

Room for Improvement :Data Stories in Hotel Operations

By

C. Mahananda Reddy(Lead)

L. Poojitha

K. Yagnya priya

M.Sanskruthi

A. Manoj

Table of Contents

Introduction

- Problem Statement
- Key Objectives
- Dataset Description
- Overview of Data Analytics and Visualization
- Data Visualisation Definition , Advantages and Disadvantages
- Importance of Power BI and Python in Data Analysis
- Compare Power BI Visualization and Python Visualization

Getting Started

- Setting Up Power BI
- Installing Python and Necessary Libraries

Data Analysis Techniques

- Descriptive Statistics
- Building Queries and Data Models

Data Visualization with Power BI

- Data Preparation
- Data Transformation
- Data Cleaning
- Data Modeling
- DAX
 - Aggregated Columns
 - Key Measures

- Data Visualization

Data Visualization with Python

- Seaborn for Statistical Visualization
- Interactive Visualization with Plotly

Summary of Key Concepts

Future Trends in Data Analytics and Visualization

Challenges

Challenges in Python:

Static Visualizations: Python visualizations are typically static images and cannot be easily shared with others who don't have Python installed

Installation and Setup: Setting up the Python environment and installing necessary libraries (e.g., Pandas, NumPy, Matplotlib) can be cumbersome and time-consuming.

Handling Large Datasets: Python can struggle with performance issues when handling large datasets, making it less efficient for big data analytics compared to other tools.

Limited Built-in Visualizations: While Python offers extensive customization through libraries, the out-of-the-box visualization options are limited compared to dedicated visualization tools

Dependency Management: Managing dependencies and ensuring compatibility between different libraries can be complex, especially in larger projects where multiple packages are used. Conflicts between library versions can cause issues and require careful handling with tools like virtual environments.

Challenges in PowerBI:

Performance Issues: Users often experience slow performance with the Power BI desktop app, especially when handling large datasets. It may take longer to process data or produce reports than other data analysis tools, depending on the size of the data set .

Poor Data Quality: Data quality is one of the most important aspects of a data analysis and is often overlooked or treated as an afterthought. Poor data quality, missing values, or outliers can affect the accuracy of visualizations. Ensuring clean and reliable data is crucial.

Learning Curve for DAX: Learning and effectively using DAX (Data Analysis Expressions) for data modeling and transformation can be difficult for new users.

Integration Issues: Integrating Power BI with various data sources and ensuring seamless data flow can sometimes be problematic and require additional configuration.

Query Editor Limitations: One significant disadvantage of Power BI is the complexity and sluggishness of its query editor. Users often struggle with delayed responses and a lack of clarity in displaying queries.

\

Problem Statement:

In an increasingly competitive retail landscape, optimising inventory management and sales strategies is crucial for sustained success. This project focuses on leveraging advanced data analytics techniques to forecast future sales trends accurately and derive actionable insights for decision-making. By combining Python for forecasting and Power BI for visualisation, the aim is to empower the retail company with the tools necessary to make informed decisions, minimize stockouts, reduce excess inventory, and ultimately enhance profitability.

Key Objectives:

1. Forecasting Future Sales Trends:

- Implement algorithms such as ARIMA, Exponential Smoothing, and Prophet to predict sales volumes with high accuracy.
- Account for seasonality, trends, and external factors influencing sales fluctuations.

2. Uncovering Actionable Insights with Power BI:

- Integrate forecasted sales data and other relevant datasets into Power BI for interactive visualization and analysis.
- Design intuitive dashboards and reports to identify sales trends, product performance, and customer behavior patterns.
- Extract actionable insights to inform pricing strategies, promotions, and marketing campaigns.

3. Driving Business Growth and Profitability:

- Translate data-driven insights into actionable business decisions to drive revenue growth and maximize profitability.
- Continuously monitor and evaluate the effectiveness of implemented strategies, iterating and optimizing as needed.

4. Ensuring Scalability and Sustainability:

- Develop scalable data infrastructure and automated workflows to support ongoing forecasting and analysis.
- Provide training and support to empower stakeholders to effectively leverage data analytics tools and insights.

By addressing these objectives, the retail chain aims to stay ahead of market dynamics, enhance customer satisfaction, and solidify its position as an industry leader through data-driven innovation and strategic decision-making.

Dataset Description

we have provided 5 CSV files in our dataset:

1. dim_date
2. dim_hotels
3. dim_rooms
4. fact_aggregated_bookings
5. Fact_bookings

Column Description for dim_date.csv:

1. **date:** This column represents the dates present in May, June and July.
2. **mmm yy:** This column represents the date in the format of mmm yy (monthname year).
3. **week no:** This column represents the unique week number for that particular date.
4. **day_type:** This column represents whether the given day is Weekend or Weekday.

Column Description for dim_hotels.csv:

1. **property_id:** This column represents the Unique ID for each of the hotels.
2. **property_name:** This column represents the name of each hotel.
3. **category:** This column determines which class[Luxury, Business] a particular hotel/property belongs to.
4. **city:** This column represents where the particular hotel/property resides in.

Column Description for dim_rooms.csv:

1. **room_id:** This column represents the type of room[RT1, RT2, RT3, RT4] in a hotel.
2. **room_class:** This column represents to which class[Standard, Elite, Premium, Presidential] particular room type belongs.

Column Description for fact_aggregated_bookings.csv :

- 1. property_id:** This column represents the Unique ID for each of the hotels.
- 2. check_in_date:** This column represents all the check_in_dates of the customers.
- 3. room_category:** This column represents the type of room[RT1, RT2, RT3, RT4] in a hotel.
- 4. successful_bookings:** This column represents all the successful room bookings that happen for a particular room type in that hotel on that particular date.
- 5. capacity:** This column represents the maximum count of rooms available for a particular room type in that hotel on that particular date.

Column Description for fact_bookings:

- 1. booking_id:** This column represents the Unique Booking ID for each customer when they booked their rooms.
- 2. property_id:** This column represents the Unique ID for each of the hotels
- 3. booking_date:** This column represents the date on which the customer booked their rooms.
- 4. check_in_date:** This column represents the date on which the customer check-in(entered) at the hotel.
- 5. check_out_date:** This column represents the date on which the customer check-out(left) of the hotel.
- 6. no_guests:** This column represents the number of guests who stayed in a particular room in that hotel.
- 7. room_category:** This column represents the type of room[RT1, RT2, RT3, RT4] in a hotel.
- 8. booking_platform:** This column represents in which way the customer booked his room.
- 9. ratings_given:** This column represents the ratings given by the customer for hotel services.
- 10. booking_status:** This column represents whether the customer cancelled his booking[Canceled], successfully stayed in the hotel[Checked Out] or booked his room but not stayed in the hotel[No show].
- 11. revenue_generated:** This column represents the amount of money generated by the hotel from a particular customer.

12. revenue_realized: This column represents the final amount of money that goes to the hotel based on booking status. If the booking status is cancelled, then 40% of the revenue generated is deducted and the remaining is refunded to the customer. If the booking status is Checked Out/No show, then full revenue generated will go to hotels.

1.Introduction

Overview of Data Analytics and Visualization:

Data analytics and visualization are crucial components in understanding patterns, trends, and insights hidden within data. They enable organizations to gain valuable insights from complex data sets, making informed decisions easier and more efficient.

- **Importance of Data Analytics**

Data analytics is the process of examining and interpreting data to extract meaningful information. It involves various techniques, such as statistical modeling, machine learning, and data mining, to identify patterns, trends, and correlations within the data. The importance of data analytics lies in its ability to:

1. **Identify Patterns and Trends:** Data analytics helps in identifying patterns and trends that might not be immediately apparent from raw data. This enables businesses to anticipate future outcomes and make informed decisions.

