Here's why it's important:

- Understanding Your Data: Exploratory data analysis (EDA) helps to understand the data better. EDA is like detective work for data. It helps you look closely at what you have before you start any heavy analysis or use machine learning.
- Building Trust in Your Data: It helps you obtain confidence in your data to a point where you're ready to engage a machine learning algorithm. It's like making sure the foundation of a house is strong before you build on it. EDA gives you confidence that your data is good to work with.
- Choosing What's Important: Think of EDA as packing for a trip. Just like you pick what to pack based on where you're going, EDA helps you decide which data points (features) are important for your analysis later on.
- Finding Hidden Clues: EDA can reveal trends and patterns you might not see at first glance, similar to finding clues in a treasure hunt.

Simply put, EDA is a crucial step that prepares and guides you on how to best use your data for deeper analysis or building models.

8. Explain descriptive, predictive, and prescriptive analytics.

Descriptive, predictive, and prescriptive analytics are three key types of data analytics, each serving a unique purpose in understanding and utilizing data to make decisions. Here's a simple way to explain them:

Descriptive Analytics

- What it does: Looks at past data to understand what happened and why.
- Example: Reviewing last year's sales data to see which months had the highest sales.

Predictive Analytics
- What it does: Uses past data to make predictions about the future.
- Example: Analyzing patterns in sales data to predict which months in the future will have higher sales.

Prescriptive Analytics
- What it does: Suggests actions you can take to affect desired outcomes in the future.
- Example: Recommending specific marketing strategies to increase sales in months predicted to have lower sales.

In summary:
- Descriptive analytics tells you about the past and present.
- Predictive analytics forecasts future trends.
- Prescriptive analytics advises on possible courses of action to achieve goals or solve problems.

9. What are the different types of sampling techniques used by data analysts?

Data analysts use various sampling techniques to draw representative samples from larger populations. Some common sampling techniques include:

1. Simple Random Sampling: Every member of the population has an equal chance of being selected. This can be done with or without replacement.
   Example: Imagine you have a bag of candies, and you pick candies from it blindly without looking. Every candy has the same chance of being picked.

2. Stratified Sampling: The population is divided into distinct subgroups or strata, and then samples are randomly selected from each stratum proportionally to the population size within the stratum.
   Example: Suppose you have different types of candies in your bag, like chocolates, gummies, and lollipops. With stratified sampling, you'd separate them into groups first and then randomly pick some from each group. This ensures you get a good mix of all candy types.

3. Systematic Sampling: Selecting every nth member from the population after randomly selecting a starting point. For example, if every 10th member is chosen and there are 1000 members, the sample would consist of 100 individuals.
   Example: Let's say you have a line of people waiting for candy, and you decide to give candy to every 5th person in line. You start randomly with the first person, and then every 5th person after that gets candy.

4. Cluster Sampling: The population is divided into clusters, and then a random sample of clusters is selected. All members of the selected clusters are included in the sample.
   Example: Imagine you have several bags of candies, and you randomly choose a few bags. Then, you pick candies from the bags you chose. This way, you're not picking candies from every single bag, just from the ones you randomly selected.

5. Convenience Sampling: Sampling individuals who are conveniently available or easy to reach. This method may introduce bias because it does not provide equal opportunity for all members of the population to be included.
   Example: If you're giving out candy in a park, you might just hand them out to the people nearby. This method is easy, but it might not represent everyone in the park since you're only picking those who are close by.

6. Snowball Sampling: Sampling starts with a small group of individuals, then additional participants are recruited through referrals from initial participants. This method is often used in studies where the population of interest is hard to reach.
   Example: Let's say you give candy to a few kids in the park, and then ask them to bring their friends over for more candy. Then, those friends bring more friends, and so on. This method helps you reach more people through referrals.

7. Quota Sampling: The population is divided into subgroups based on certain characteristics, and then participants are selected non-randomly according to pre-defined quotas.
   Example: Imagine you want to give candy to kids of different ages. You decide you want 50% of the candy to go to kids under 10 and 50% to kids over 10. So, you keep giving out candy until you meet those quotas.

8. Purposive Sampling: Also known as judgmental or selective sampling, this method involves selecting individuals based on the researcher's judgment about which participants will best serve the research objectives.
   Example: Suppose you're specifically looking for feedback on a new chocolate candy. You might only give samples to people who you think are likely to enjoy chocolate, like chocolate lovers or kids who always ask for chocolate

Each method has its own way of selecting people or items for your sample, and they're useful in different situations depending on what you're studying and who you're trying to reach.

10. Describe univariate, bivariate, and multivariate analysis.

1. Univariate Analysis:
    - **What is it?** Univariate analysis focuses on analyzing one variable at a time. It's like zooming in on just one aspect of your data.
    - **Example**: If you're analyzing the heights of students in a class, univariate analysis would involve looking at the distribution of heights, calculating measures like the mean or median height, and visualizing it with histograms or box plots.
    - **Purpose**: It helps you understand one specific thing without getting distracted by other factors.
2. Bivariate Analysis:
    - **What is it?** Bivariate analysis involves analyzing the relationship between two variables at the same time. You're looking at how one variable behaves in relation to another.
    - **Example**: Continuing with the height example, bivariate analysis could involve examining the relationship between height and weight. You might scatter plot height against weight to see if there's a pattern.
    - **Purpose**: It helps you see if there's any connection or pattern between two factors
3. Multivariate Analysis:
    - **What is it?** Multivariate analysis involves analyzing more than two variables simultaneously. It's like looking at a complex web of relationships between multiple variables.
    - **Example**: Extending the example further, multivariate analysis might involve considering height, weight, age, and gender all together to see how they collectively influence each other.
    - **Purpose**: t helps you understand how all these different factors work together and influence each other. It's like seeing the big picture with lots of pieces Multivariate analysis can be performed using Multiple regression, Factor

analysis, Classification & regression trees, Cluster analysis, Principal component analysis, Dual-axis charts.

In summary, univariate analysis looks at one variable, bivariate analysis examines the relationship between two variables, and multivariate analysis considers the interactions among multiple variables. Each type of analysis provides valuable insights into different aspects of your data, helping you draw meaningful conclusions and make informed decisions.

11. What are your strengths and weaknesses as a data analyst?

The answer to this question may vary from a case-to-case basis. However, some general strengths of a data analyst may include strong analytical skills, attention to detail, proficiency in data manipulation and visualization, and the ability to derive insights from complex datasets. Weaknesses could include limited domain knowledge, lack of experience with certain data analysis tools or techniques, or challenges in effectively communicating technical findings to non-technical stakeholders.

12. What are the ethical considerations of data analysis?

Some of the most the ethical considerations of data analysis includes:

1. Privacy: We need to make sure people's personal information is kept safe and only used for the reasons they agreed to.
2. Transparency: We should be clear about how we collect and use data so people understand what's happening with their information.
3. Bias and Fairness: We must be fair when analyzing data and try to avoid unfair treatment or stereotypes based on things like race or gender.
4. Data Quality: We need to make sure the data we use is accurate and reliable, without any mistakes or errors.

5. Informed Consent: People should agree to let us use their data and understand what we're going to do with it before we start analyzing it.
6. Data Security: We must protect data from being accessed or used by people who shouldn't have it, keeping it safe from hackers or unauthorized use.
7. Responsibility: We need to think about how our data analysis might affect people and communities and use it in a way that helps without causing harm.
8. Professional Integrity: We need to be honest and accountable in our work, avoiding conflicts of interest and following the rules of our profession.

By considering these things, we can make sure our data analysis is done in a fair, safe, and responsible way.

13. What are some common data visualization tools you have used?

You should name the tools you have used personally, however here's a list of the commonly used data visualization tools in the industry:

- Tableau
- Microsoft Power BI
- QlikView
- Google Data Studio
- Plotly
- Matplotlib (Python library)
- Excel (with built-in charting capabilities)
- SAP Lumira
- IBM Cognos Analytics

Data Analyst Interview Questions On Statistics

14. How can you handle missing values in a dataset?

Handling missing values in a dataset is essential to ensure the accuracy and reliability of data analysis. Here are some common techniques to handle missing values:

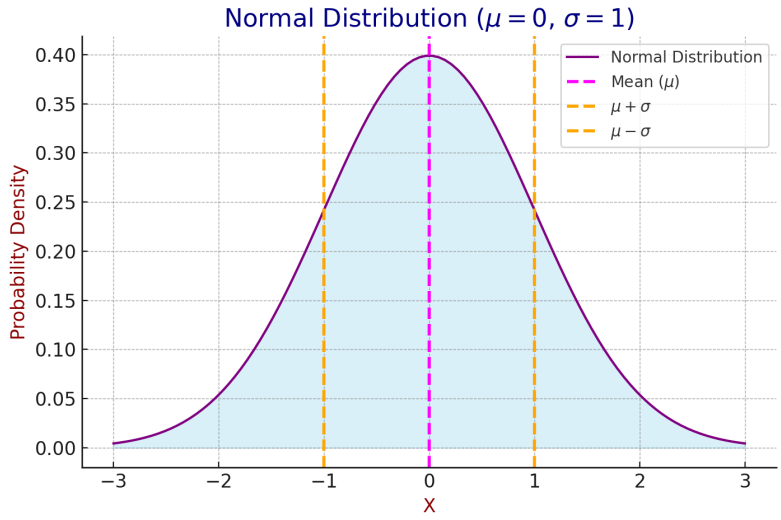
1. **Removal**: Delete rows or columns with missing values. This is suitable when missing values are few and randomly distributed, ensuring minimal impact on the analysis.
2. **Imputation**: Fill in missing values with a calculated estimate. Common imputation methods include:
    - **Mean/Median/Mode imputation**: Replace missing values with the mean, median, or mode of the respective column.
    - **Regression imputation**: Predict missing values based on the relationship with other variables using regression models.
    - **K-nearest neighbors (KNN) imputation**: Use the values of nearest neighbors to impute missing values.
    - **Multiple imputation**: Generate multiple imputed datasets to account for uncertainty in imputed values.
3. **Predictive Models**: Use machine learning algorithms to predict missing values based on other variables in the dataset.
4. **Domain-specific Imputation**: Utilize domain knowledge to impute missing values with plausible estimates.
5. **Flagging**: Create an additional indicator variable to flag missing values, allowing models to differentiate between observed and missing data.
6. **Interpolation**: Estimate missing values based on trends or patterns observed in the data.
7. **Manual Entry**: Manually fill in missing values based on external sources or expert judgment, particularly for critical data.
8. **Special Handling**: Depending on the context, some missing values may have specific meanings (e.g., "N/A" for non-applicable). It's important to recognize and appropriately handle such cases.

The choice of method depends on factors such as the nature of the missing data, the underlying mechanism causing missingness, the analysis goals, and the potential impact on results. It's essential to document and justify the chosen approach to ensure transparency and reproducibility in data analysis.

15. Explain the term Normal Distribution.

The normal distribution, also known as the Gaussian distribution or bell curve, is a statistical concept that describes how data is spread out or distributed. Here's a simple explanation:
Imagine you have a bunch of data points, like test scores from a class.
The normal distribution tells you how these scores are likely to be distributed around the average score.
In a normal distribution:

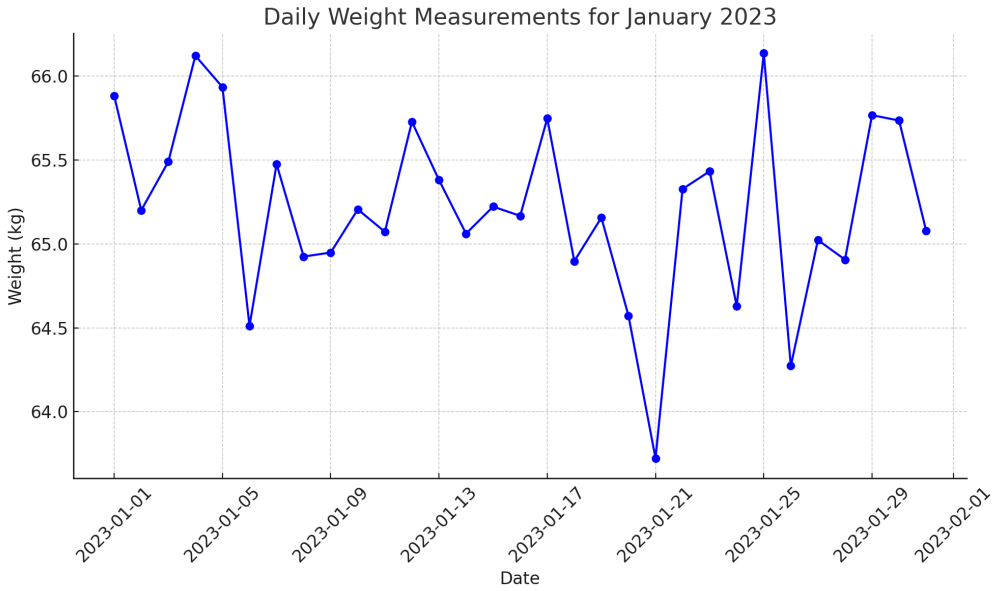
- The data clusters around the middle, or the average, which is represented by the highest point of the curve.
- As you move away from the average towards higher or lower scores, the number of data points decreases gradually.
- The curve is symmetrical, meaning the same number of data points fall on both sides of the average.
- The shape of the curve resembles a bell, hence the nickname "bell curve."



16. What is Time Series analysis?

A time series is a sequence of data points or observations collected or recorded at successive and equally spaced intervals of time. These data points are typically ordered chronologically, with each observation representing a measurement, value, or characteristic of a variable of interest at a specific point in time. Time series data is often used to analyze and understand how variables change over time, identify patterns, trends, and seasonality, and make predictions about future values based on past observations. Examples of time series data include stock prices, weather measurements, economic indicators, and sales figures collected over time intervals such as days, months, or years.

Example: A time series is like a timeline of information. Imagine you have a diary where you write down your weight every day for a month. Each entry in your diary represents a data point in the time series. The time series simply shows how your weight changes over time, like a story of your weight journey. It's just a fancy way of saying we're keeping track of something over a period of time, whether it's daily, weekly, monthly, or yearly. So, a time series is basically a record of how something changes over time, like your weight, the temperature each day, or the stock prices over a month.



17. How is Overfitting different from Underfitting?

Overfitting and underfitting are two common problems that occur in machine learning and statistical models, affecting their performance on new, unseen data. They represent opposite ends of the spectrum in terms of model complexity and how well the model generalizes to new data.

Overfitting

Description: Overfitting occurs when a model learns the detail and noise in the training data to the extent that it negatively impacts the model's performance on new data. This means the model is too complex, capturing noise and patterns that do not generalize well outside of the training set.

Characteristics:

- High accuracy on training data but poor accuracy on test/unseen data.
- The model has too many parameters or too much flexibility.
- It captures random fluctuations in the training data as if they were important patterns.

Prevention and Solutions:

- Simplify the model by selecting a less complex model or reducing the number of parameters.
- Use techniques like regularization (L1, L2) to penalize overly complex models.
- Employ cross-validation to assess model performance on unseen data.
- Collect more training data to help the model generalize better.
- Use techniques like pruning in decision trees to remove parts of the model that do not contribute much to its predictive power.

Underfitting

Description: Underfitting occurs when a model is too simple to learn the underlying structure of the data. This means the model does not

fit the training data well and, as a result, cannot perform well on new data either.