2. **Improve Decision-Making:** By providing actionable insights, data analytics supports better decision-making. It helps stakeholders to make informed choices based on data-driven insights rather than relying on intuition or anecdotal evidence.
3. **Optimise Operations:** Data analytics can be used to optimize business operations by identifying areas of inefficiency and suggesting improvements. This leads to increased productivity and cost savings.

- **Importance of Data Visualization**

Data visualisation is the process of presenting data in a visual format, such as charts, graphs, maps, and infographics, to facilitate understanding and communication. The importance of data visualisation lies in its ability to:

1. **Communicate Insights:** Data visualisation helps in communicating complex data insights to both technical and non-technical stakeholders in an easily understandable format. This facilitates collaboration and decision-making across different levels of an organisation.
2. **Facilitate Storytelling:** Data visualisation enables the creation of compelling narratives around data, making it easier to convey insights and trends to stakeholders. This storytelling aspect is critical in data science, as it helps in presenting complex data in a way that resonates with the audience.
3. **Enhance Exploration:** Data visualisation allows for interactive exploration of data, enabling users to focus on specific aspects of the data and gain a deeper understanding of the insights presented.
4. **Improve Decision-Making:** By providing a quick and effective way to communicate information, data visualisation supports faster decision-making. It helps stakeholders to identify trends and patterns more quickly, leading to more informed and timely decisions.

Definition:

Data visualisation is the process of transforming raw data into a visual format, such as charts, graphs, maps, and infographics, to facilitate understanding and communication. It involves presenting complex data in a way that is easily digestible and actionable for various stakeholders.

- **Advantages of Data Visualization**

1. **Better Agreement:** Data visualisation helps in comparing the performances of two elements or scenarios by presenting the data in a pictorial form, making it easier to understand and analyse.
2. **Actionable Insights:** Visualization tools make data more accessible and understandable, allowing a broad spectrum of personnel to quickly absorb information and make informed decisions.

3. **Exploration of Complex Relationships:** Advanced visualisation platforms can display complex relationships among data points and metrics, enabling faster data-based decision-making.
4. **Compelling Storytelling:** Data dashboards that are visually compelling maintain audience interest and facilitate the communication of complex data insights.
5. **Accessibility:** Visualisation tools make data more accessible and understandable, even for non-technical users.
6. **Interactivity:** Interactive dashboards allow users to explore data in more detail, enhancing the understanding of complex data.
7. **Simplified Communication:** Data visualisation simplifies complex data by expressing it visually and demonstrating relationships between data points, making it easier for the human brain to process and remember.
8. **Attention-Grabbing:** Data visualisation can quickly grab the attention of the audience and help them remember the information presented.
9. **Increased Credibility:** Data visualisation demonstrates a business's understanding of its audience and ability to present meaningful data, increasing credibility.

● **Disadvantages of Data Visualization**

1. **Assessment Not Exactness:** Data visualisation provides an assessment of the data but not exactness, as the interpretation of the data can be subjective.
2. **One-Sided:** Data visualisation can be one-sided, presenting only a specific perspective and potentially missing important details.
3. **Complexity:** Highly complicated visualisations can appear cluttered or difficult to understand, requiring training on the tools used.
4. **Potential for Misinterpretation:** Users may draw incorrect conclusions from detailed visualisations, highlighting the importance of careful analysis.
5. **Data Privacy and Security:** Data visualisation must ensure the security and privacy of the data being presented, as a platform can be susceptible to cyberattacks or data sets may not comply with regulations.
6. **Bias:** Visualisations and the data they are based on must be scrutinised to ensure they are not intentionally or unintentionally biased.
7. **False Correlations:** Data visualisation can create false correlations among data points, leading to inaccurate assumptions and conclusions.
8. **Axes Make a Difference:** The size and number of axes in a visualisation can significantly impact how data is interpreted.
9. **Average Is Not the Best Statistic:** The average of a data set can be misleading if there are outliers, and it is crucial to consider both the mean and median.

Importance of Power BI and Python in Data Analysis:

Power BI and Python are powerful tools for data analysis and visualisation, offering a range of features and capabilities that enhance the data analysis process. Here are some key reasons why they are important:

- **Power BI**

1. **Interactive Reports and Dashboards:** Power BI provides a user-friendly platform for creating interactive reports and dashboards that facilitate data visualisation and business intelligence. It allows users to create custom visualisations, share insights with stakeholders, and collaborate on projects.
2. **Data Visualisation:** Power BI offers a variety of visualisation tools, including charts, graphs, maps, and infographics, to help users understand complex data and communicate insights effectively.
3. **Business Intelligence:** Power BI is designed for business intelligence, enabling users to analyse and visualise data from various sources, making it a valuable tool for data-driven decision-making.
4. **Integration with Python:** Power BI can integrate with Python, allowing users to leverage Python's data science and machine learning capabilities within the Power BI environment. This integration extends Power BI's capabilities for data transformation, visualisation, and machine learning.

- **Python**

1. **Data Science and Machine Learning:** Python is a popular programming language for data science and machine learning, offering a wide range of libraries and tools for data analysis, transformation, and visualisation.
2. **Data Transformation:** Python can perform complex data transformations, such as data cleaning, preprocessing, and feature engineering, which are essential steps in data analysis.
3. **Custom Visualisations:** Python can be used to create custom visualisations, such as heatmaps and correlation matrices, that are not easily achievable with Power BI alone.
4. **Automation:** Python scripts can automate tedious tasks, such as data loading and transformation, freeing up time for more complex and high-value tasks.

Power Bi Visualization VS Python Visualization

Power BI and Python are both powerful tools for data visualisation, but they have different strengths and use cases:

Power BI Visualization:

- Power BI provides a user-friendly platform for creating interactive reports and dashboards.
- It offers a wide variety of built-in visualisation types, including charts, graphs, maps, and infographics.
- Power BI is designed for business intelligence and is well-suited for analysing data from various sources and communicating insights to stakeholders.

- It can integrate with Python, allowing users to leverage Python's data science and machine learning capabilities within the Power BI environment.
- Power BI visualisations are best suited for interactive dashboards, real-time data monitoring, and tracking key performance indicators (KPIs).

Python Visualisation:

Python offers more flexibility and customization options for creating visualisations using libraries like Matplotlib and Seaborn.

It is preferred for in-depth statistical analysis, custom analytical methods, complex calculations, and advanced machine learning techniques that may not be feasible in Power BI or Tableau.

Python is helpful when data analysis requires specialised statistical analyses, predictive modelling, or advanced machine learning techniques.

Python visualisations are static images and cannot be easily shared with others who don't have Python installed.

Python is better suited for smaller to medium-scale projects and is more challenging to use for creating interactive dashboards compared to Power BI or Tableau.

When to Choose Power BI vs Python for Visualization

Choose Power BI when you need to create interactive dashboards, track real-time metrics, and share insights with non-technical users.

Choose Python when you require specialised calculations, custom analytical methods, complex statistical models, or advanced machine learning techniques.

Python is preferred for in-depth statistical analysis, hypothesis testing, and handling diverse data sources with complex transformations.

A combination of both tools is often used, with Python for data cleaning, preprocessing, and building advanced models, and Power BI for creating visualisations and sharing insights.

In summary, Power BI is a powerful tool for creating interactive dashboards and communicating insights to stakeholders, while Python offers more flexibility and customization options for advanced data analysis and visualisation tasks.

2. Getting Started

Setting Up Power BI:

- ***Download Power BI Desktop***
 1. Go to the official Power BI website and click on the "Download Free" button.

2. Select the appropriate version (32-bit or 64-bit) based on your system architecture.
3. Click "Next" to download the Power BI Desktop setup file to your computer.