Characteristics:

- Poor performance on both training and test data.
- The model has too few parameters or too little flexibility to capture the underlying trends of the data.
- It fails to capture important patterns or relationships in the data.

Prevention and Solutions:

- Increase model complexity by selecting a more complex model or adding more parameters.
- Feature engineering to introduce more relevant features that can help the model learn better.
- Reduce the amount of regularization if it's too high and constraining the model excessively.

Key Difference

The key difference between overfitting and underfitting lies in the model's complexity and its performance on training versus new data. Overfitting involves a complex model that performs well on training data but poorly on new data, while underfitting involves a too-simple model that performs poorly on both training and new data. Achieving a balance between the two, where the model is complex enough to learn the underlying patterns but not so complex that it learns the noise, is key to building models that generalize well to unseen data.

18. How do you treat outliers in a dataset?

Treating outliers in a dataset is crucial for accurate statistical analysis and machine learning modeling, as outliers can significantly affect the results. Here are some common strategies for handling outliers:

1. Identification

First, identify the outliers. This can be done through various methods:

Visual methods: Use box plots, scatter plots, or histograms to visually inspect the data for outliers.
Statistical methods: Employ rules based on standard deviations (e.g., values more than 3 standard deviations from the mean) or interquartile ranges (e.g., below Q1 - 1.5*IQR or above Q3 + 1.5*IQR, where Q1 and Q3 are the first and third quartiles, respectively, and IQR is the interquartile range).

2. Analysis
Once identified, analyze the outliers to determine their cause. Consider whether they result from data entry errors, measurement errors, or are natural variations in the data.

3. Treatment Options
After analysis, choose an appropriate method to handle them:
Removing: If outliers are due to errors or are not relevant to the study, they may be removed. However, care must be taken as this can lead to loss of information.
Transforming: Apply transformations to reduce the impact of outliers. Logarithmic, square root, or Box-Cox transformations are common.
Imputation: Replace outliers with more representative values, such as the mean, median, or a value predicted by a model. This method should be used cautiously to not introduce bias.
Capping: Outliers are capped at a certain value. For example, values above the 99th percentile might be set to the value of the 99th percentile.
Keeping: In some cases, outliers are genuine rare events or variations that are of interest to the study. In such cases, they are kept and analyzed separately.

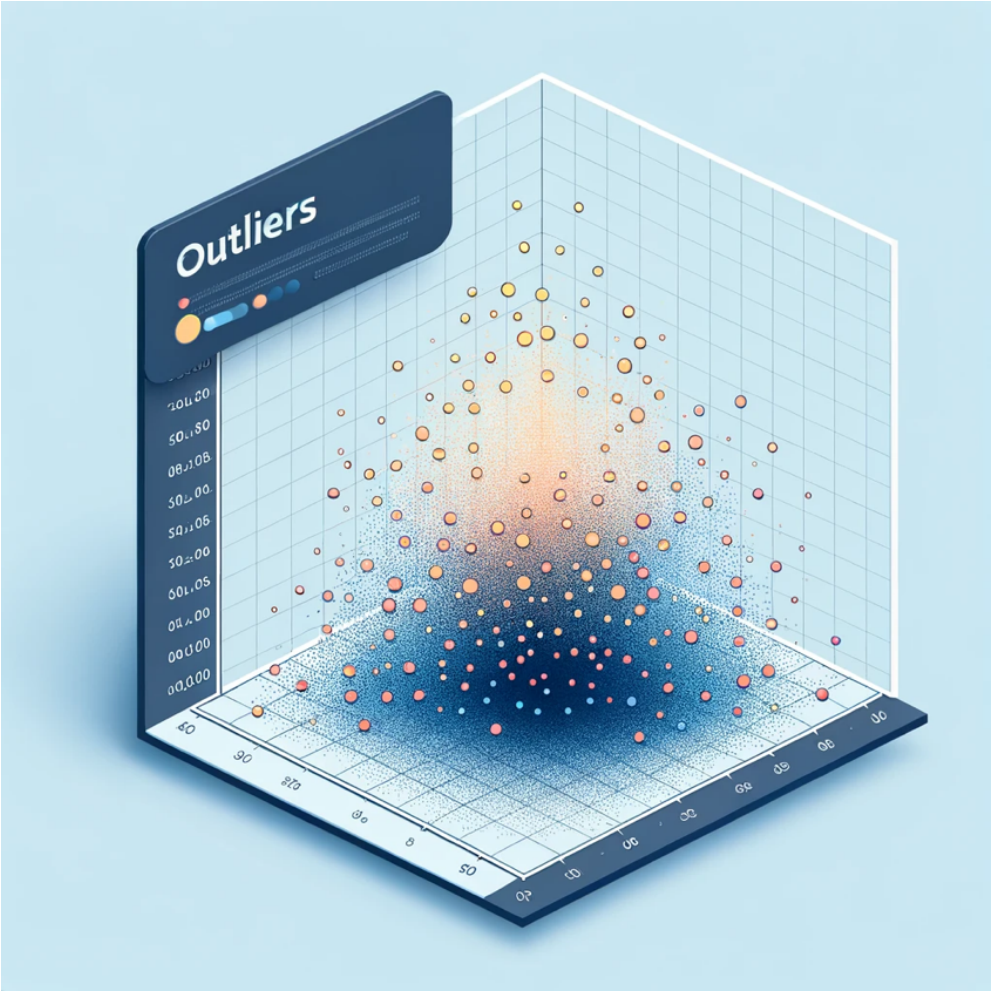
4. Modeling With Outliers
Some machine learning algorithms are more robust to outliers than others. For example, tree-based methods are generally less sensitive to outliers than linear regression or clustering algorithms. Choosing

or modifying algorithms to be more robust to outliers can be an effective strategy.

Best Practices

- The treatment of outliers should be informed by domain knowledge and the specific objectives of the analysis or modeling effort.
- Document the identification and treatment process for transparency and reproducibility.
- Be cautious of over-manipulating data, which can lead to bias or loss of valuable information.

Properly handling outliers is a balance between improving your model's performance and maintaining the integrity and representativeness of your data.



19. What are the different types of Hypothesis testing?

Hypothesis testing is the procedure used by statisticians and scientists to accept or reject statistical hypotheses. There are mainly two types of hypothesis testing:

- Null hypothesis: It states that there is no relation between the predictor and outcome variables in the population. H0 denoted it.

Example: There is no association between a patient's BMI and diabetes.

- Alternative hypothesis: It states that there is some relation between the predictor and outcome variables in the population. It is denoted by H1.

Example: There could be an association between a patient's BMI and diabetes.

- A/B testing, also known as split testing, is a specific application of hypothesis testing designed to compare two versions of a single variable to determine which version performs better on a given metric. Typically used in marketing, product development, and web design, A/B testing involves a controlled experiment with two variants: A (the control) and B (the treatment or the variant).

20. Explain the Type I and Type II errors in Statistics?

In statistics, Type I and Type II errors relate to incorrect decisions made based on the result of a hypothesis test.

- Type I Error: Occurs when the null hypothesis (H0*H*0) is incorrectly rejected when it is actually true. It's also known as a "false

positive." The probability of making a Type I error is denoted by **α** (alpha), which is the significance level of the test.

- Type II Error: Occurs when the null hypothesis (H0*H*0) is incorrectly accepted when it is false. It's also known as a "false negative." The probability of making a Type II error is denoted by **β** (beta), and 1−**β**1− represents the power of the test.

In summary, a Type I error is falsely claiming something is true when it's not, and a Type II error is failing to recognize something is true when it actually is.

21. How would you handle missing data in a dataset?

Handling missing data in a dataset involves several strategies, depending on the nature of the data and the extent of the missing values. Here are common methods:

1. Deletion:
    - Listwise deletion: Remove entire records where any data is missing.
    - Pairwise deletion: Use available data while ignoring missing values during statistical analysis.
2. Imputation:
    - Mean/Median/Mode substitution: Replace missing values with the mean, median, or mode of the column.
    - Predictive models: Use statistical models (e.g., regression, k-nearest neighbors) to predict and fill in missing values based on other data.
    - Hot-deck imputation: Replace a missing value with an observed response from a similar unit.
3. Using Algorithms that Support Missing Values: Some machine learning algorithms can handle missing data directly.
4. Assign a Unique Category: For categorical data, treat missing values as a separate category.

The choice of method depends on the analysis goals, data type, and missingness mechanism—whether it's missing completely at random (MCAR), missing at random (MAR), or missing not at random (MNAR).

22. Explain the concept of outlier detection and how you would identify outliers in a dataset.

Outlier detection refers to the process of identifying data points that deviate significantly from the majority of the data, suggesting a different mechanism generated them. Identifying outliers is crucial because they can affect the results of statistical analyses and models. Here's how you can identify outliers:

1. Statistical Methods:
    - Z-Score: Data points with a Z-score (the number of standard deviations from the mean) beyond a certain threshold (e.g., 3 or -3) are considered outliers.
    - IQR (Interquartile Range) Method: Data points lying below Q1 - 1.5*IQR or above Q3 + 1.5*IQR (where Q1 and Q3 are the first and third quartiles, respectively) are outliers.
2. Visualization:
    - Box Plots: Visualize the distribution of the data and identify outliers as points that fall outside of the whiskers.
    - Scatter Plots: Help in visually spotting outliers by showing data points that fall far away from the cluster of other data points.
3. Machine Learning Methods:
    - Algorithms like Isolation Forest, DBSCAN, and Local Outlier Factor (LOF) can automatically detect outliers in complex datasets.

Choosing the right method depends on the nature of the data and the context of the analysis. It's also important to investigate the cause of outliers before deciding to remove or adjust them, as they can sometimes be valuable for understanding the dataset better.

Data Analyst Excel Interview Questions

23. In Microsoft Excel, a numeric value can be treated as a text value if it precedes with what?

In Microsoft Excel, a numeric value can be treated as a text value if it is preceded by an apostrophe ('). This tells Excel to treat the following numeric entry as text, preserving any leading zeros or formatting that would normally be lost if the value were entered as a number.

24. What is the difference between COUNT, COUNTA, COUNTBLANK, and COUNTIF in Excel?

In Excel, **COUNT**, **COUNTA**, **COUNTBLANK**, and **COUNTIF** are functions used to count cells in a range based on different criteria:

- **COUNT**: Counts the number of cells that contain numeric values in a range.
- **COUNTA**: Counts the number of cells that are not empty in a range, including numbers, text, errors, etc.
- **COUNTBLANK**: Counts the number of empty cells in a specified range.
- **COUNTIF**: Counts the number of cells that meet a single condition specified by a criterion, such as counting cells that contain a specific value or meet a certain condition (e.g., ">10").

25. How do you make a dropdown list in MS Excel?

To create a dropdown list in MS Excel, follow these short steps:

1. **Select the cell** where you want the dropdown list.
2. Go to the **Data** tab on the ribbon.
3. Click on **Data Validation** in the 'Data Tools' group.
4. In the Data Validation dialog box, under the **Settings** tab, select **List** from the 'Allow' dropdown menu.

5. In the **Source** box, either type the list items separated by commas or refer to a range of cells where your list items are located.
6. Click **OK**.

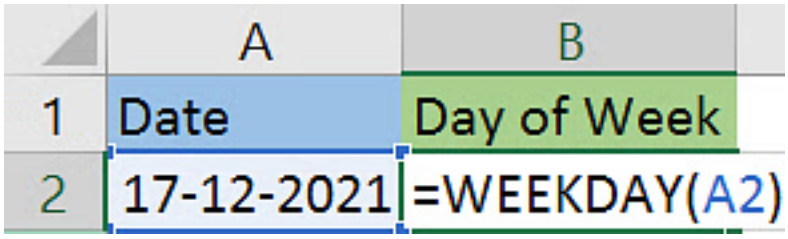
Now, the selected cell will have a dropdown list with the options you specified.

26. Can you provide a dynamic range in "Data Source" for a Pivot table?

Yes, you can provide a dynamic range in the "Data Source" of Pivot tables. To do that, you need to create a named range using the offset function and base the pivot table using a named range constructed in the first step.

27. What is the function to find the day of the week for a particular date value?

The get the day of the week, you can use the WEEKDAY() function.



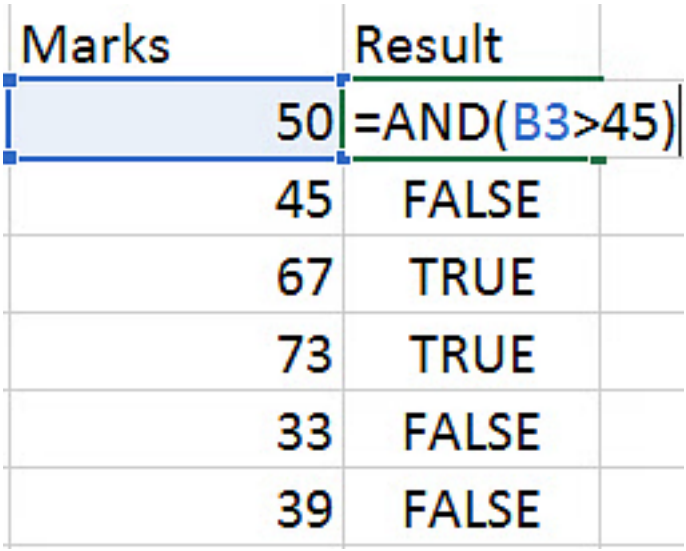
The above function will return 6 as the result, i.e., 17th December is a Saturday.

28. How does the AND( ) function work in Excel?

AND( ) is a logical function that checks multiple conditions and returns TRUE or FALSE based on whether the conditions are met.

Syntax: AND(logica1,[logical2],[logical3]....)

In the below example, we are checking if the marks are greater than 45. The result will be true if the mark is >45, else it will be false.



Marks	Result
50	=AND(B3>45)
45	FALSE
67	TRUE
73	TRUE
33	FALSE
39	FALSE

29. Explain how VLOOKUP works in Excel?

VLOOKUP (Vertical Lookup) in Excel searches for a value in the first column of a specified table range and returns a value in the same row from a specified column. The basic syntax is:

```
VLOOKUP(lookup_value, table_array, col_index_num, [range_lookup])
```

- **lookup_value**: The value you're searching for in the first column of the table array.
- **table_array**: The range of cells containing the data. VLOOKUP searches for the lookup_value in the first column of this range.
- **col_index_num**: The column number in the table from which to retrieve the value. The first column in the table is 1, the second is 2, and so on.
- **[range_lookup]**: Optional. If TRUE or omitted, VLOOKUP returns the closest match. If FALSE, it only returns an exact match or #N/A if no match is found.

VLOOKUP works vertically, scanning the first column of the table_array to find the matching lookup_value, then moves right to return the value from the specified column index.

Let's understand VLOOKUP with an example.

	A	B	C	D	E
1	First Name	Last Name	Department	City	Date Hired
2	Ben	Zampa	HR	Chicago	10-11-2001
3	Stuart	Carry	Marketing	Kansas	20-06-2002
4	Jenson	Button	Operations	New York	01-12-2004
5	Lucy	Davis	Sales	Los Angeles	25-02-2011
6	Trent	Patinson	IT	Boston	17-08-2015
7	Jhonny	Evans	Sales	Houston	10-01-2018

If you wanted to find the department to which Stuart belongs to, you could use the VLOOKUP function as shown below:

9	Vlookup				
10	First Name	Last Name	Department	City	Date Hired
11	Stuart		=VLOOKUP(A11,A2:E7,3,0)		

Here, A11 cell has the lookup value, A2:E7 is the table array, 3 is the column index number with information about departments, and 0 is the range lookup.