- ***Install Power BI Desktop***

1. Open the downloaded setup file.
2. Click "Next" in the installation wizard to proceed.
3. Accept the licence agreement by checking the box and clicking "Next".
4. Choose the installation folder or use the default location, then click "Next".
5. Review the installation settings and click "Install" to begin the installation process.
6. Wait for the installation to complete, then click "Finish" to launch Power BI Desktop.

- ***Set Up Power BI Desktop***

1. When prompted, you can fill out the form with your name, company, email, and job title, or simply close the window.
2. If asked to sign in, you can skip this step for now.
3. You will now see the Power BI Desktop interface, which includes a ribbon with tabs like "Home," "View," "Modelling," and "Help".
4. On the left side, you'll find three layouts: "Report," "Data," and "Model".
5. In the "Report" layout, you can build your reports and dashboards.
6. The "Data" layout shows your uploaded or created data tables.
7. The "Model" layout allows you to create relationships between multiple tables based on common columns.
8. On the right side of the "Report" layout, you'll find sections for "Filters," "Visualisations," and "Fields".

Installing Python and Necessary Libraries

To install Python and essential libraries like Pandas, NumPy, Matplotlib, Seaborn, and Plotly for data analysis and visualisation, follow these detailed steps:

- **Installing Python**

1. Visit the official Python website and download the latest version of Python for your operating system.
2. Run the downloaded Python installer and follow the installation wizard instructions.
3. During installation, ensure to check the box that adds Python to your system's PATH to make it easier to run Python from the command line.

Installing Essential Libraries

Pandas, NumPy, Matplotlib, Seaborn, and Plotly

1. Open a command prompt or terminal after installing Python.

2. Use the following commands to install each library using Python's package manager, pip:

```
pip install pandas
```

```
pip install numpy
```

```
pip install matplotlib
```

```
pip install seaborn
```

```
pip install plotly
```

3. Verify the installation of each library by importing them in a Python script or Jupyter Notebook.

By following these steps, you will successfully install Python and the essential libraries Pandas, NumPy, Matplotlib, Seaborn, and Plotly for data analysis and visualisation.

3.Data Preparation

- **Data Acquisition**

Databases

1. Connect to databases like SQL Server, MySQL, PostgreSQL, etc. using database connectors or drivers
2. Write SQL queries to extract data from database tables and views
3. Use database APIs or libraries in programming languages like Python, R, Java, etc. to interact with databases

CSV and Excel Files

1. Read CSV files using libraries like pandas in Python or readr in R
2. Import data from Excel spreadsheets using language-specific libraries
3. Specify the file path or URL of the CSV/Excel file to read the data into a data frame or tibble

APIs

1. Make HTTP requests to API endpoints to retrieve data in formats like JSON or XML
2. Use API client libraries in programming languages to simplify API interactions

3. Handle authentication (API keys, OAuth, etc.) and pagination when making API requests

Web Scraping

1. Use web scraping libraries like BeautifulSoup in Python or rvest in R to parse HTML and extract data
2. Identify the relevant HTML elements containing the desired data using CSS selectors or XPath
3. Implement web scraping while respecting robots.txt and avoiding excessive requests to avoid overloading websites

Other techniques include:

1. Connecting to cloud storage services like Google Drive, Dropbox, etc. to access files
2. Reading data from real-time data streams or event logs
3. Importing data from statistical software formats like SPSS, SAS, Stata, etc.
4. The choice of data acquisition method depends on the data source, format, and accessibility. It's important to understand the data structure and ensure data quality during the acquisition process.

● Data Cleaning and Preprocessing:

To handle missing values, duplicates, outliers, and inconsistencies in data during the data cleaning and preprocessing process, several strategies can be employed:

Missing Values:

1. Identification: Use functions like `is.na()` in R or similar methods in Python to detect missing values.
2. Imputation: Fill missing values using techniques like mean, median, mode imputation, or advanced methods like k-Nearest Neighbors (KNN) imputation.
3. Removal: Eliminate rows or columns with excessive missing values using functions like `na.omit()`.

Duplicates:

1. Detection: Identify duplicate entries that overrepresent data by checking for identical records.
2. Handling: Decide whether to remove duplicates or merge conflicting information to ensure data integrity.

Outliers:

1. Detection: Use methods like the IQR method or Z-score method to identify extreme values that deviate significantly from the majority of data points.
2. Handling: Address outliers by removing them, transforming the data, or using robust statistical methods less sensitive to outliers.

Inconsistencies:

1. Identification: Look for inconsistencies in data formats, naming conventions, or variable types that can lead to errors in analysis.
2. Correction: Standardised formats, correct naming discrepancies, and ensure uniformity in data representation to enhance data consistency.

Best Practices:

1. Documentation: Maintain a record of changes made during the cleaning process for transparency and reproducibility.
2. Validation: Validate and test the effectiveness of data cleaning steps iteratively to ensure data quality.
3. Strategic Decisions: Make informed decisions on imputation, removal, or transformation based on the nature of the data and the analysis requirements.

By implementing these strategies, data scientists can ensure that the data is accurate, consistent, and free of errors, leading to more reliable and meaningful analysis results.

● **Data Transformation:**

Data transformation is the process of converting raw data into a structured, standardised format that is suitable for analysis and visualisation. This process involves several techniques to reshape, aggregate, and combine data to make it more

meaningful and useful for decision-making. Here are some key methods and techniques used in data transformation:

Reshaping Data

1. Smoothing: Removes noise from the dataset to highlight important features and patterns.
2. Data Generalization: Converts low-level attributes to high-level attributes to create a clear data snapshot.
3. Data Manipulation: Changes or alters data to make it more readable and organised.

Aggregating Data

1. Data Aggregation: Combines data from multiple sources into a single format for accurate analysis and reporting.
2. Data Discretization: Converts continuous data into discrete categories or bins *to improve efficiency and easier analysis.*

Combining Data

1. Data Integration: Combines data from multiple sources, such as databases and spreadsheets, into a single format.
2. Data Normalisation: Scales the data to a common range of values to facilitate comparison and analysis.

Other Techniques

1. Attribute Construction: Creates new attributes from existing attributes to ease data mining and analysis.
2. Log Transformation: Transforms data by taking the logarithm of the data to stabilise variance and improve normality.

Benefits of Data Transformation

1. Improved Data Quality: Removes errors, inconsistencies, and missing values.
2. Facilitates Data Integration: Enables integration of data from multiple sources.
3. Improves Data Analysis: Prepares data for analysis and modelling by normalising, reducing dimensionality, and discretizing data.
4. Increases Data Security: Masks sensitive data or removes sensitive information to increase security.
5. Enhances Data Mining Algorithm Performance: Reduces dimensionality and scales data to improve algorithm performance.

Challenges and Considerations

1. Time-Consuming: Data transformation can be a time-consuming process, especially for large datasets.
2. Complexity: Data transformation can be complex, requiring specialised skills and knowledge.
3. Data Loss: Data transformation can result in data loss, such as when discretizing continuous data.
4. Biassed Transformation: Data transformation can result in bias if not properly understood or used.
5. High Cost: Data transformation can be expensive, requiring significant investments in hardware, software, and personnel.

Tools and Libraries

1. Python Libraries: Pandas, NumPy, Matplotlib, Seaborn, Plotly
2. R Libraries: readr, dplyr, tidyr, ggplot2
3. SQL: SQL Server, MySQL, PostgreSQL

Best Practices

1. Documentation: Maintain a record of changes made during the transformation process for transparency and reproducibility.
2. Validation: Validate and test the effectiveness of transformation steps iteratively to ensure data quality.
3. Strategic Decisions: Make informed decisions on imputation, removal, or transformation based on the nature of the data and the analysis requirements.
- 4.