If you hit enter, it will return "Marketing", indicating that Stuart is from the marketing department.

30. What function would you use to get the current date and time in Excel?

In Excel, you can use the TODAY( ) and NOW( ) function to get the current date and time.

31. Using the below sales table, calculate the total quantity sold by sales representatives whose name starts with A, and the cost of each item they have sold is greater than 10.

	A	B	C	D	E	F	G
1	Date	Sales Rep	Item	Cost each	Quantity	Sale total	Cost range
2	05-07-2005	A. Yamamoto	J21344A	₹ 19.50	4	$ 78.00	Low
3	06-07-2005	Q. Ackerman	Q00345B	₹ 39.00	19	$ 741.00	High
4	26-07-2005	J. Wilson	L98700F	₹ 8.25	2	$ 16.50	Low
5	12-07-2005	F. Rosenstein	B20011A	₹ 22.15	19	$ 420.85	Average
6	27-07-2005	J. Wilson	J21344A	₹ 19.50	19	$ 370.50	Low
7	28-07-2005	Q. Ackerman	Q00345B	₹ 39.00	19	$ 741.00	High
8	23-07-2005	A. Yamamoto	C55440D	₹ 16.75	2	$ 33.50	Low
9	01-07-2005	F. Rosenstein	Q00345B	₹ 39.00	5	$ 195.00	High
10	24-07-2005	A. Mathews	L98700F	₹ 8.25	15	$ 123.75	Low
11	29-07-2005	D.F. Chang	J21344A	₹ 19.50	11	$ 214.50	Low
12	21-07-2005	F. Rosenstein	B20011A	₹ 22.15	15	$ 332.25	Average
13	06-07-2005	J. Wilson	Q00345B	₹ 39.00	18	$ 702.00	High
14	18-07-2005	S. Muller	C55440D	₹ 16.75	17	$ 284.75	Low
15	03-07-2005	O. McBride	J21344A	₹ 19.50	9	$ 175.50	Low
16	28-07-2005	L. Sanchez	J21344A	₹ 19.50	15	$ 292.50	Low
17	22-07-2005	F. Rosenstein	L98700F	₹ 8.25	11	$ 90.75	Low
18	04-07-2005	J. Wilson	C55440D	₹ 16.75	4	$ 67.00	Low
19	07-07-2005	W. Carver	B20011A	₹ 22.15	3	$ 66.45	Average
20	03-07-2005	A. Symonds	J21344A	₹ 19.50	7	$ 136.50	Low

You can use the SUMIFS() function to find the total quantity.
For the Sales Rep column, you need to give the criteria as "A*" - meaning the name should start with the letter "A". For the Cost each column, the criteria should be ">10" - meaning the cost of each item is greater than 10.

	A	B	C	D	E	F	G	H	I	J
1	Date	Sales Rep	Item	Cost each	Quantity	Sale total	Cost range			
2	05-07-2005	A. Yamamoto	J21344A	₹ 19.50	4	$ 78.00	Low			
3	06-07-2005	Q. Ackerman	Q00345B	₹ 39.00	19	$ 741.00	High			
4	26-07-2005	J. Wilson	L98700F	₹ 8.25	2	$ 16.50	Low			
5	12-07-2005	F. Rosenstein	B20011A	₹ 22.15	19	$ 420.85	Average			
6	27-07-2005	J. Wilson	J21344A	₹ 19.50	19	$ 370.50	Low			
7	28-07-2005	Q. Ackerman	Q00345B	₹ 39.00	19	$ 741.00	High			
8	23-07-2005	A. Yamamoto	C55440D	₹ 16.75	2	$ 33.50	Low			
9	01-07-2005	F. Rosenstein	Q00345B	₹ 39.00	5	$ 195.00	High	=SUMIFS(E2:E20,B2:B20, "A*", D2:D20, ">10")		
10	24-07-2005	A. Mathews	L98700F	₹ 8.25	15	$ 123.75	Low			
11	29-07-2005	D.F. Chang	J21344A	₹ 19.50	11	$ 214.50	Low			
12	21-07-2005	F. Rosenstein	B20011A	₹ 22.15	15	$ 332.25	Average			
13	06-07-2005	J. Wilson	Q00345B	₹ 39.00	18	$ 702.00	High			
14	18-07-2005	S. Muller	C55440D	₹ 16.75	17	$ 284.75	Low			
15	03-07-2005	O. McBride	J21344A	₹ 19.50	9	$ 175.50	Low			
16	28-07-2005	L. Sanchez	J21344A	₹ 19.50	15	$ 292.50	Low			
17	22-07-2005	F. Rosenstein	L98700F	₹ 8.25	11	$ 90.75	Low			
18	04-07-2005	J. Wilson	C55440D	₹ 16.75	4	$ 67.00	Low			
19	07-07-2005	W. Carver	B20011A	₹ 22.15	3	$ 66.45	Average			
20	03-07-2005	A. Symonds	J21344A	₹ 19.50	7	$ 136.50	Low			

The result is 13.

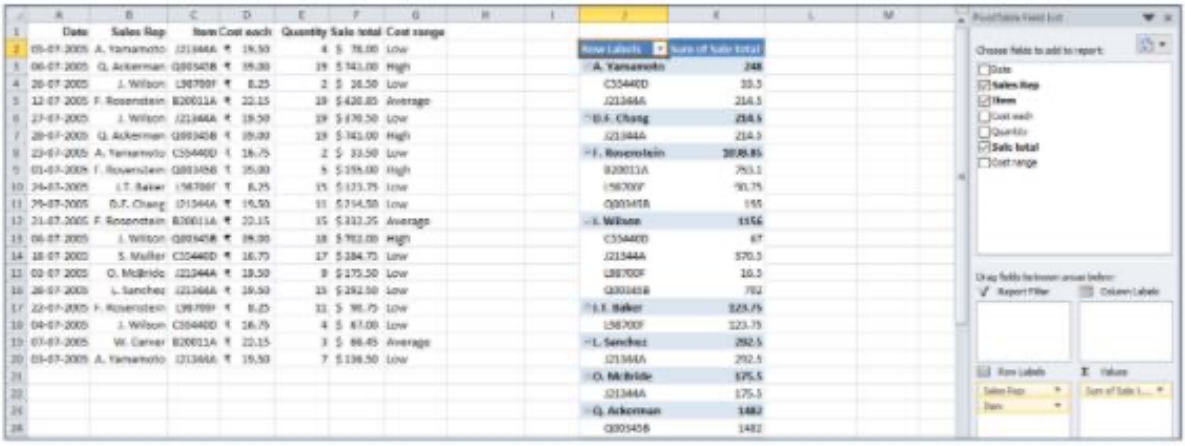
33. Using the data given below, create a pivot table to find the total sales made by each sales representative for each item. Display the sales as % of the grand total

	A	B	C	D	E	F	G
1	Date	Sales Rep	Item	Cost each	Quantity	Sale total	Cost range
2	05-07-2005	A. Yamamoto	J21344A	₹ 19.50	4	$ 78.00	Low
3	06-07-2005	Q. Ackerman	Q00345B	₹ 39.00	19	$ 741.00	High
4	26-07-2005	J. Wilson	L98700F	₹ 8.25	2	$ 16.50	Low
5	12-07-2005	F. Rosenstein	B20011A	₹ 22.15	19	$ 420.85	Average
6	27-07-2005	J. Wilson	J21344A	₹ 19.50	19	$ 370.50	Low
7	28-07-2005	Q. Ackerman	Q00345B	₹ 39.00	19	$ 741.00	High
8	23-07-2005	A. Yamamoto	C55440D	₹ 16.75	2	$ 33.50	Low
9	01-07-2005	F. Rosenstein	Q00345B	₹ 39.00	5	$ 195.00	High
10	24-07-2005	J.T. Baker	L98700F	₹ 8.25	15	$ 123.75	Low
11	29-07-2005	D.F. Chang	J21344A	₹ 19.50	11	$ 214.50	Low
12	21-07-2005	F. Rosenstein	B20011A	₹ 22.15	15	$ 332.25	Average
13	06-07-2005	J. Wilson	Q00345B	₹ 39.00	18	$ 702.00	High
14	18-07-2005	S. Muller	C55440D	₹ 16.75	17	$ 284.75	Low
15	03-07-2005	O. McBride	J21344A	₹ 19.50	9	$ 175.50	Low
16	28-07-2005	L. Sanchez	J21344A	₹ 19.50	15	$ 292.50	Low
17	22-07-2005	F. Rosenstein	L98700F	₹ 8.25	11	$ 90.75	Low
18	04-07-2005	J. Wilson	C55440D	₹ 16.75	4	$ 67.00	Low
19	07-07-2005	W. Carver	B20011A	₹ 22.15	3	$ 66.45	Average
20	03-07-2005	A. Yamamoto	J21344A	₹ 19.50	7	$ 136.50	Low

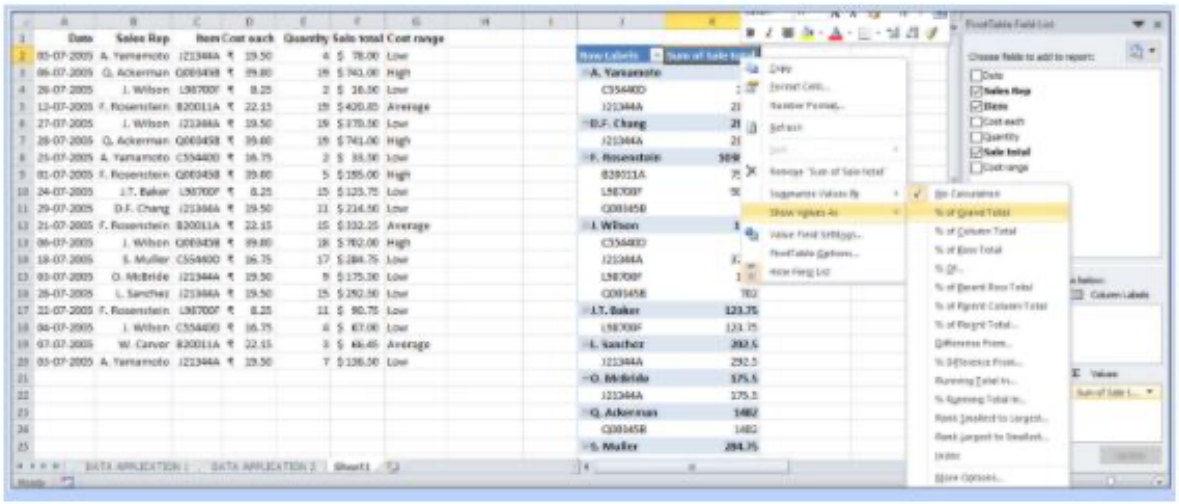
- Select the entire table range, click on the Insert tab and choose PivotTable

- Select the table range and the worksheet where you want to place the pivot table

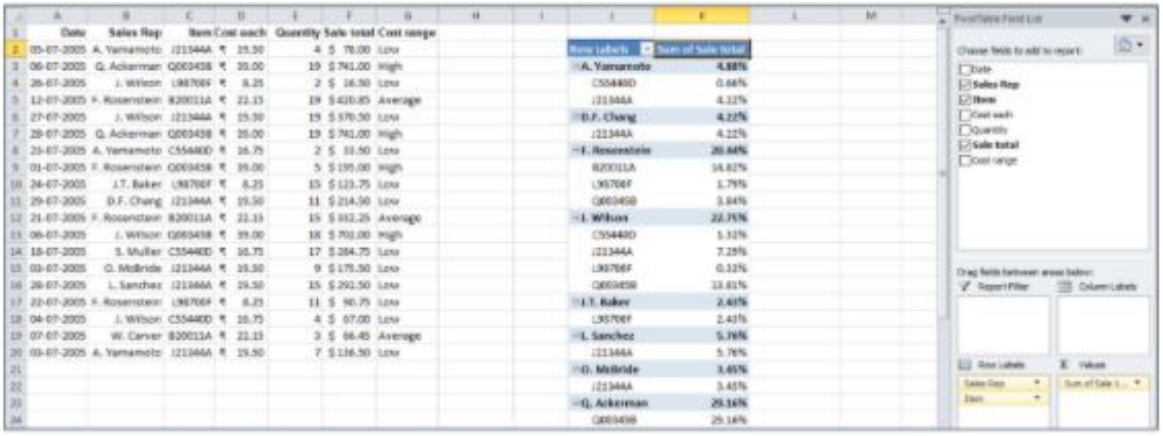
- Drag Sale total on to Values, and Sales Rep and Item on to Row Labels. It will give the sum of sales made by each representative for every item they have sold.



- Right-click on "Sum of Sale Total' and expand Show Values As to select % of Grand Total.



- Below is the resultant pivot table.



SQL Interview Questions for Data Analysts

34. How do you subset or filter data in SQL?

To subset or filter data in SQL, we use WHERE and HAVING clauses.

Consider the following movie table.

Title	Director	Year	Duration
Race	Stephen Hopkins	2016	134
Cars	John Lasseter	2006	117
Toy Story	John Lasseter	1995	81
The Incredibles	Brad Bird	2004	116
Brave	Brenda Chapman	2012	102
Ratatouille	Brad Bird	2007	115
Vertigo	Alfred Hitchcock	1958	128

Using this table, let's find the records for movies that were directed by Brad Bird.

```
select * from Movies where Director = 'Brad Bird';
```

Title	Director	Year	Duration
The Incredibles	Brad Bird	2004	116
Ratatouille	Brad Bird	2007	115

Now, let's filter the table for directors whose movies have an average duration greater than 115 minutes.

```
select Director, sum(Duration) as total_duration,
avg(Duration) as avg_duration from Movies
group by Director having avg(Duration)>115
```

Director	total_duration	avg_duration
Stephen Hopkins	134	134
Brad Bird	231	115.5
Alfred Hitchcock	128	128

35. What is the difference between a WHERE clause and a HAVING clause in SQL?

In SQL, both the WHERE and HAVING clauses are used to filter records; however, they serve different purposes and are used in different contexts.

1. WHERE Clause:
    - The WHERE clause is used to filter rows based on specified conditions before any groupings are made.
    - It is applied to individual rows in the source tables.
    - You can use the WHERE clause with SELECT, UPDATE, DELETE, etc., to filter records based on specific conditions.
    - The WHERE clause cannot be used with aggregate functions like SUM(), AVG(), COUNT(), etc., directly because it filters rows before these functions are applied.

2. HAVING Clause:
    - The HAVING clause is used to filter groups based on specified conditions after the groupings are made (after the GROUP BY operation).
    - It is typically used with the GROUP BY clause but can be used without it if you are filtering on an aggregate function.
    - The HAVING clause is specifically designed to be used with aggregate functions (SUM(), AVG(), COUNT(), etc.), allowing you to specify conditions that filter group results.
    - You cannot use the HAVING clause to filter individual rows directly; it filters groups of rows that meet the aggregate conditions.

Example Usage:

- WHERE Clause Example:

```
SELECT employee_id, name, salary
FROM employees
WHERE salary > 50000;
```

This query filters individual employee records to retrieve only those with a salary greater than 50,000 before any grouping is considered.

- HAVING Clause Example:

```
SELECT department_id, AVG(salary) AS average_salary
FROM employees
GROUP BY department_id
HAVING AVG(salary) > 50000;
```

This query first groups employees by **department_id**, calculates the average salary for each department, and then filters out the groups (departments) where the average salary is not greater than 50,000.
In summary, the main difference is that the **WHERE** clause is used to filter rows before grouping, while the **HAVING** clause is used to filter groups or aggregates after grouping.

36. Is the below SQL query correct? If not, how will you rectify it?

```
SELECT custid, YEAR(order_date) AS order_year
FROM Order WHERE order_year >= 2016;
```

The query stated above is incorrect as we cannot use the alias name while filtering data using the WHERE clause. It will throw an error.

```
SELECT custid, YEAR(order_date) AS order_year
FROM Order WHERE YEAR(order_date) >= 2016;
```

37. How are Union, Intersect, and Except used in SQL?

In SQL, the UNION, INTERSECT, and EXCEPT (or MINUS in some databases) operators are used to combine the results of two or more SELECT statements. However, each of them serves a different purpose and is used under different scenarios:

1. UNION:
    - The UNION operator is used to combine the result sets of two or more SELECT statements into a single result set, including all the rows belonging to all the selected data.
    - It automatically removes duplicate rows from the result.
    - Each SELECT statement within the UNION must have the same number of columns in the result sets with similar data types. The columns in each SELECT statement must also be in the same order.
    - Syntax Example:

```
SELECT column_name(s) FROM table1
UNION
SELECT column_name(s) FROM table2;
```

- If you want to include all duplicates, you can use UNION ALL which is faster than UNION because it doesn't check for duplicates.

2. INTERSECT:
    - The INTERSECT operator is used to return the rows that are common to the result sets of two or more SELECT statements.
    - It essentially finds the intersection of the result sets.
    - Like UNION, each SELECT statement within the INTERSECT must have the same number of columns with the columns having similar data types, and the columns must also be in the same order.
    - Syntax Example:

```
SELECT column_name(s) FROM table1
INTERSECT
SELECT column_name(s) FROM table2;
```

3. EXCEPT (or MINUS):

- The **EXCEPT** operator (known as **MINUS** in some RDBMS like Oracle) is used to return the rows from the first **SELECT** statement that are not present in the result set of the second **SELECT** statement.
- It essentially subtracts the result set of the second **SELECT** from the first.
- The same rules apply regarding the number of columns, data types, and order as with **UNION** and **INTERSECT**.
- Syntax Example:

```
SELECT column_name(s) FROM table1
EXCEPT
SELECT column_name(s) FROM table2;
```

Usage Considerations:

- These operators are powerful tools for performing set operations in SQL, allowing for complex queries and data analysis.
- When using these operators, it's important to ensure that the data types and the order of columns match across all **SELECT** statements.
- The choice between these operators depends on the specific requirement, whether it's to combine all results (**UNION**), find common results (**INTERSECT**), or identify unique results in one query that are not found in another (**EXCEPT**).

38. What is a Subquery in SQL?

A subquery in SQL is a query nested inside another SQL query. It's essentially a query within a query, used to perform operations that require multiple steps of logic in a single query. Subqueries can return individual values, a single list of values, or a full result set, depending on how they are used.

Subqueries can be used in various parts of the main SQL statement, including the **SELECT**, **FROM**, **WHERE**, and **HAVING** clauses, among

others. They are a powerful feature that allows for complex data retrieval operations that would be difficult or impossible to perform with a single level of querying.

Types of Subqueries

1. **Scalar Subqueries:** These return a single value (one row and one column). They are often used in the SELECT list or in a WHERE clause for comparison purposes.
2. **Column Subqueries:** These return a single column but potentially multiple rows. They are typically used in the WHERE clause with the IN operator.
3. **Row Subqueries:** These return one or more rows but are limited to a specific number of columns. They are used for comparisons with multiple columns.
4. **Table Subqueries (or Derived Tables):** These return a full result set and are often used in the FROM clause. They act as a temporary table that can be queried in the outer select statement.

Usage Examples

- Scalar Subquery Example:

```
SELECT employee_name, salary
FROM employees
WHERE salary > (SELECT AVG(salary) FROM employees);
```

- This example selects the names and salaries of employees who earn more than the average salary.

- Column Subquery Example:

```
SELECT employee_name
FROM employees
WHERE department_id IN (SELECT department_id FROM departments
WHERE location = 'New York');
```

This query finds the names of employees working in departments located in New York.

- Table Subquery Example:

```
SELECT a.
FROM (SELECT * FROM employees WHERE salary > 50000) AS a
JOIN departments d ON a.department_id = d.department_id;
```

This example joins a derived table (employees with a salary over 50,000) with the departments table.

Considerations

- **Performance:** Subqueries can sometimes lead to less efficient query plans. It's important to consider performance, especially with complex or deeply nested subqueries.
- **Readability:** While subqueries can make queries more logically comprehensive, they can also make them harder to read and maintain. Balancing complexity and maintainability is key.
- **Alternatives:** Sometimes, joining tables or using window functions can achieve the same result as a subquery but with better performance or readability.
  Subqueries are a versatile tool in SQL, enabling sophisticated and dynamic data analysis within a single query operation

39. Using the product_price table, write an SQL query to find the record with the fourth-highest market price.

	Sec_id	Prc_date	Mkt_Price	Currency	Pricing_factor
1	HCL205	2013-07-17	487.39	INR	1
2	HDFC305	2013-07-15	1187.15	INR	1
3	HUL109	2013-03-23	20	USD	100
4	ICIC201	2012-06-24	50	GBP	150
5	INC501	2011-01-10	15	SGD	50
6	INF409	2012-04-01	25	USD	100
7	Mar408	2009-07-05	1257.39	PKR	0.7
8	Ran208	2008-05-11	112	CHF	80
9	TCS103	2007-09-08	114	AUD	90
10	WIP309	2008-10-05	120	AUD	90

Fig: Product Price table

```
select top 1 * from
(select top 4 * from product_price
order by mkt_price desc) as sp order by mkt_price asc
```

select top 4 * from product_price order by mkt_price desc;

	Sec_id	Prc_date	Mkt_Price	Currency	Pricing_factor
1	Mar408	2009-07-05	1257.39	PKR	0.7
2	HDFC305	2013-07-15	1187.15	INR	1
3	HCL205	2013-07-17	487.39	INR	1
4	WIP309	2008-10-05	120	AUD	90

Now, select the top one from the above result that is in ascending order of mkt_price.

	Sec_id	Prc_date	Mkt_Price	Currency	Pricing_factor
1	WIP309	2008-10-05	120	AUD	90

40. From the product_price table, write an SQL query to find the total and average market price for each currency where the average market price is greater than 100, and the currency is in INR or AUD.

	Sec_id	Prc_date	Mkt_Price	Currency	Pricing_factor
1	HCL205	2013-07-17	487.39	INR	1
2	HDFC305	2013-07-15	1187.15	INR	1
3	HUL109	2013-03-23	20	USD	100
4	ICIC201	2012-06-24	50	GBP	150
5	INC501	2011-01-10	15	SGD	50
6	INF409	2012-04-01	25	USD	100
7	Mar408	2009-07-05	1257.39	PKR	0.7
8	Ran208	2008-05-11	112	CHF	80
9	TCS103	2007-09-08	114	AUD	90
10	WIP309	2008-10-05	120	AUD	90

The SQL query is as follows:

```
select currency,sum(mkt_price) as total_price,avg(mkt_price) as avg_price
from product_price where currency in('inr','aud') group by currency
having avg(mkt_price)>50
```

The output of the query is as follows:

	currency	total_price	avg_price
1	AUD	234	117
2	INR	1674.54	837.27

41. Using the product and sales order detail table, find the products with total units sold greater than 1.5 million.

	ProductID	Name	ProductNumber	MakeFlag	FinishedGoodsFlag	Color	SafetyStockLevel	ReorderPoint	StandardCost	ListPrice	Size	SizeUnitMeasureCode	Wei
1	1	Adjustable Race	AR-5381	0	0	NULL	1000	750	0.00	0.00	NULL	NULL	NU
2	2	Bearing Ball	BA-8327	0	0	NULL	1000	750	0.00	0.00	NULL	NULL	NU
3	3	BB Ball Bearing	BE-2349	1	0	NULL	800	600	0.00	0.00	NULL	NULL	NU
4	4	Headset Ball Bearings	BE-2908	0	0	NULL	800	600	0.00	0.00	NULL	NULL	NU
5	316	Blade	BL-2036	1	0	NULL	800	600	0.00	0.00	NULL	NULL	NU
6	317	LL Crankarm	CA-5965	0	0	Black	500	375	0.00	0.00	NULL	NULL	NU
7	318	ML Crankarm	CA-6738	0	0	Black	500	375	0.00	0.00	NULL	NULL	NU
8	319	HL Crankarm	CA-7457	0	0	Black	500	375	0.00	0.00	NULL	NULL	NU
9	320	Chainring Bolts	CB-2903	0	0	Silver	1000	750	0.00	0.00	NULL	NULL	NU
10	321	Chainring Nut	CN-6137	0	0	Silver	1000	750	0.00	0.00	NULL	NULL	NU
11	322	Chainring	CR-7833	0	0	Black	1000	750	0.00	0.00	NULL	NULL	NU
12	323	Crown Race	CR-9981	0	0	NULL	1000	750	0.00	0.00	NULL	NULL	NU
13	324	Chain Stays	CS-2812	1	0	NULL	1000	750	0.00	0.00	NULL	NULL	NU
14	325	Decal 1	DC-8732	0	0	NULL	1000	750	0.00	0.00	NULL	NULL	NU
15	326	Decal 2	DC-9824	0	0	NULL	1000	750	0.00	0.00	NULL	NULL	NU
16	327	Down Tube	DT-2377	1	0	NULL	800	600	0.00	0.00	NULL	NULL	NU
17	328	Mountain End Caps	EC-M092	1	0	NULL	1000	750	0.00	0.00	NULL	NULL	NU

Fig: Products table

	SalesOrderID	SalesOrderDetailID	CarrierTrackingNumber	OrderQty	ProductID	SpecialOfferID	UnitPrice	UnitPriceDiscount	LineTotal	rowguid
1	43659	1	4911-403C-98	1	776	1	2024.994	0.00	2024.994000	B207C96D-D9E6-402B-8470-2CC176C42283
2	43659	2	4911-403C-98	3	777	1	2024.994	0.00	6074.982000	7ABB600D-1E77-41BE-9FE5-89142CFC08FA
3	43659	3	4911-403C-98	1	778	1	2024.994	0.00	2024.994000	475CF8C6-49F6-486E-B0AD-AFC6A50CDD2F
4	43659	4	4911-403C-98	1	771	1	2039.994	0.00	2039.994000	04C4DE91-5815-45D6-8670-F462719FBCE3
5	43659	5	4911-403C-98	1	772	1	2039.994	0.00	2039.994000	5A74C7D2-E641-438E-A7AC-37BF23280301
6	43659	6	4911-403C-98	2	773	1	2039.994	0.00	4079.988000	CE472532-A4C0-45BA-816E-EEFD3FD848B3
7	43659	7	4911-403C-98	1	774	1	2039.994	0.00	2039.994000	80667040-F962-4EE3-96E0-AECA108E0D4F
8	43659	8	4911-403C-98	3	714	1	28.8404	0.00	86.521200	E9D54907-E7B7-4969-80D9-76BA69F8A836
9	43659	9	4911-403C-98	1	716	1	28.8404	0.00	28.840400	AA542630-BDCD-4CE5-89A0-C1BF82747725
10	43659	10	4911-403C-98	6	709	1	5.70	0.00	34.200000	AC769034-3C2F-495C-A5A7-3B71CDB25D4E
11	43659	11	4911-403C-98	2	712	1	5.1865	0.00	10.373000	06A66921-6B9F-4199-A912-DDAFD3834728
12	43659	12	4911-403C-98	4	711	1	20.1865	0.00	80.746000	0E371EE3-253E-4BB0-B813-E3CF4224F972
13	43660	13	6431-4D57-83	1	762	1	419.4589	0.00	419.458900	419A1302-AC7A-4044-97B2-66D9D14CD02E
14	43660	14	6431-4D57-83	1	758	1	874.794	0.00	874.794000	5D0B2B03-1D4C-4C34-9696-C14C58E7301C
15	43661	15	4E0A-4F89-AE	1	745	1	809.76	0.00	809.760000	EDE1759E-6733-4C7B-A43F-DC6F48002D8A
16	43661	16	4E0A-4F89-AE	1	743	1	714.7043	0.00	714.704300	FE10BF09-D477-485B-9541-27AE8053A6D4
17	43661	17	4E0A-4F89-AE	2	747	1	714.7043	0.00	1429.408600	B136852E-24C9-4006-8048-B14AEFE8C337

Fig: Sales order detail table

We can use an inner join to get records from both the tables. We'll join the tables based on a common key column, i.e., ProductID.

```
select pp.name,SUM(sod.UnitPrice) as sales, pp.ProductID
from Production.Product as pp inner join Sales.SalesOrderDetail as sod
on pp.ProductID=sod.ProductID group by pp.name,pp.ProductID
having SUM(sod.UnitPrice)>1500000
```

The result of the SQL query is shown below.

	name	sales	ProductID
1	Mountain-200 Black, 38	2166145.9708	782
2	Mountain-200 Black, 42	2090728.5016	783
3	Mountain-200 Black, 46	1944554.8535	784
4	Mountain-200 Silver, 38	1990662.4446	779
5	Mountain-200 Silver, 42	1884496.3881	780
6	Mountain-200 Silver, 46	1919478.5226	781

42. How do you write a stored procedure in SQL?

You must be prepared for this question thoroughly before your next data analyst interview. The stored procedure is an SQL script that is used to run a task several times.

Let's look at an example to create a stored procedure to find the sum of the first N natural numbers' squares.

- Create a procedure by giving a name, here it's squaresum1
- Declare the variables
- Write the formula using the set statement
- Print the values of the computed variable
- To run the stored procedure, use the EXEC command

```
CREATE PROCEDURE squaresum1

(@n int)
as
begin
declare @sum int
set @sum=@n*(@n+1)*(2*@n+1)/6

print ' first  '+cast(@n as varchar(20))+' natural numbers'
print ' sum of square is  '+cast(@sum as varchar(40))
END
```

Output: Display the sum of the square for the first four natural numbers

```
EXEC squaresum1 4
```

Messages

```
first  4 natural numbers
sum of square is  30
```

43. Write an SQL stored procedure to find the total even number between two users given numbers.

```
create procedure count_even
(@n1 int,@n2 int)
as
begin
declare @count int
set @count=0
while(@n1<@n2)
begin
if(@n1%2=0)
begin
set @count=@count+1
print 'even number:' + cast(@n1 as varchar(10)) +
' count is:'+cast(@count as varchar(10))
      end
    --  else
     -- print 'odd number' + cast(@n1 as varchar(10))
  set @n1=@n1+1
  end
  print 'total number of even numbers is :' + cast(@count as varchar(10))
end
```

Here is the output to print all even numbers between 30 and 45.

```
exec count_even 30,45
```

```
even number:30 count is:1
even number:32 count is:2
even number:34 count is:3
even number:36 count is:4
even number:38 count is:5
even number:40 count is:6
even number:42 count is:7
even number:44 count is:8
total number of even numbers is :8
```

Tableau Data Analyst Interview Questions

44. How is joining different from blending in Tableau?

In Tableau, joining and blending are two distinct methods for combining data from multiple sources, but they serve different purposes and are used in different scenarios.