By understanding these methods and techniques, data scientists can effectively transform raw data into a structured and standardised format, making it suitable for analysis and visualisation.

Data Analysis Techniques

Regression Analysis

1. Analyses the relationship between one or more independent variables and a dependent variable

2. Used to explain how changes in the independent variables influence the dependent variable
3. Types include simple linear regression, multiple linear regression, logistic regression, etc.

Cluster Analysis

1. Groups data points into clusters based on similarity
2. Helps identify patterns and relationships in complex datasets
3. Commonly used for customer segmentation and anomaly detection

Time Series Analysis

1. Tracks data over time to identify trends, patterns, and relationships
2. Used for forecasting, anomaly detection, and understanding cyclical behavior
3. Techniques include moving averages, exponential smoothing, ARIMA models, etc.

Factor Analysis

1. Reduces a large number of variables to a smaller set of underlying factors
2. Uncovers hidden patterns and relationships in complex data
3. Helps with dimensionality reduction and feature extraction

Cohort Analysis

1. Breaks down data into related groups (cohorts) for deeper analysis
2. Allows comparing the behaviour of different cohorts over time
3. Commonly used in marketing and product analytics

Monte Carlo Simulation

1. Models the probability of different outcomes in a process that cannot easily be predicted
2. Incorporates multiple variables and scenarios to assess risk
3. Used for risk analysis, forecasting, and optimization problems

Sentiment Analysis

1. Analyses text data to determine the sentiment (positive, negative, neutral) expressed
2. Helps understand customer opinions, emotions, and attitudes
3. Widely used in social media monitoring and customer service

The choice of data analysis technique depends on the type of data, the research question, and the desired insights. Combining multiple techniques can provide a more comprehensive understanding of the data.

Introduction to Power BI

- **Interface Overview:**

1. Ribbon: The Ribbon at the top of Power BI Desktop contains tabs such as Home, View, and Modeling, each with various tools and commands for creating and managing your reports, data, and visualisations.

2. Report View: This is the main area where you create and design your reports. You can add and arrange visualisations, create dashboards, and customise the layout to present your data effectively.

3. Data View: This view allows you to inspect and manage the data that you've loaded into Power BI. You can see the tables, perform transformations, and make sure the data is structured correctly before using it in reports.

4. Visualisation Pane: Located on the right side, this pane contains a variety of visual elements you can use in your reports, such as charts, graphs, and maps. You can drag and drop these visualisations onto the Report View and customise their properties.

Power Bi Data Sources:

Microsoft Power BI is a business analytics service that provides interactive visualisations and business intelligence capabilities with an interface simple enough for end users to create their own reports and dashboards. With Power BI, you can connect to, transform, and visualise data from a wide range of sources, including

Excel, [BigQuery](#), SQL Server, and cloud-based services such as Azure SQL Database and [Salesforce](#). You can also use Power BI to create custom dashboards and reports that can be shared with your team or .

Connecting to Data Sources:

· Open Power BI Desktop:

- Launch Power BI Desktop on your computer.

· Get Data:

- Click on the "Home" tab in the Ribbon.
- Click on the "Get Data" button.

· Select CSV File:

- In the "Get Data" window, scroll down and select "Text/CSV" from the list of data sources.
- Click the "Connect" button.

· Choose CSV File:

- A file dialog will appear. Navigate to the location where your CSV file is stored.
- Select the CSV file and click "Open."

· Preview Data:

- Power BI will show a preview of the data in the CSV file.
- Check that the data looks correct. If everything is fine, click the "Load" button to import the data into Power BI.

· Transform Data (Optional):

- If you need to clean or transform the data before loading, click the "Transform Data" button instead of "Load."
- This will open the Power Query Editor, where you can perform various data transformations such as filtering rows, changing data types, splitting columns, etc.
- After making the necessary changes, click "Close & Apply" to load the transformed data into Power BI.

· Data Loaded:

- The data from the CSV file will now be loaded into Power BI and will appear in the Fields pane on the right side of the interface.
- You can now use this data to create visualizations and build your report.

Building Queries and Data Models :

1. Importing Data:

- **Get Data:** Start by importing your data into Power BI. Click on the "Home" tab, select "Get Data," choose your data source (e.g., CSV file), and load the data into Power BI.

2. Using Power Query Editor:

- **Open Power Query Editor:** After loading your data, click on "Transform Data" in the Home tab to open the Power Query Editor.
- **Transform Data:** In the Power Query Editor, you can clean and transform your data. Common transformations include removing unnecessary columns, filtering rows, splitting columns, merging tables, and changing data types.
- **Apply Changes:** Once you have made the necessary transformations, click "Close & Apply" to load the transformed data into Power BI.

3. Building Data Models:

- **Model View:** Switch to the Model view by clicking on the Model icon on the left side of the interface. This view allows you to see all your tables and their relationships.
- **Create Relationships:** Drag and drop fields between tables to create relationships. Relationships define how data from different tables can be combined. Make sure to set the correct relationship cardinality (one-to-one, one-to-many, many-to-many) and the correct cross-filter direction.
- **Manage Relationships:** Use the "Manage Relationships" button in the Modeling tab to view and edit existing relationships.

4. Enhancing the Data Model:

- **Create Calculated Columns:** Add new columns to your tables using DAX (Data Analysis Expressions). Go to the Modeling tab and select "New Column" to create calculated columns.
- **Create Measures:** Measures are calculations used in your reports. They are defined using DAX and can be created by clicking "New Measure" in the Modeling tab.
- **Hierarchies:** Organize your data by creating hierarchies. Right-click on a field in the Fields pane, select "New hierarchy," and add fields to it. This is useful for drill-downs in reports.

5. Building Visualizations:

- **Create Visuals:** Drag fields from the Fields pane onto the Report view to create visualizations. Choose from various visual types such as bar charts, line charts, pie charts, maps, and tables.

- **Customize Visuals:** Use the Visualization pane to customize your visuals. Adjust properties like colors, data labels, titles, and interactions.
- **Add Filters:** Apply filters to your report or specific visuals to refine the data displayed. You can use the Filters pane to add visual-level, page-level, or report-level filters.

6. Designing Reports:

- **Layout:** Arrange your visuals on the Report view to create a cohesive report. Use grids and alignment tools to ensure a clean layout.
- **Interactivity:** Set up interactions between visuals. Click on a visual, go to the Format tab, and choose how it interacts with other visuals on the page.
- **Bookmarks and Buttons:** Use bookmarks to capture report states and buttons to create navigation between report pages or to specific report states.

7. Publishing and Sharing:

- **Publish:** Once your report is ready, publish it to the Power BI service by clicking the "Publish" button in the Home tab.
- **Share:** In the Power BI service, you can share your report with others by creating dashboards, sharing links, or embedding reports in other applications.

Data Visualization with Power BI:

Data visualisation with Power BI involves creating interactive and visually appealing dashboards and reports from various data sources. Power BI is a powerful business analytics tool by Microsoft that allows users to transform raw data into meaningful insights through visually engaging charts, graphs, and maps.

Data Collection:

Data Collection for Hotel Management using CSV Files

Making data ready for Power BI for hotel management ranges from collecting data from different sources for each and every separate thing in the hotel like booking data, customer data, revenue data, inventory data, staff details, etc. The format of this data is usually a CSV as these files are simple and can be read by many systems.

- **dim_date.csv**

This is a data file that loads order logs and most importantly, a dataset for date-related dimensions, which allows us to have a well-established calendar to delve into the time-based analysis. The dimension contains columns like Date, mmm yy, Week no, day_type for deriving time series insight and trend analysis.

- **dim_rooms. csv**

Contains descriptive information about the hotel rooms recording important fields like Room_ID, Room_Class. This data carries dimensions, analysing room occupancy, availability, revenue per room type etc.