Joining

- **What it is:** Joining is the process of merging two or more tables by a common field (column), creating a single table where the columns from one table are appended to those of another, based on the join condition. Joins are performed within a single data source.
- **How it works:** Joins in Tableau work similarly to SQL joins. You can perform inner, left, right, and full outer joins. The data is combined at the database level before it is brought into Tableau for analysis.
- **Use cases:** Joining is ideal when your data resides in the same database or data source and you want to analyze it as a single dataset. It's used when you need to combine detailed, row-level data from related tables.

Blending

- **What it is:** Data blending is the process of combining data from multiple data sources on a single sheet in Tableau. Instead of merging the data at the database level, blending happens at the visualization level, allowing you to display integrated data from separate sources in a single view.
- **How it works:** In data blending, one data source is designated as the primary, and others as secondary. Tableau will aggregate the data from each source independently and then blend the

summarized data based on common dimensions specified by the user.

- Use cases: Blending is useful when you have data in separate data sources (for example, different databases, spreadsheets, or cloud services) and you want to analyze them together without creating a single, unified database. It's particularly useful for integrating aggregated data or when joining data sources is not possible or practical.

Key Differences

- Level of Integration: Joining combines data at a more fundamental level, creating a unified dataset before analysis. Blending combines data at a higher level, aggregating data from each source separately before integrating it for visualization.
- Data Source Limitations: Joining is limited to tables within the same data source, while blending allows for the integration of data across different data sources.
- Performance: Joins can be more efficient, especially with large datasets, as the data combination is handled by the database before analysis. Blending might introduce performance issues due to the need to aggregate data from multiple sources independently before combining it.
- Flexibility in Data Analysis: Blending provides a flexible approach to integrating disparate data sources, which might not be possible with joins. However, this flexibility can come with limitations in the granularity of data analysis, as blending operates on aggregated data rather than row-level detail.
  In summary, the choice between joining and blending in Tableau depends on the specific requirements of your data analysis project, including the structure and location of your data, as well as the level of detail and performance considerations for your analysis

45. What do you understand by LOD in Tableau?

LOD, or Level of Detail expressions in Tableau, provide a way to specify the granularity at which you want to perform calculations, independent of the visualization's level of detail. LOD expressions allow for more complex calculations across different levels of granularity in your data, offering flexibility in how data aggregations and computations are performed in your Tableau visualizations.

There are three types of LOD expressions in Tableau:

1. **FIXED:** This LOD expression calculates the value at the level of detail specified by the dimensions in the expression, regardless of the visualization's level of detail. The calculation is fixed at the level specified by the expression.
    - Example: **{FIXED [Region]: SUM([Sales])}** calculates the total sales for each region, regardless of the detail level in the view.
2. **INCLUDE:** This LOD expression calculates the value at the level of detail specified by the dimensions in the visualization, plus any additional dimensions specified in the expression. It effectively increases the level of detail of the calculation by including additional dimensions.
    - Example: **{INCLUDE [Product]: AVG([Profit])}** calculates the average profit per product, including the product dimension in the calculation even if it's not present in the visualization.
3. **EXCLUDE:** This LOD expression calculates the value at the level of detail of the visualization, excluding the dimensions specified in the expression. It decreases the level of detail of the calculation by removing dimensions from the calculation context.
    - Example: **{EXCLUDE [Product]: SUM([Sales])}** calculates total sales, excluding the product level detail even if products are detailed in the view.

Uses and Importance

- **Advanced Aggregations:** LOD expressions allow for more sophisticated aggregations that are not constrained by the view's level of detail. This is particularly useful for creating benchmarks, comparisons, and performing calculations that need to be standardized across different granularities.
- **Custom Granularity Control:** They provide a way to precisely control the granularity of calculations, enabling analysts to perform computations at exactly the right level of detail for their analysis needs.
- **Flexibility in Analysis:** LOD expressions offer flexibility in analyzing data at multiple levels of detail within the same visualization, facilitating deeper insights and more complex data stories.
- **Solving Complex Analytical Problems:** They are essential for solving analytical problems that require calculations beyond the current view's scope, such as calculating percentages of totals, differences from averages, and other sophisticated metrics.
  LOD expressions are a powerful feature in Tableau that expands the analytical capabilities of Tableau users, allowing for nuanced and complex data analysis beyond what's possible with simple aggregations and calculations.

46. Can you discuss the process of feature selection and its importance in data analysis?

Feature selection is the process of selecting a subset of relevant features from a larger set of variables or predictors in a dataset. It aims to improve model performance, reduce overfitting, enhance interpretability, and optimize computational efficiency.

Here's an overview of the process and its importance:
Importance of Feature Selection:

Improved Model Performance: By selecting the most relevant features, the model can focus on the most informative variables, leading to better predictive accuracy and generalization.

Overfitting Prevention: Including irrelevant or redundant features can lead to overfitting, where the model learns noise or specific patterns in the training data that do not generalize well to new data. Feature selection mitigates this risk.

Interpretability and Insights: A smaller set of selected features makes it easier to interpret and understand the model's results, facilitating insights and actionable conclusions.

Computational Efficiency: Working with a reduced set of features can significantly improve computational efficiency, especially when dealing with large datasets.

47. What are the different connection types in Tableau Software?

In Tableau, there are two primary types of data connections you can use to connect to your data sources: Live Connections and Extract Connections. Each type has its own set of advantages and disadvantages, depending on the requirements of your data analysis, such as the need for real-time data, data volume, and performance considerations.
Live Connection

- Description: A Live Connection establishes a direct link between Tableau and your data source. With a live connection, Tableau queries your data source directly whenever you interact with your dashboard or refresh your view. This means that your data is always up-to-date with the latest information from your data source.

- Advantages:
  - Real-Time Data: Since Tableau queries the data source directly, you always get the most current data available.
  - No Data Storage: There's no need to store data within Tableau, as it accesses your data source directly.
- Disadvantages:
  - Performance: Depending on the complexity of your queries and the capabilities of your data source, performance might be slower than with an extract, especially with large datasets.
  - Dependency on Data Source Availability: If the data source is down or experiencing issues, your Tableau visualizations might not be able to update.

Extract Connection

- Description: An Extract Connection involves importing a snapshot of the data from your data source into Tableau. This snapshot is saved as a Tableau Data Extract (TDE) or Hyper file. Tableau then queries this local copy of the data instead of the original data source.
- Advantages:
  - Performance: Extracts can significantly improve performance, especially with large datasets, as the data is optimized for quick aggregation and retrieval by Tableau.
  - Offline Access: Once the data is extracted into Tableau, you can access and analyze your data without needing a continuous connection to the original data source.
- Disadvantages:
  - Data Freshness: Since the data is a snapshot, it might not always reflect the most current state of your data source. You can schedule extracts to refresh at regular intervals, but real-time updates are not possible.
  - Data Volume: Extracts might become large and consume significant disk space, especially for very large datasets.

Choosing Between Live and Extract Connections

The choice between a live connection and an extract connection in Tableau depends on several factors:

- **Data Freshness Requirements:** If real-time data is crucial, a live connection might be necessary. If not, an extract could offer better performance.
- **Data Volume and Performance:** For large datasets, an extract might provide better performance, but consider the space required for storing the extract.
- **Data Source Availability and Reliability:** If your data source is not always available or reliable, an extract provides a stable alternative for analysis.
- **Update Frequency:** If the data does not change frequently, an extract might be sufficient and more efficient.
  In practice, many Tableau users leverage both connection types across different scenarios to balance performance, data freshness, and reliability according to their specific analytical needs.

48. What are the different joins that Tableau provides?

Tableau provides a variety of join types to combine data from multiple tables based on a common field, similar to SQL joins. These join types allow users to tailor how data is merged according to their specific analytical needs. The different join types offered by Tableau include:

1. **Inner Join:**
   - Combines rows from two or more tables where there is a match in the join condition for both tables. Rows that do not match the join condition in either table are not included in the result set.
2. **Left (Outer) Join:**
   - Returns all rows from the left table, and the matched rows from the right table. If there is no match, the result is NULL on the side of the right table.
3. **Right (Outer) Join:**
   - Returns all rows from the right table, and the matched rows from the left table. If there is no match, the result is NULL on the side of the left table.
4. **Full (Outer) Join:**

    - Combines rows from both tables, returning all rows from both the left and right tables, with matched rows from both sides where available. If there is no match, the missing side will contain NULL.
5. Cross Join (Cartesian Join):
    - Returns a Cartesian product of the two tables, meaning it joins every row of the left table with every row of the right table. This type of join does not require a join condition. Because of its nature, a cross join can result in a very large number of rows and should be used judiciously.

Each of these join types can be used to address different data relationship needs:

- Inner joins are used when you only want to combine rows that have matching values in both tables.
- Left and right joins are useful when you want to include all rows from one side of the join, regardless of whether they have matching rows in the other table.
- Full joins are helpful when you need a complete picture of both tables, including rows that do not have matches in the opposite table.
- Cross joins are used when you need to combine every possible pair of rows from the two tables, often for combinatorial or exhaustive pairing analyses.

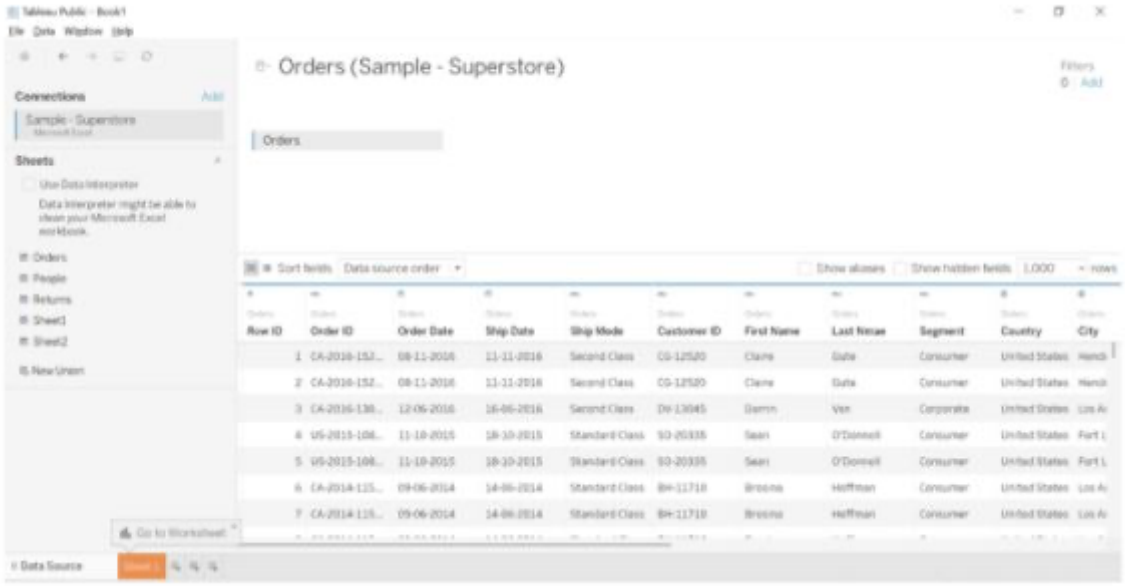
Tableau's drag-and-drop interface simplifies the process of creating these joins, allowing users to visually specify how tables should be combined without writing SQL code. This enables both technical and non-technical users to effectively merge and analyze data from multiple tables.

49. What is a Gantt Chart in Tableau?

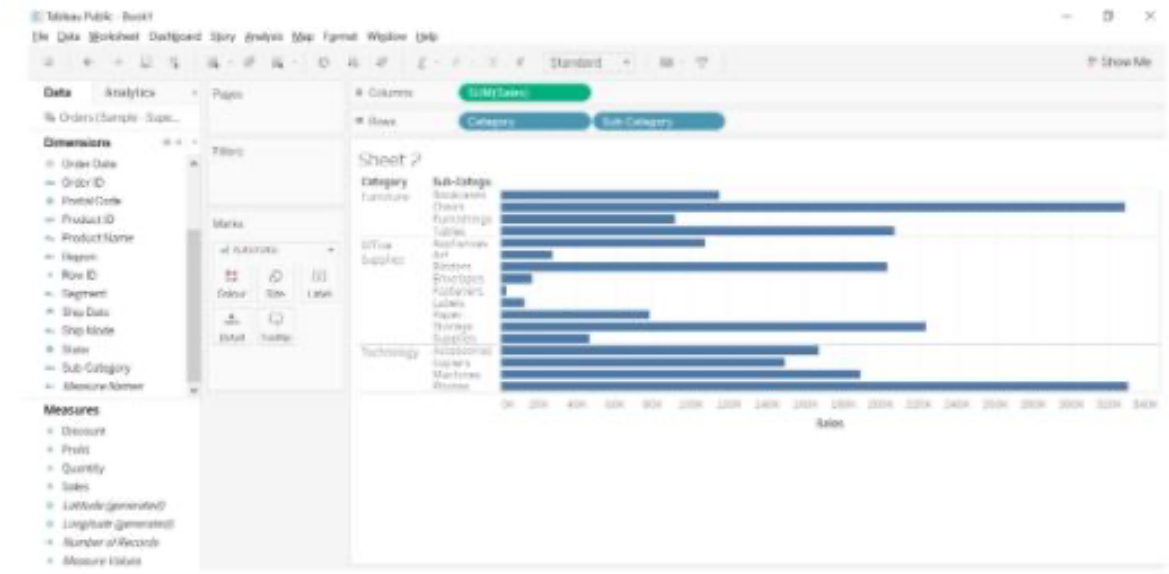
A Gantt chart in Tableau depicts the progress of value over the period, i.e., it shows the duration of events. It consists of bars along with the time axis. The Gantt chart is mostly used as a project management tool where each bar is a measure of a task in the project.

50. Using the Sample Superstore dataset, create a view in Tableau to analyze the sales, profit, and quantity sold across different subcategories of items present under each category.

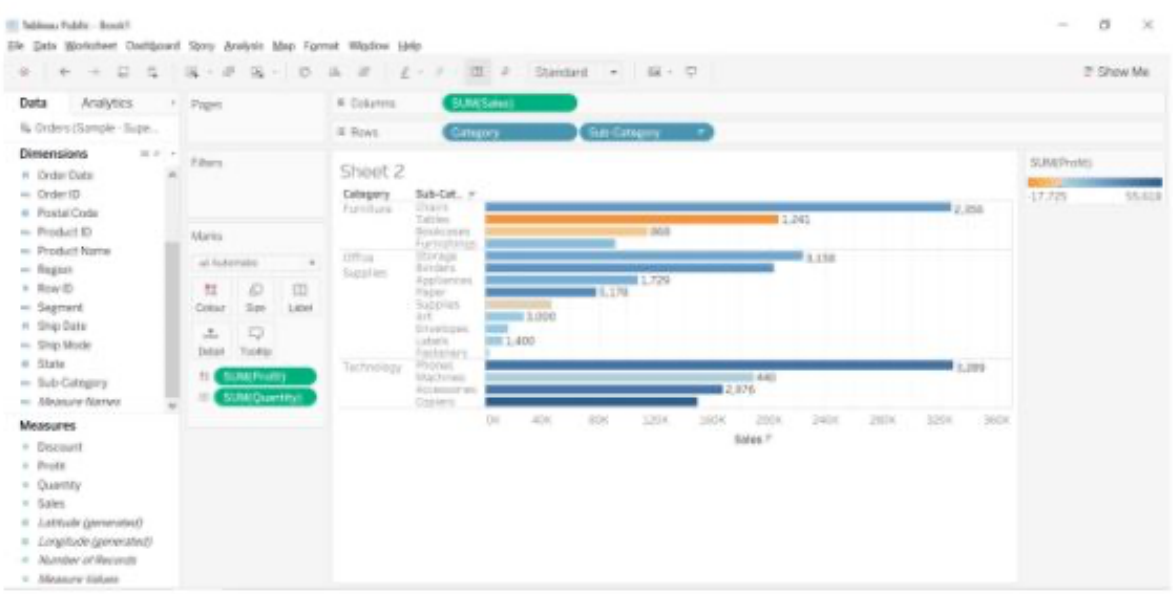
- Load the Sample - Superstore dataset



- Drag Category and Subcategory columns into Rows, and Sales on to Columns. It will result in a horizontal bar chart.

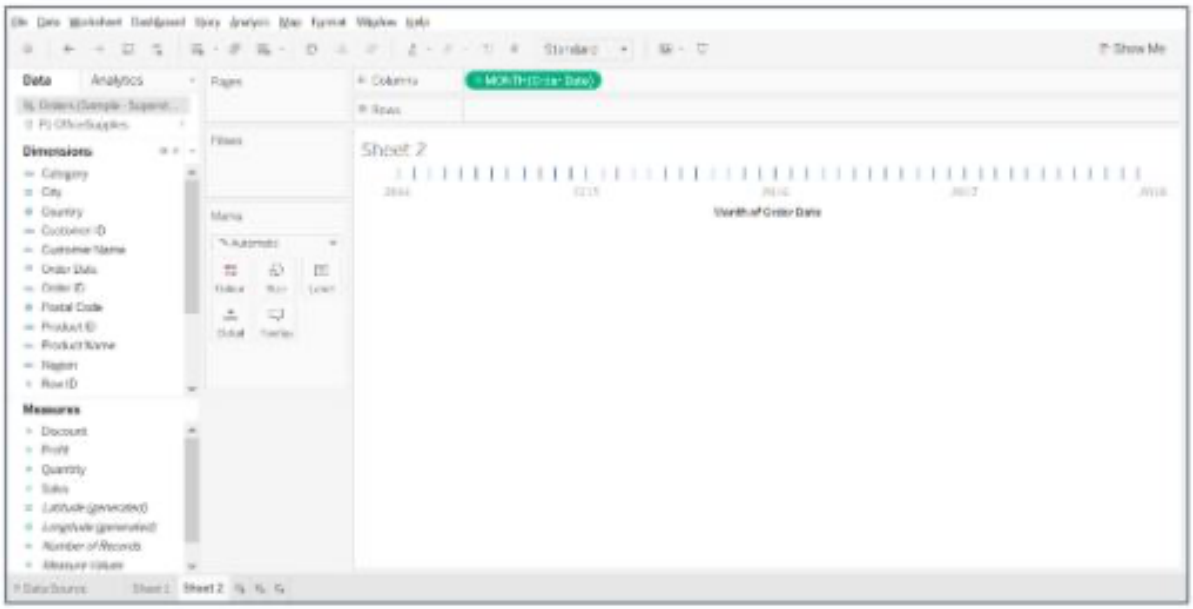


- Drag Profit on to Colour, and Quantity on to Label. Sort the Sales axis in descending order of the sum of sales within each sub-category.

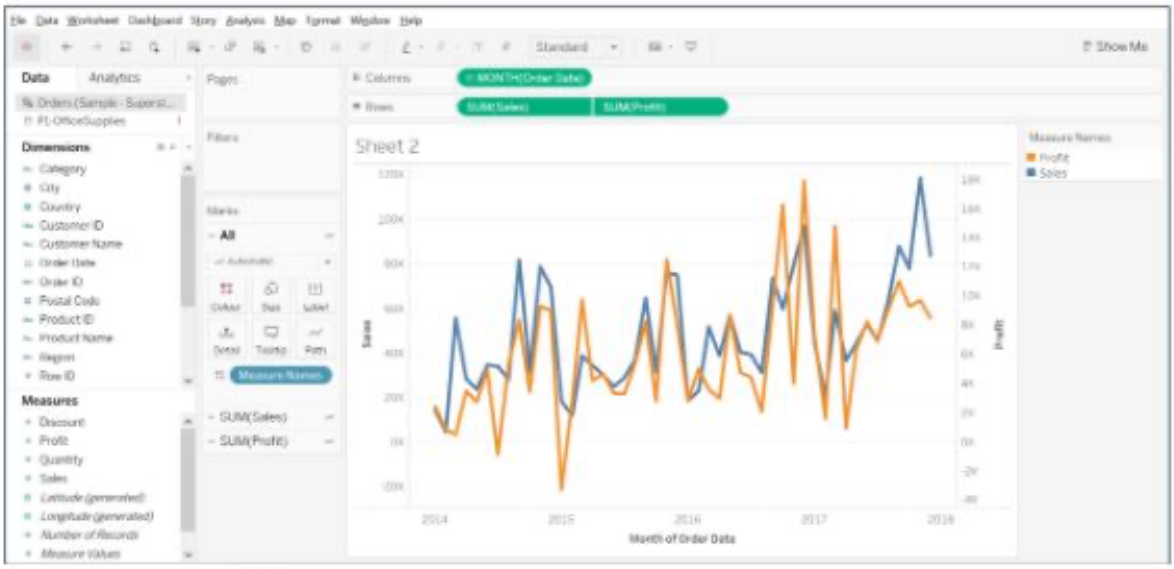


51. Create a dual-axis chart in Tableau to present Sales and Profit across different years using the Sample Superstore dataset.

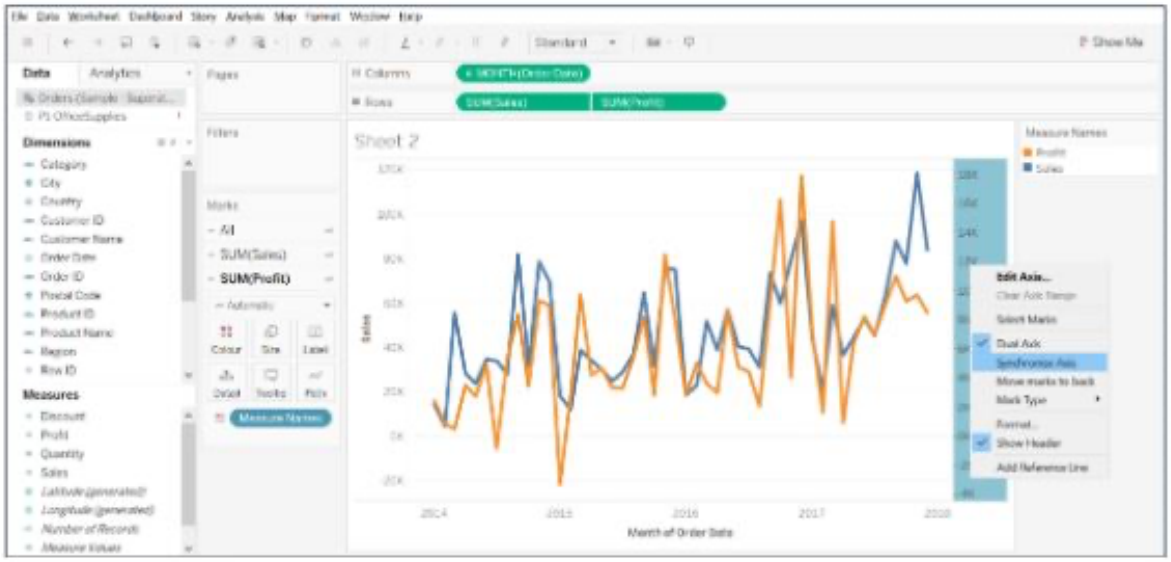
- Drag the Order Date field from Dimensions on to Columns, and convert it into continuous Month.



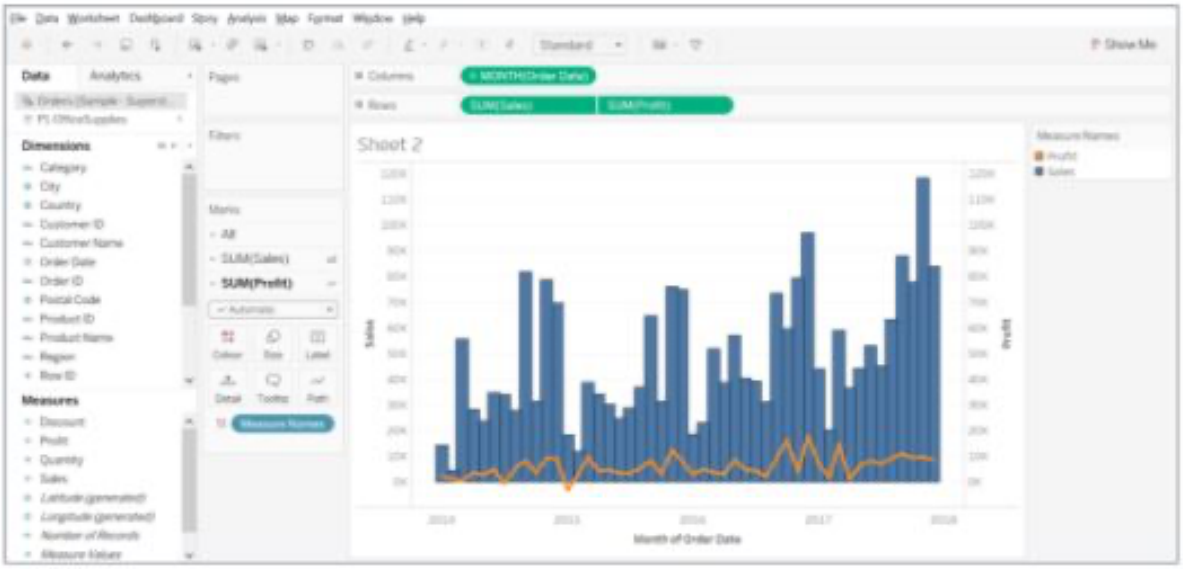
- Drag Sales on to Rows, and Profits to the right corner of the view until you see a light green rectangle.



- Synchronize the right axis by right-clicking on the profit axis.

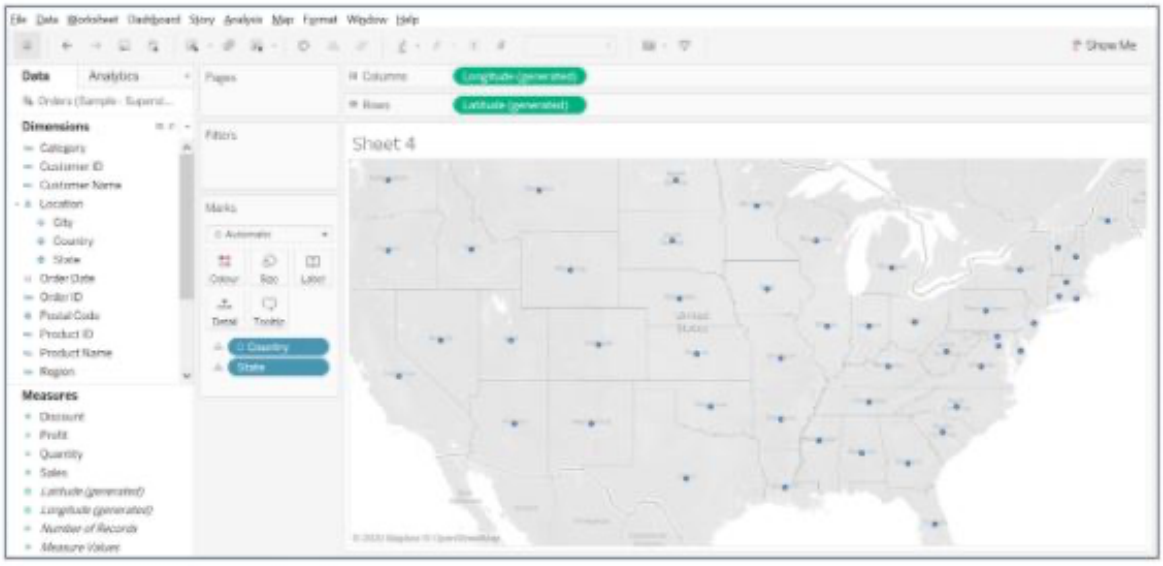


- Under the Marks card, change SUM(Sales) to Bar and SUM(Profit) to Line and adjust the size.

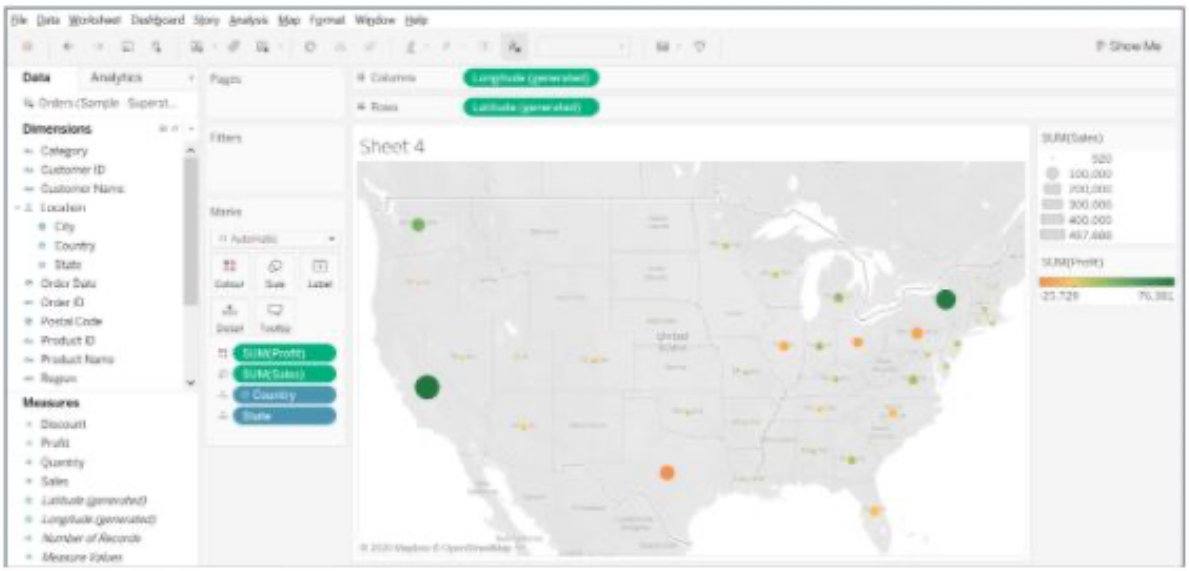


52. Design a view in Tableau to show State-wise Sales and Profit using the Sample Superstore dataset.

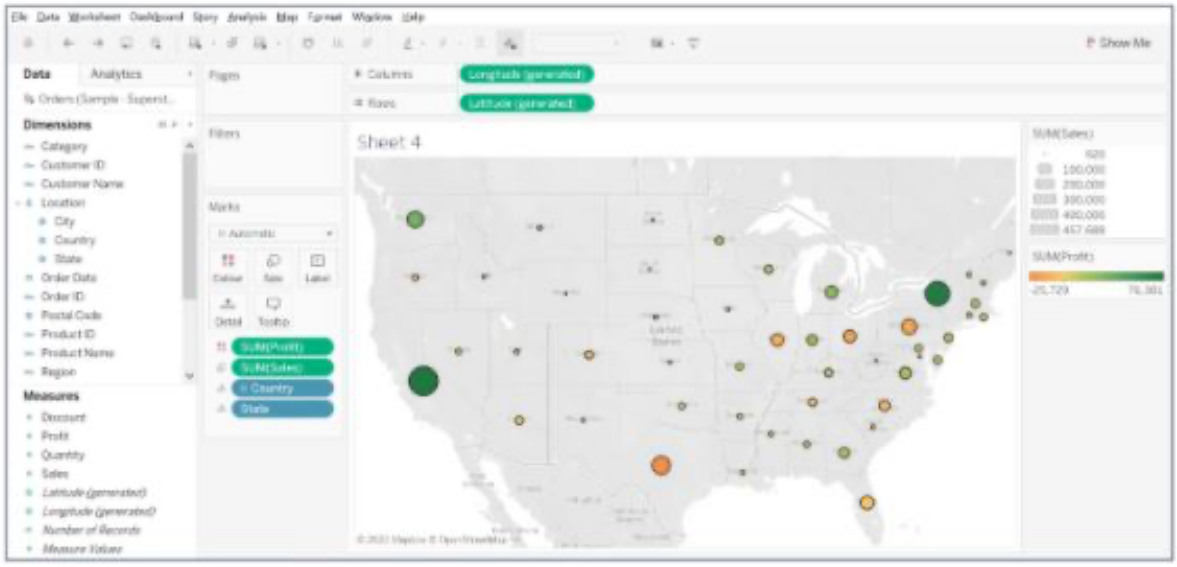
- Drag the Country field on to the view section and expand it to see the States.