- **dim_hotels. csv**

It contains information on different hotels in the chain or network. The above table contains the columns as property_id, property_name, category ,city. This is important in order to compare across different properties.

- **fact_aggregated_bookings. csv file**

This file maintains an overview of all the booking data into a common file of elevated booking metrics such as daily, monthly, or yearly. The columns include property_id, check_in_date, room_categories, successful_bookings, capacity This abstract fact table is for awesome performance monitoring and trend analysis.

- **fact_bookings. csv**

This file contains the data of each type of individual booking. It consists a set of columns - booking_id, property_id, booking_date, check_in_date, check_out_date, no_guests, room_categories, booking_platform, ratings_given, revenue_generated, revenue_realized. This granular data helps in detailed analysis of booking trends, user slippage, and revenue generations.

Data Cleaning:

Data cleaning is a crucial step in preparing your data for analysis in Power BI. The Power Query Editor in Power BI provides a robust set of tools for cleaning and transforming your data.

Missing values:

1. Identification: When working with data in Power BI, missing values will display as null or blanks in your data preview. Use the Column Quality tool to see how many values are valid, error, and empty for each column.
2. Imputation: Imputation of defaults, means, medians, modes within Power BI itself when handling missing data For advanced algorithms, examples being k-Nearest Neighbors (KNN), please export the data to Python or R to impute and re-import cleaned data.
3. Removal: Eliminate rows with missing values using functions like remove empty or remove blank rows. For columns with excessive missing data, simply delete the entire column to maintain data integrity.

Duplicates:

1. Detection: Identify duplicate entries in Power BI by comparing records for exact matches, aiding in the identification of overrepresented data.
2. Handling: Decide between removing duplicates outright or merging conflicting information to uphold data integrity, ensuring accurate analysis in Power BI.

Outliers:

1. Detection: Utilise methods like IQR or Z-score in Power BI to pinpoint extreme values that deviate significantly from the dataset's majority, aiding in outlier identification.
2. Handling: Address outliers by either removing them outright, transforming the data, or applying robust statistical methods less sensitive to outlier influence.

Inconsistencies:

1. Identification: In Power BI, spot inconsistencies in data formats, naming conventions, or variable types that may compromise analysis accuracy.
2. Correction: Standardise formats, rectify naming discrepancies, and ensure uniform data representation to enhance overall consistency and reliability.

Best Practices:

1. Documentation: Maintain transparent records of cleaning processes in Power BI to ensure reproducibility and transparency.
2. Validation: Continuously validate and test data cleaning steps in Power BI to guarantee data quality and reliability.
3. Strategic Decisions: In Power BI, base decisions on imputation, removal, or transformation on data nature and analysis requirements for optimal results.

Data Transformation:

Reshaping Data:

1. Smoothing: Use Power BI functions to remove noise and highlight important features and patterns in the dataset.
2. Data Generalization: Convert low-level attributes to high-level attributes using Power Query Editor for a clearer data snapshot.
3. Data Manipulation: Power Query provides tools to change or alter data to improve readability and organisation.

Aggregating Data:

1. Data Aggregation: Combine data from various sources into a single format using Power BI's data modelling capabilities for accurate analysis and reporting.

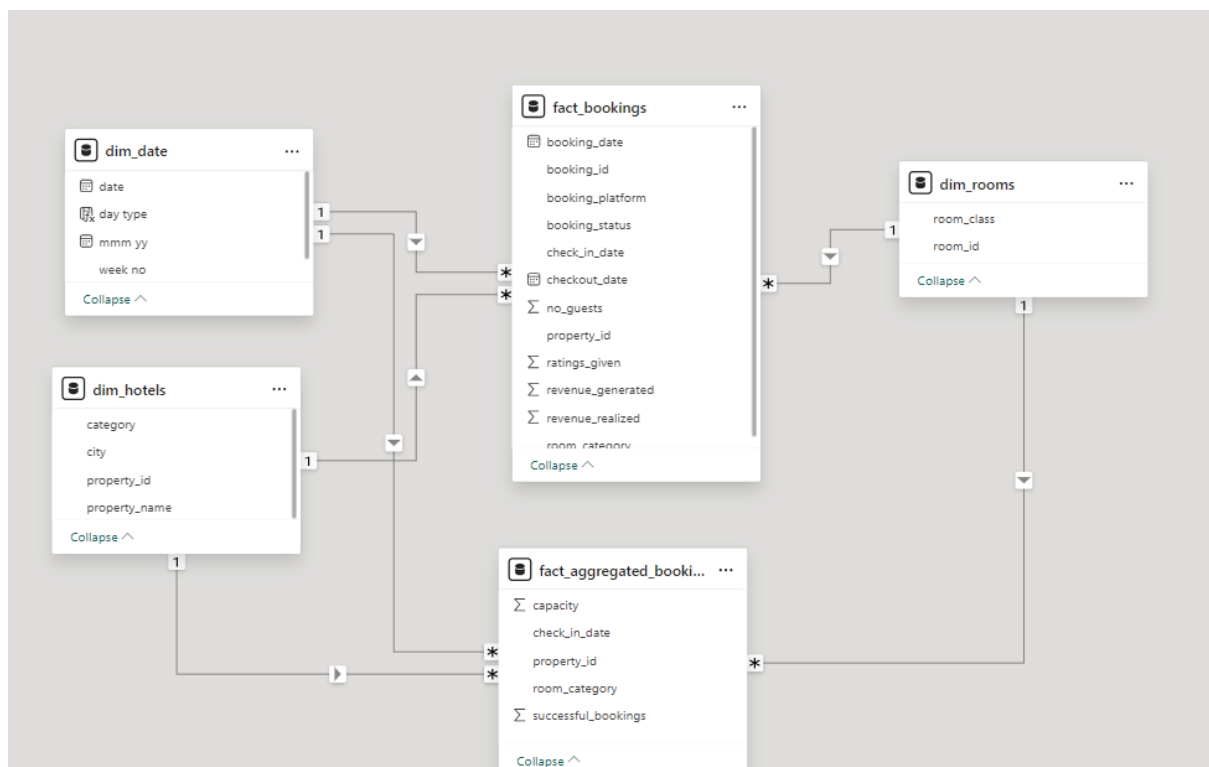
2. Data Discretization: Use Power BI features to convert continuous data into discrete categories or bins for easier analysis.

Combining Data:

1. Data Integration: Combine data from multiple sources like databases and spreadsheets into a single format using Power Query's data integration capabilities.
2. Data Normalization: Scale data to a common range of values using Power BI functions to facilitate comparison and analysis.
3. Attribute Construction: Create new attributes from existing ones using Power Query Editor to ease data mining and analysis.

Data Modelling:

Data modeling involves creating a logical structure for organizing and analyzing data. The provided datasets include dim_date.csv, dim_hotels.csv, dim_rooms.csv, fact_aggregated_bookings.csv, and fact_bookings.csv. These datasets can be used to create a logical data model (LDM) with relationships between them. The LDM Modeler in tools like Good Data can be used to create datasets, relationships, and datasets. This enables date-based analysis and supports aggregation at different levels. The datasets can be joined to combine data from different sources, and date functions can be used for data analysis.



In the center of the diagram is a fact table, called fact_bookings. Fact tables store measures of business processes. In this example, the fact table likely stores transactional data about hotel bookings, such as the number of guests, room rate, and revenue generated.

Six dimension tables surround the fact table. Dimension tables store the attributes of the facts in the fact table. For example, the dim_date table likely stores dates and their attributes, such as day type or week number.

Each dimension table is connected to the fact table by a foreign key relationship. A foreign key is a column in one table that references the primary key of another table. This creates a one-to-many relationship between the dimension tables and the fact table. This means that a single record in the fact table can be linked to multiple records in the dimension tables. For example, a single booking record (fact table) can be linked to multiple date records (dim_date table).

Relationships between tables :

The relationships between the tables are all one-to-many means; single record in a fact table (fact_bookings or fact_aggregated_bookings) can be linked to multiple records in the dimension tables.

- dim_hotels.csv: One-to-many with fact_bookings.csv (a hotel can have many bookings).
- dim_hotels.csv: One-to-many with fact_aggregated_bookings.csv (a hotel can have many booking summaries).
- dim_rooms.csv : One-to-many with fact_bookings.csv (a room category can have many bookings).
- Dim_date.csv : Many-to-one with fact_bookings.csv (a booking can be on a specific date, but the date table might not be directly linked).

DAX:

DAX (Data Analysis Expressions) is a formula expression language used in Power BI, Power Pivot in Excel, and SQL Server Analysis Services (SSAS). It is used to perform advanced calculations and queries on data in related tables and columns in tabular data models. DAX formulas include functions, operators, and values to create new fields and tables in the model, allowing for dynamic and powerful calculations.

Key Concepts and Functions

Syntax: Proper DAX syntax includes various elements, such as functions, constants, and strings.

Functions: DAX functions are predefined formulas that take parameters and perform specific calculations. Examples include aggregate functions like MIN, MAX, SUM, and AVERAGE, as well as text, date, and logical functions.

Context: DAX uses context to determine which rows should be used to perform a calculation. There are two types of context: row context and filter context. Row context applies when a formula filters a single row in a table, while filter context applies when additional filters are applied to narrow down the values

Using DAX in Power BI

- **Calculated Columns:** DAX formulas can be used to create calculated columns in a table, which are calculated immediately and stored in the in-memory data model.
- **Calculated Measures:** DAX formulas can also be used to create calculated measures, which are used to perform calculations and are typically displayed in reports.
- **Joining Tables:** DAX can be used to join two tables by leveraging existing relationships between the tables and using functions like ADDCOLUMNS to add additional columns

Sno	Measures	Description/purpose	DAX formula	Table
1	Revenue	To get the total revenue_realized	Revenue = SUM(fact_bookings[revenue_realized])	fact_bookings
2	Total Bookings	To get the total number of bookings happened	Total Bookings = COUNT(fact_bookings[booking_id])	fact_bookings
3	Total Capacity	To get the total capacity of rooms present in hotels	Total Capacity = SUM(fact_aggregated_bookings[capacity])	fact_aggregated_bookings
4	Total Successful Bookings	To get the total successful bookings happened for all hotels	Total Successful Bookings = SUM(fact_aggregated_bookings[successful_bookings])	fact_aggregated_bookings
5	Occupancy %	Occupancy means total successful bookings happened to the total rooms	Occupancy % = DIVIDE([Total Successful Bookings],[Total Capacity],0)	fact_aggregated_bookings

		available(capacity)		
6	Average Rating	Get the average ratings given by the customers	Average Rating = AVERAGE(fact_bookings[ratings_given])	fact_bookings
7	No of days	To get the total number of days present in the data. In our case, we have data from May to July. So 92 days.	No of days = DATEDIFF(MIN(dim_date[date]),MAX(dim_date[date]),DAY) +1	dim_date
8	Total canceled bookings	To get the "Canceled" bookings out of all Total bookings happened	Total canceled bookings = CALCULATE([Total Bookings],fact_bookings[booking_status]="Canceled")	fact_bookings
9	Cancellation %	calculating the cancellation percentage.	Cancellation % = DIVIDE([Total canceled bookings],[Total Bookings])	fact_bookings
10	Total Checked Out	To get the successful 'Checked out' bookings out of all Total bookings happened	Total Checked Out = CALCULATE([Total Bookings],fact_bookings[booking_status]="Checked Out")	fact_bookings
11	Total no show bookings	To get the "No Show" bookings out of all Total bookings happened ("No show" means those customers who neither canceled nor attend to their booked rooms)	Total no show bookings = CALCULATE([Total Bookings],fact_bookings[booking_status]="No Show")	fact_bookings
12	No Show rate %	calculating the no show percentage.	No Show rate % = DIVIDE([Total no show bookings],[Total Bookings])	fact_bookings
13	Booking % by Platform	To show the percentage contribution of each booking platform for bookings in hotels. We have booking platforms like makeyourtrip, logtrip, tripster etc)	Booking % by Platform = DIVIDE([Total Bookings], CALCULATE([Total Bookings], ALL(fact_bookings[booking_platform]))*100	fact_bookings

14	Booking % by Room class	<p>To show the percentage contribution of each room class over total rooms booked.</p> <p>We have room classes like Standard, Elite, Premium, Presidential</p>	<p>Booking % by Room class = $\text{DIVIDE}([\text{Total Bookings}], \text{CALCULATE}([\text{Total Bookings}], \text{ALL}(\text{dim_rooms}[\text{room_class}])) * 100$</p>	fact_bookings, dim_rooms
15	ADR	<p>Calculate the ADR(Average Daily rate)</p> <p>It is the ratio of revenue to the total rooms booked/sold. It is the measure of the average paid for rooms sold in a given time period</p>	<p>$\text{ADR} = \text{DIVIDE}([\text{Revenue}], [\text{Total Bookings}], 0)$</p>	fact_bookings
16	Realisation %	<p>calculate the realisation percentage.</p> <p>It is nothing but the successful "checked out" percentage over all bookings happened.</p>	<p>$\text{Realisation \%} = 1 - ([\text{Cancellation \%}] + [\text{No Show rate \%}])$</p>	fact_bookings
17	RevPAR	<p>Calculate the RevPAR(Revenue Per Available Room)</p> <p>RevPAR represents the revenue generated per available room, whether or not they are occupied. RevPAR helps hotels measure their revenue generating performance to accurately price rooms. RevPAR can help hotels measure themselves against other properties or brands.</p>	<p>$\text{RevPAR} = \text{DIVIDE}([\text{Revenue}], [\text{Total Capacity}])$</p>	fact_bookings, fact_agg_bookings
18	DBRN	<p>calculate DBRN(Daily Booked Room Nights)</p>	<p>$\text{DBRN} = \text{DIVIDE}([\text{Total Bookings}], [\text{No of days}])$</p>	fact_bookings, dim_date

		This metrics tells on average how many rooms are booked for a day considering a time period		
19	DSRN	<p>calculate DSRN(Daily Sellable Room Nights)</p> <p>This metrics tells on average how many rooms are ready to sell for a day considering a time period</p>	DSRN = DIVIDE([Total Capacity], [No of days])	fact_agg_bookings, dim_date
20	DURN	<p>calculate DURN(Daily Utilized Room Nights)</p> <p>This metric tells on average how many rooms are successfully utilized by customers for a day considering a time period</p>	DURN = DIVIDE([Total Checked Out],[No of days])	fact_bookings , dim_date
21	Revenue WoW change %	<p>To get the revenue change percentage week over week.</p> <p>Here, revcw for current week revpw for previous week</p>	<p>Revenue WoW change % =</p> <p>Var selv =</p> <p>IF(HASONEFILTER(dim_date[wn]),SELECTEDVALUE(dim_date[wn]),MAX(dim_date[wn]))</p> <p>var revcw =</p> <p>CALCULATE([Revenue],dim_date[wn]= selv)</p> <p>var revpw =</p> <p>CALCULATE([Revenue],FILTER(ALL(dim_date),dim_date[wn]= selv-1))</p> <p>return</p> <p>DIVIDE(revcw,revpw,0)-1</p>	dim_date