- Drag the Sales field on to Size, and Profit on to Colour.

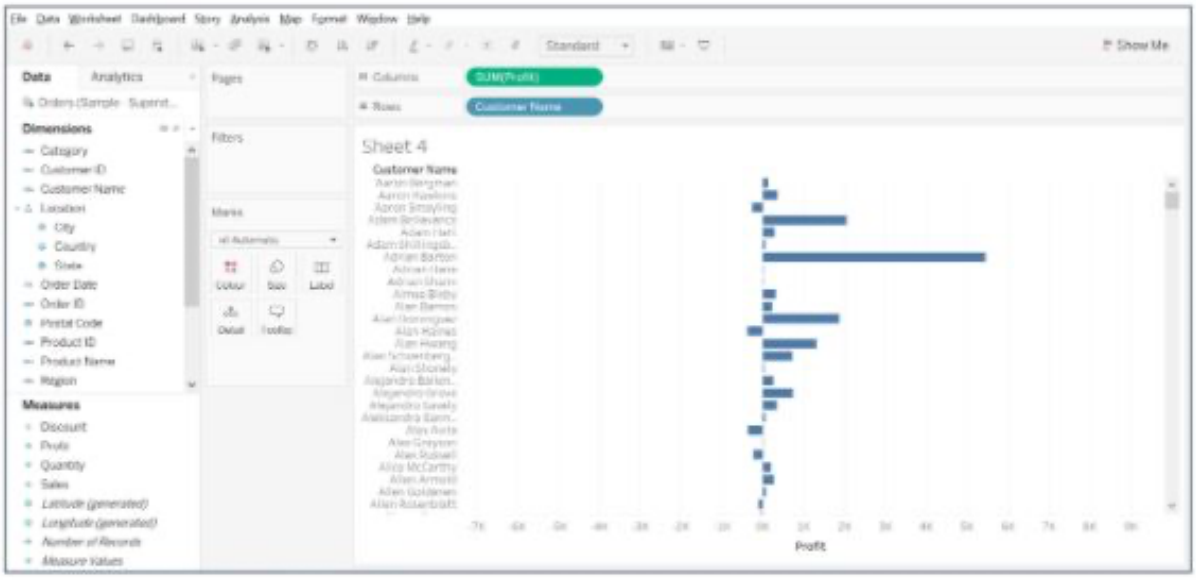


- Increase the size of the bubbles, add a border, and halo color.

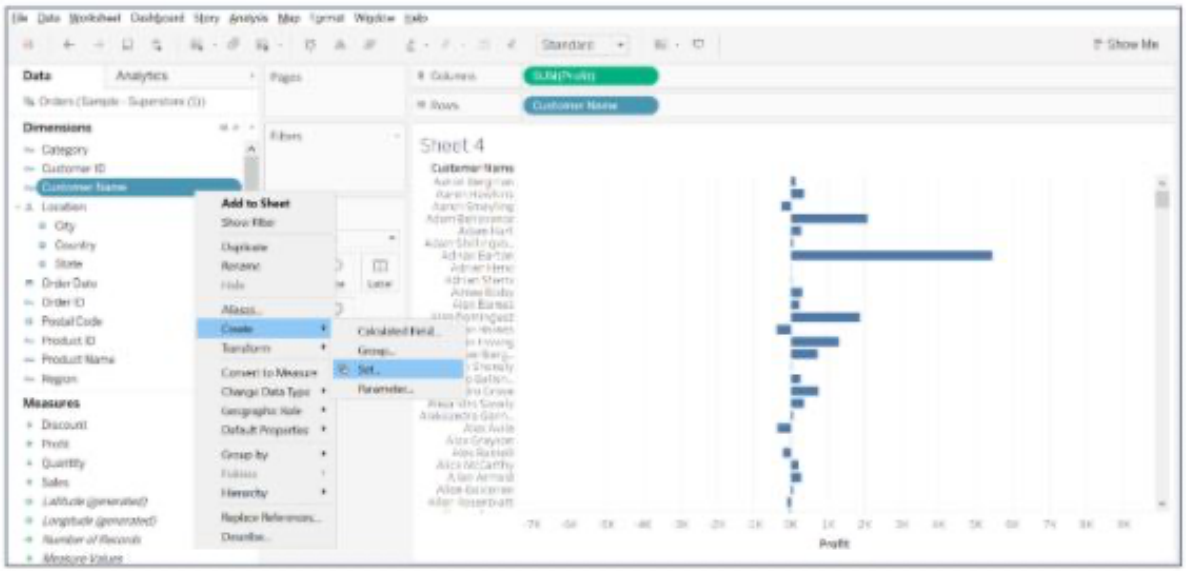


From the above map, it is clear that states like Washington, California, and New York have the highest sales and profits. While Texas, Pennsylvania, and Ohio have good amounts of sales but the least profits.

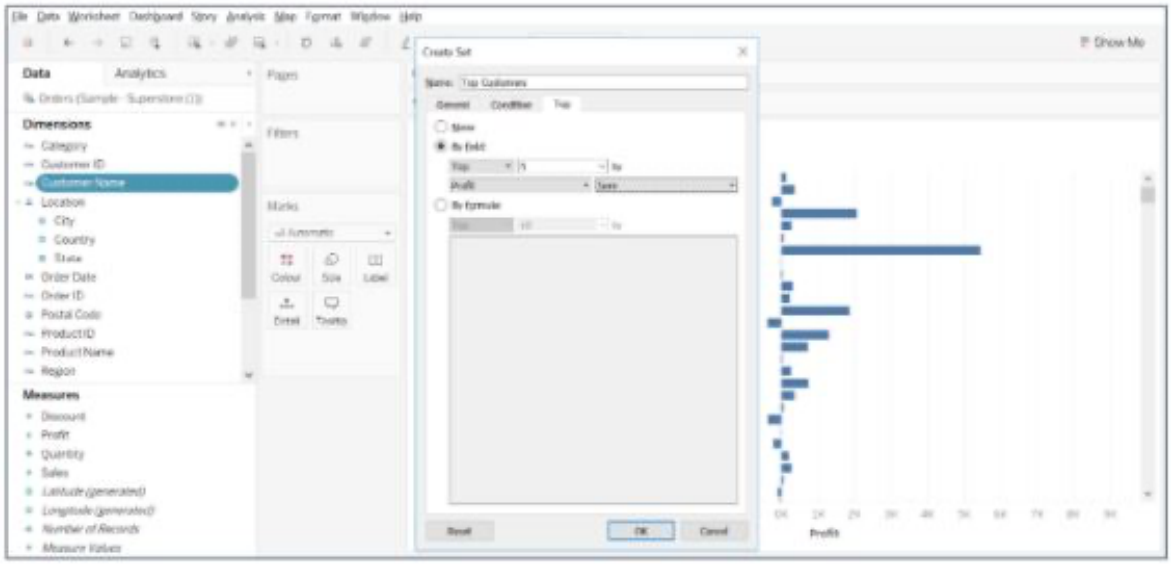
54. Using the Sample Superstore dataset, display the top 5 and bottom 5 customers based on their profit.



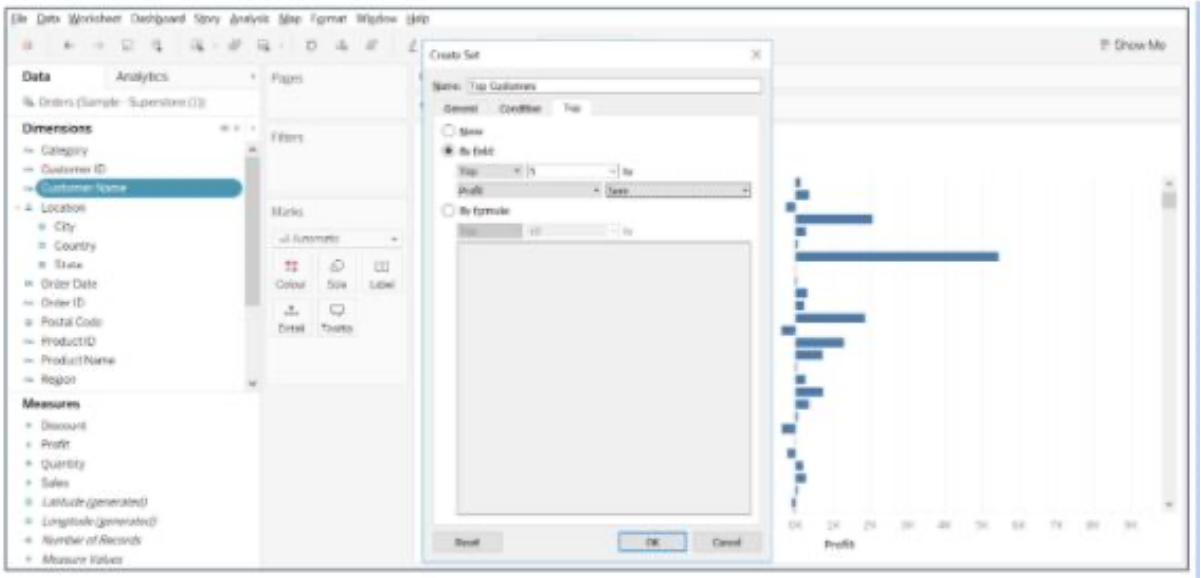
- Drag Customer Name field on to Rows, and Profit on to Columns.



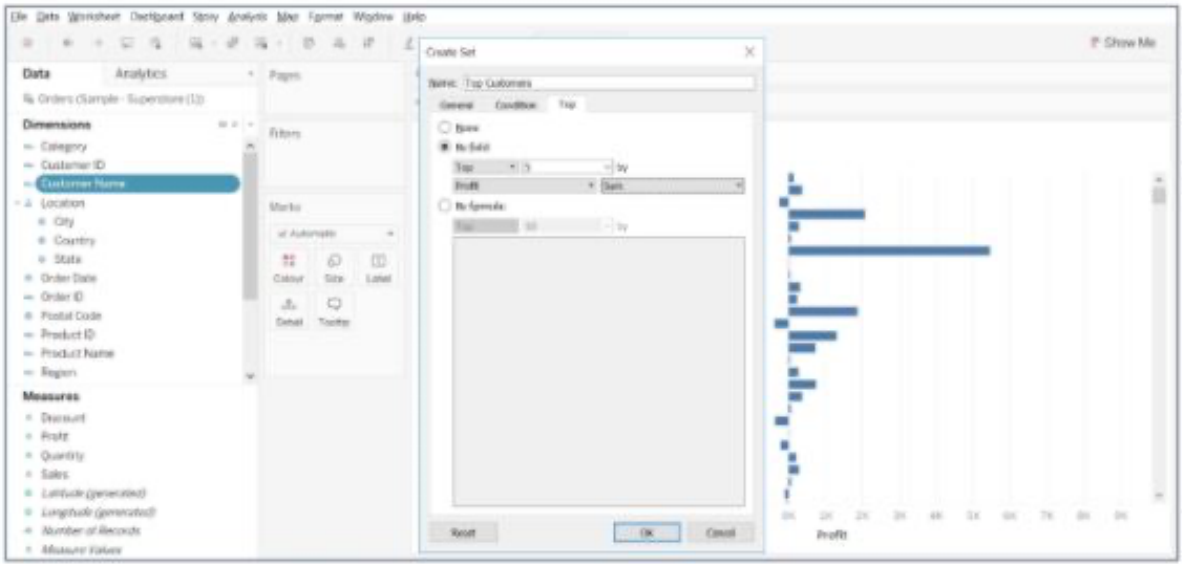
- Right-click on the Customer Name column to create a set



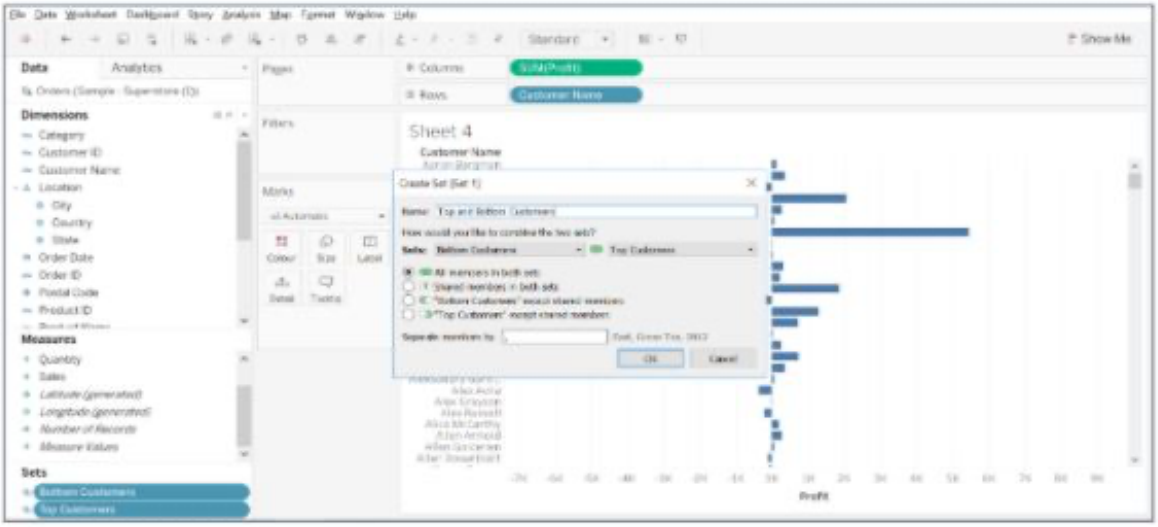
- Give a name to the set and select the top tab to choose the top 5 customers by sum(profit)



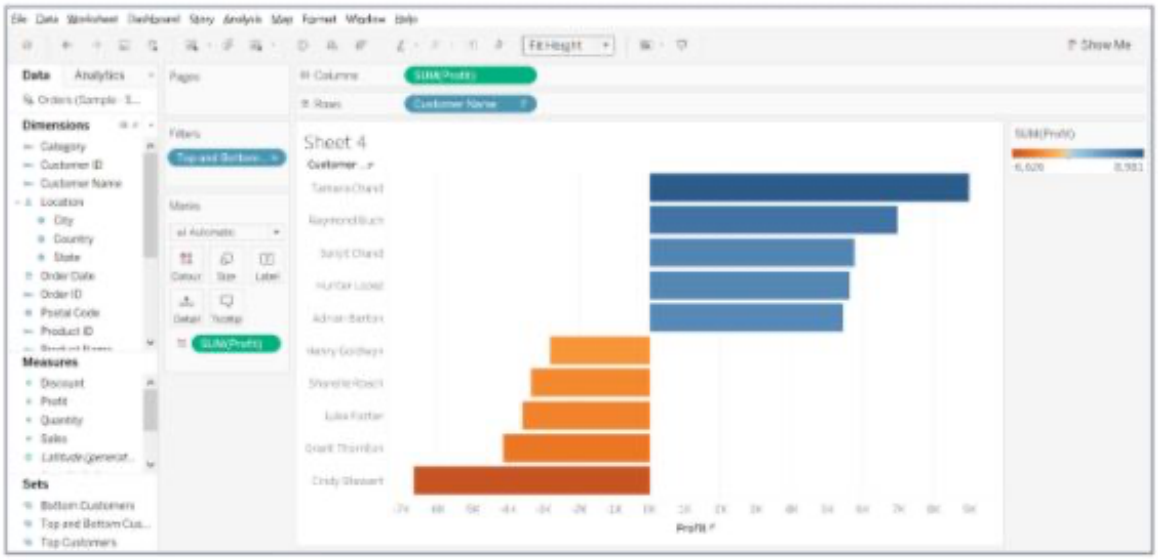
- Similarly, create a set for the bottom five customers by sum(profit)



- Select both the sets, right-click to create a combined set. Give a name to the set and choose All members in both sets.

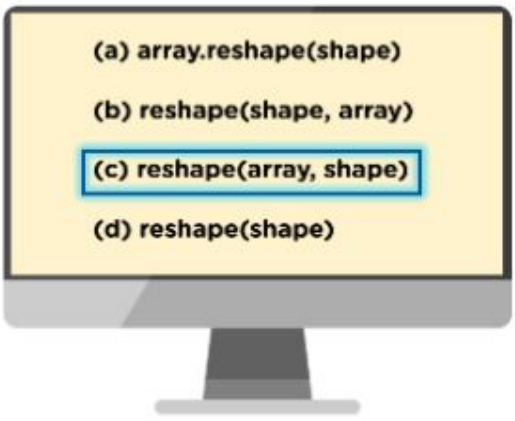


- Drag top and bottom customers set on to Filters, and Profit field on to Colour to get the desired result.



Data Analyst Interview Questions On Python

55. What is the correct syntax for reshape() function in NumPy?



Example

```
import numpy as np

a = np.array([[1,2,3,4,5],[1,2,3,4,5]])
np.reshape(a, (2,5))
```

```
array([[1, 2, 3, 4, 5],
       [1, 2, 3, 4, 5]])
```

56. What are the different ways to create a data frame in Pandas?

There are two ways to create a Pandas data frame.

- By initializing a list

```
import pandas as pd

# Initialize list of lists
data = [['tom', 30], ['Jerry', 20], ['Angela', 35]]

# Create the DataFrame
df = pd.DataFrame(data, columns = ['Name', 'Age'])
```

```
df
```

	Name	Age
0	tom	30
1	Jerry	20
2	Angela	35

- By initializing a dictionary

```
import pandas as pd

# Intialise data of lists.
data = {'Name':['Tom', 'Jerry', 'Angela', 'Mary'], 'Age':[20, 21, 19, 18]}

# Create the DataFrame
df = pd.DataFrame(data)
```

```
# Print the output.
df
```

	Name	Age
0	Tom	20
1	Jerry	21
2	Angela	19
3	Mary	18

57. Write the Python code to create an employee's data frame from the "emp.csv" file and display the head and summary.

To create a DataFrame in Python, you need to import the Pandas library and use the read_csv function to load the .csv file.

Give the right location where the file name and its extension follow the dataset.

```
import pandas as pd

employees = pd.read_csv("D:/Chrome Downloads/human-resources-data-set/emp.csv")
```

To display the head of the dataset, use the head() function.

```
employees.head()
```

	Employee_Name	EmpID	MarriedID	MaritalStatusID	GenderID	EmpStatusID	DeptID	PerfScoreID	FromDiversityJobFairID	PayRate	...	Departm
0	Brown, Mia	1.103024e+09	1.0	1.0	0.0	1.0	1.0	3.0	1.0	28.50	...	Adr Offi
1	LaRotonda, William	1.106027e+09	0.0	2.0	1.0	1.0	1.0	3.0	0.0	23.00	...	Adr Offi
2	Steans, Tyrone	1.302053e+09	0.0	0.0	1.0	1.0	1.0	3.0	0.0	29.00	...	Adr Offi
3	Howard, Estelle	1.211051e+09	1.0	1.0	0.0	1.0	1.0	3.0	0.0	21.50	...	Adr Offi
4	Singh, Nan	1.307060e+09	0.0	0.0	0.0	1.0	1.0	3.0	0.0	16.56	...	Adr Offi

5 rows × 35 columns

The 'describe' method is used to return the summary statistics in Python.

```
employees.describe
```

```
<bound method NDFrame.describe of           Employee_Name        EmpID  MarriedID  MaritalStatusID  GenderID  \
0             Brown, Mia  1.103024e+09        1.0              1.0       0.0
1    LaRotonda, William   1.106027e+09        0.0              2.0       1.0
2        Steans, Tyrone   1.302053e+09        0.0              0.0       1.0
3         Howard, Estelle  1.211051e+09        1.0              1.0       0.0
4             Singh, Nan   1.307060e+09        0.0              0.0       0.0
..                   ...           ...        ...              ...       ...
396                  NaN           NaN        NaN              NaN       NaN
397                  NaN           NaN        NaN              NaN       NaN
398                  NaN           NaN        NaN              NaN       NaN
399                  NaN           NaN        NaN              NaN       NaN
400                  NaN           NaN        NaN              NaN       NaN

     EmpStatusID  DeptID  PerfScoreID  FromDiversityJobFairID  PayRate  ...  \
0            1.0     1.0          3.0                     1.0    28.50  ...
1            1.0     1.0          3.0                     0.0    23.00  ...
2            1.0     1.0          3.0                     0.0    29.00  ...
3            1.0     1.0          3.0                     0.0    21.50  ...
4            1.0     1.0          3.0                     0.0    16.56  ...
..           ...     ...          ...                     ...      ...  ...
396          NaN     NaN          NaN                     NaN      NaN  ...
397          NaN     NaN          NaN                     NaN      NaN  ...
398          NaN     NaN          NaN                     NaN      NaN  ...
399          NaN     NaN          NaN                     NaN      NaN  ...
400          NaN     NaN          NaN                     NaN      NaN  ...
```

58. How will you select the Department and Age columns from an Employee data frame?

```
# Print the output.
Employees
```

	Name	Age	Department
0	Nick	30	Manufacturing
1	Ricky	42	IT
2	Mathew	45	Marketing
3	Andrew	35	Sales

You can use the column names to extract the desired columns.

```
Employees[['Department', 'Age']]
```

	Department	Age
0	Manufacturing	30
1	IT	42
2	Marketing	45
3	Sales	35

59. Suppose there is an array, what would you do? num = np.array([[1,2,3],[4,5,6],[7,8,9]]). Extract the value 8 using 2D indexing.

```
import numpy as np

num = np.array([[1,2,3],[4,5,6],[7,8,9]])
print(num)
```

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
```

Since the value eight is present in the 2nd row of the 1st column, we use the same index positions and pass it to the array.

```
num[2,1]
```

```
8
```

60. Suppose there is an array that has values [0,1,2,3,4,5,6,7,8,9]. How will you display the following values from the array - [1,3,5,7,9]?

```
import numpy as np

arr = np.arange(10)
arr
```

```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

Since we only want the odd number from 0 to 9, you can perform the modulus operation and check if the remainder is equal to 1.

```
arr[arr % 2 == 1]
```

```
array([1, 3, 5, 7, 9])
```

61. There are two arrays, ‘a’ and ‘b’. Stack the arrays a and b horizontally using the NumPy library in Python.

```
a = np.arange(10).reshape(2,-1)
a
```

```
array([[0, 1, 2, 3, 4],
       [5, 6, 7, 8, 9]])
```

```
b = np.repeat(1, 10). reshape(2, -1)
b
```

```
array([[1, 1, 1, 1, 1],
       [1, 1, 1, 1, 1]])
```

You can either use the concatenate() or the hstack() function to stack the arrays.

Method 1:
Using concatenate function

```
np.concatenate([a, b], axis=1)
```

```
array([[0, 1, 2, 3, 4, 1, 1, 1, 1, 1],
       [5, 6, 7, 8, 9, 1, 1, 1, 1, 1]])
```

Method 2:
Using hstack function

```
np.hstack([a, b])
```

```
array([[0, 1, 2, 3, 4, 1, 1, 1, 1, 1],
       [5, 6, 7, 8, 9, 1, 1, 1, 1, 1]])
```

62. How can you add a column to a Pandas Data Frame?

Suppose there is an emp data frame that has information about a few employees.
Let's add an Address column to that data frame.

```
emp = {'Name': ['Sam', 'Prince', 'Tom', 'Andy'],
       'Height': [5.1, 6.2, 6.9, 7.2],
       'Qualification': ['Msc', 'MA', 'Msc', 'MBA']}
```

```
emp
```

```
{'Name': ['Sam', 'Prince', 'Tom', 'Andy'],
 'Height': [5.1, 6.2, 6.9, 7.2],
 'Qualification': ['Msc', 'MA', 'Msc', 'MBA']}
```

```
df = pd.DataFrame(emp)
```

```
df
```

	Name	Height	Qualification
0	Sam	5.1	Msc
1	Prince	6.2	MA
2	Tom	6.9	Msc
3	Andy	7.2	MBA

Declare a list of values that will be converted into an address column.

```
address = ['New York', 'California', 'Boston', 'Washington']
```

```
df['Address'] = address
```

```
df
```

	Name	Height	Qualification	Address
0	Sam	5.1	Msc	New York
1	Prince	6.2	MA	California
2	Tom	6.9	Msc	Boston
3	Andy	7.2	MBA	Washington

63. How will you print four random integers between 1 and 15 using NumPy?
To generate Random numbers using NumPy, we use the random.randint() function.

```
import numpy as np
rand_arr = np.random.randint(1,15,4)
print('\n Random numbers from 1 to 15 are ',rand_arr)
```

```
 Random numbers from 1 to 15 are  [ 7 11  3  9]
```

64. From the below DataFrame, how will you find each column's unique values and subset the data for Age<35 and Height>6?

df

	Name	Height	Age
0	Sam	5.9	30
1	Prince	6.8	45
2	Tom	6.1	25
3	Andy	7.1	69
4	Harry	6.2	51
5	Angela	5.9	25
6	Lucy	6.5	30

To find the unique values and number of unique elements, use the unique() and nunique() function.

```
# For finding the unique elements for each column
df['Height'].unique()
```

```
array([5.9, 6.8, 6.1, 7.1, 6.2, 6.5])
```

```
df['Age'].unique()
```

```
array([30, 45, 25, 69, 51], dtype=int64)
```

```
# To find the number of unique elements
df['Age'].nunique()
```

```
5
```

```
df['Height'].nunique()
```

```
6
```

Now, subset the data for Age<35 and Height>6.

```
# To subset the dataframe
new_df = df[(df['Age']<35) & (df['Height']>6)]
```

```
new_df
```

	Name	Height	Age
2	Tom	6.1	25
6	Lucy	6.5	30

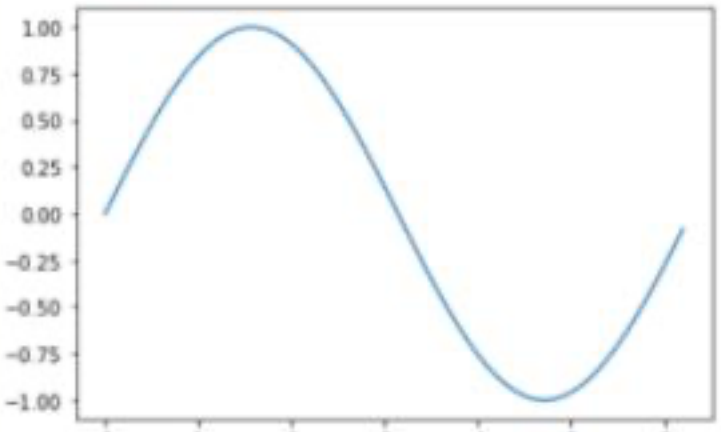
65. Plot a sine graph using NumPy and Matplotlib library in Python.

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
x= np.arange(0,2*np.pi,0.1)
y=np.sin(x)
print(x)
```

```
[0.  0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.  1.1 1.2 1.3 1.4 1.5 1.6 1.7
 1.8 1.9 2.  2.1 2.2 2.3 2.4 2.5 2.6 2.7 2.8 2.9 3.  3.1 3.2 3.3 3.4 3.5
 3.6 3.7 3.8 3.9 4.  4.1 4.2 4.3 4.4 4.5 4.6 4.7 4.8 4.9 5.  5.1 5.2 5.3
 5.4 5.5 5.6 5.7 5.8 5.9 6.  6.1 6.2]
```

Below is the result sine graph.

```
plt.plot(x,y)
plt.show()
```



66. Using the below Pandas data frame, find the company with the highest average sales. Derive the summary statistics for the sales column and transpose the statistics.

```
df = pd.DataFrame(data)
df
```

	Company	Person	Sales
0	HP	Richard	2000
1	HP	Angela	1200
2	DELL	Mary	3400
3	DELL	Rick	1245
4	FB	Julia	2430
5	FB	Kevin	3500

- Group the company column and use the mean function to find the average sales

```
by_comp = df.groupby("Company")
```

```
by_comp.mean()
```

Company	Sales
DELL	2322.5
FB	2965.0
HP	1600.0

- Use the describe() function to find the summary statistics

```
by_comp.describe()
```

	Sales							
	count	mean	std	min	25%	50%	75%	max
Company								
DELL	2.0	2322.5	1523.815113	1245.0	1783.75	2322.5	2861.25	3400.0
FB	2.0	2965.0	756.604256	2430.0	2697.50	2965.0	3232.50	3500.0
HP	2.0	1600.0	565.685425	1200.0	1400.00	1600.0	1800.00	2000.0

- Apply the transpose() function over the describe() method to transpose the statistics

```
by_comp.describe().transpose()
```

	Company	DELL	FB	HP
Sales	count	2.000000	2.000000	2.000000
	mean	2322.500000	2965.000000	1600.000000
	std	1523.815113	756.604256	565.685425
	min	1245.000000	2430.000000	1200.000000
	25%	1783.750000	2697.500000	1400.000000
	50%	2322.500000	2965.000000	1600.000000
	75%	2861.250000	3232.500000	1800.000000
	max	3400.000000	3500.000000	2000.000000