22	Occupancy WoW change %	To get the occupancy change percentage week over week. Here, revcw for current week revpw for previous week	Occupancy WoW change % = Var selv = IF(HASONEFILTER(dim_da te[wn]),SELECTEDVALUE(dim_date[wn]),MAX(dim_da te[wn])) var revcw = CALCULATE([Occupancy %],dim_date[wn]= selv) var revpw = CALCULATE([Occupancy %],FILTER(ALL(dim_date),d im_date[wn]= selv-1)) return DIVIDE(revcw,revpw,0)-1	dim_date
23	ADR WoW change %	To get the ADR(Average Daily rate) change percentage week over week. Here, revcw for current week revpw for previous week	ADR WoW change % = Var selv = IF(HASONEFILTER(dim_da te[wn]),SELECTEDVALUE(dim_date[wn]),MAX(dim_da te[wn])) var revcw = CALCULATE([ADR],dim_d ate[wn]= selv) var revpw = CALCULATE([ADR],FILTE R(ALL(dim_date),dim_date[wn]= selv-1)) return DIVIDE(revcw,revpw,0)-1	dim_date
24	Revpar WoW change %	To get the RevPar(Revenue Per Available Room) change percentage week over week. Here, revcw for current week revpw for previous week	Revpar WoW change % = Var selv = IF(HASONEFILTER(dim_da te[wn]),SELECTEDVALUE(dim_date[wn]),MAX(dim_da te[wn])) var revcw = CALCULATE([RevPAR],dim _date[wn]= selv) var revpw = CALCULATE([RevPAR],FIL TER(ALL(dim_date),dim_da te[wn]= selv-1))	dim_date

















			return DIVIDE(revcw,revpw,0)-1	
25	Realisation WoW change %	To get the Realisation change percentage week over week. Here, revcw for current week revpw for previous week	Realisation WoW change % = Var selv = IF(HASONEFILTER(dim_da te[wn]),SELECTEDVALUE(dim_date[wn]),MAX(dim_dat e[wn])) var revcw = CALCULATE([Realisation %],dim_date[wn]= selv) var revpw = CALCULATE([Realisation %],FILTER(ALL(dim_date),d im_date[wn]= selv-1)) return DIVIDE(revcw,revpw,0)-1	dim_date
26	DSRN WoW change %	To get the DSRN(Daily Sellable Room Nights) change percentage week over week. Here, revcw for current week revpw for previous week	DSRN WoW change % = Var selv = IF(HASONEFILTER(dim_da te[wn]),SELECTEDVALUE(dim_date[wn]),MAX(dim_dat e[wn])) var revcw = CALCULATE([DSRN],dim_ date[wn]= selv) var revpw = CALCULATE([DSRN],FILT ER(ALL(dim_date),dim_date [wn]= selv-1)) return DIVIDE(revcw,revpw,0)-1	dim_date

Data Analysis Expressions(DAX) in AtliQ Hospitality analysis:Aggregated columns:

Sn o.	Calculated Column name	Description/purpose	DAX formula	Table
1	wn	To get the week number from the corresponding date	wn=WEEKNUM(dim_date[date])	dim_date
2	day type	Based on the feedback from stakeholder, we considered Friday and Saturday as weekend and weekdays from Sunday to Thursday. In PowerBI, Sunday weekday number is 1, Monday is 2 and so on. So, if weekday number is greater than 5, the weekend or else weekday.	day type = Var wkd = WEEKDAY(dim_date[date],1) return IF(wkd>5,"Weekend", "Weekday")	dim_date

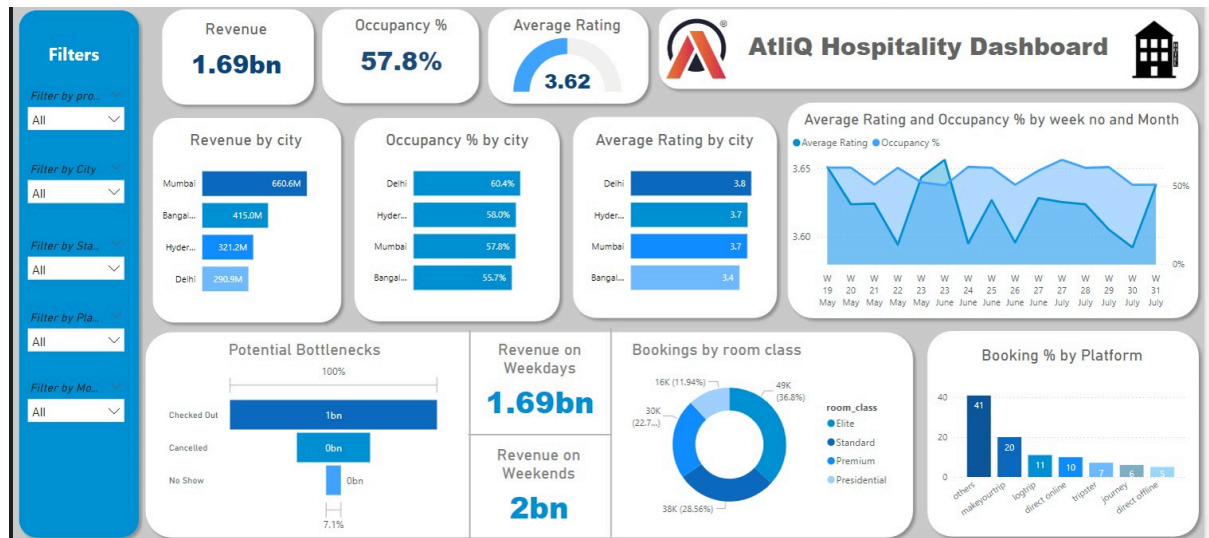
Key measures:

-  Total Bookings
-  Total cancelled bookings
-  Total Capacity
-  Total Checked Out
-  Total no show bookings
-  Total Successful Bookings

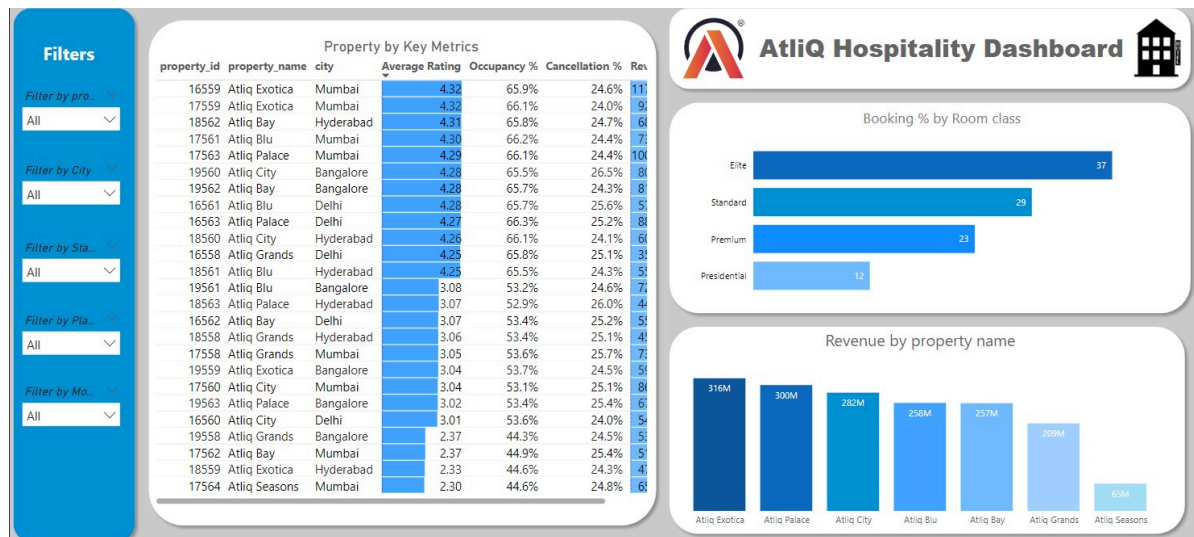
-  ADR
-  ADR WoW change %
-  Average Rating
-  Booking % by Platform
-  Booking % by Room class
-  Cancellation %
-  DBRN
-  DSRN
-  DSRN WoW change %
-  DURN
-  No of days
-  No Show rate %
-  Occupancy %
-  Occupancy WoW change %
-  Realisation %
-  Realisation WoW change %

Data Visualisation :

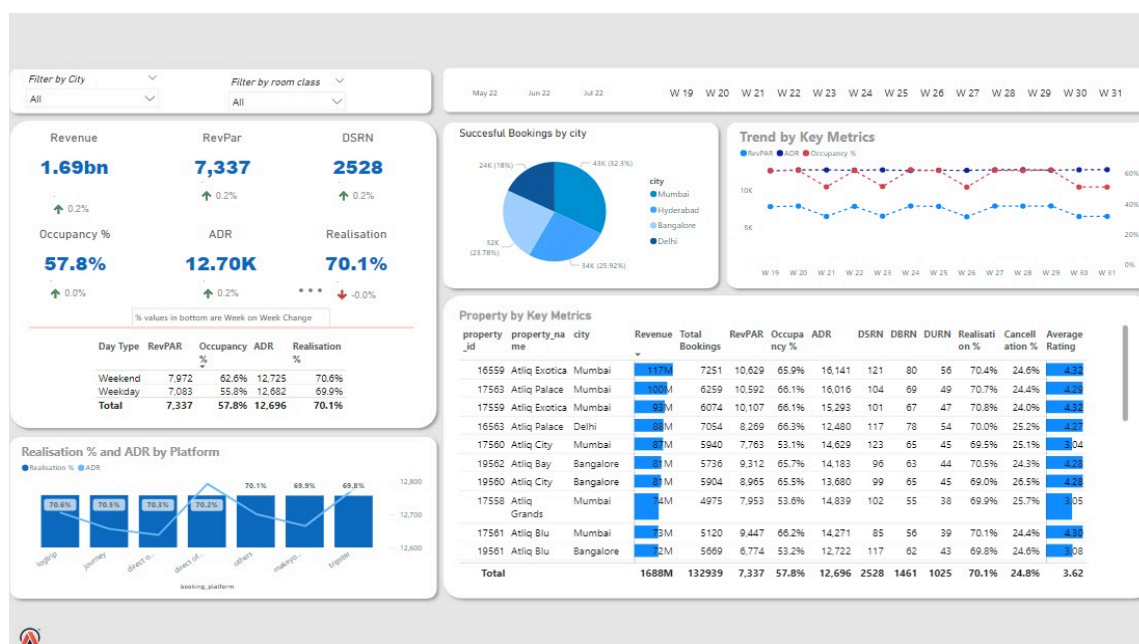
DASHBOARD:



The **AtliQ Hospitality Dashboard** provides a comprehensive overview of key performance indicators for a hospitality business. The main metrics displayed include **total revenue of 1.69 billion**, an **occupancy rate of 57.8%**, and an **average customer rating of 3.62**. Revenue is further broken down by city, with Mumbai leading at 660.6 million, followed by Bangalore, Hyderabad, and Delhi. **Occupancy percentages and average ratings** are also shown by city, highlighting **Delhi with the highest average rating of 3.8**. A line graph displays the average rating and occupancy percentage over several weeks and months. Additional details include potential bottlenecks, where **checked-out revenue reaches 1 billion**, and data on bookings by room class and platform. Bookings by room class indicate that Standard and Premium classes are the most popular, while the booking percentage by platform shows a significant share from 'others' and 'make you trip'. Filters on the left allow for data segmentation by property, city, state, platform, and month.

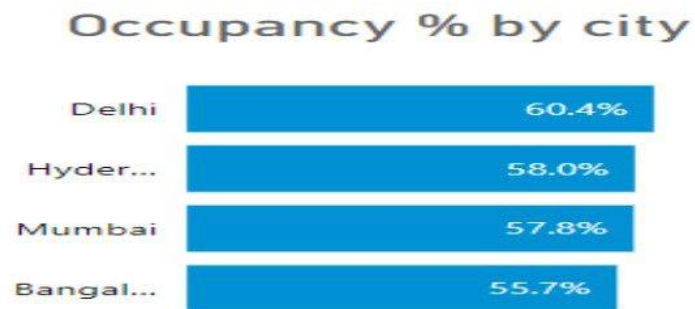


The second section of the AtliQ Hospitality Dashboard focuses on detailed property-specific metrics and booking data. The table lists properties by key metrics such as property ID, name, city, average rating, occupancy percentage, cancellation rate, and revenue. Notably, properties like AtliQ Exotica and AtliQ Bay in Mumbai and Hyderabad respectively, show high average ratings above 4.3 and occupancy rates above 65%. Booking percentages by room class are depicted in a bar chart, indicating the Elite class is the most booked at 37%, followed by Standard and Premium classes. A bar chart illustrates revenue by property name, with AtliQ Exotica generating the highest revenue at 316 million, followed by AtliQ Palace and AtliQ City with 300 million and 282 million respectively. Filters on the left side allow for data segmentation by property, city, state, platform, and month, enabling customised views of the data.



The third section of the AtliQ Hospitality Dashboard presents an in-depth analysis of various performance metrics and trends. Key figures include total revenue of 1.69 billion, RevPAR (Revenue per Available Room) at 7,337, DSRN (Days Since Reservation Number) at 2,528, an occupancy rate of 57.8%, ADR (Average Daily Rate) at 12,700, and a realization rate of 70.1%. The dashboard shows week-on-week changes for these metrics. A pie chart displays successful bookings by city, with Mumbai leading at 32.3%. A trend graph tracks RevPAR, ADR, and occupancy over several weeks. Property-specific data include total bookings, revenue, RevPAR, occupancy, ADR, DSRN, DBRN (Days Before Reservation Number), DURN (Days Until Reservation Number), realization percentage, cancellation rate, and average rating. AtliQ Exotica in Mumbai tops the list with 117 million in revenue and a high average rating of 4.32. Additionally, there is a comparison of realization percentage and ADR by booking platform, showing that platforms like 'logtrip' and 'makeyoutrip' have higher realization percentages. Filters for city and room class allow for tailored data views.

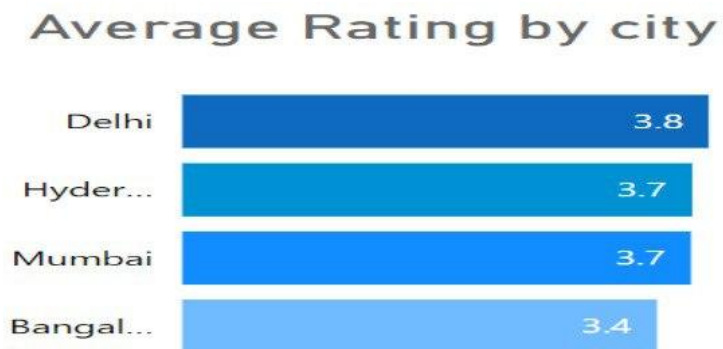
CLUSTERED BAR CHART



Occupancy Percentage By City

A clustered bar chart is a type of bar chart that displays multiple data series in a single chart. Each data series is represented by a cluster of bars, with each bar in the cluster corresponding to a specific category. The chart can be used to show changes over time, compare different groups, or illustrate the relationship between two variables. Clustered bar charts can be created using various tools such as SPSS Statistics, Excel, and Power BI, and are often used in business and scientific applications to analyze and present complex data effectively.

CLUSTERED BAR CHART:



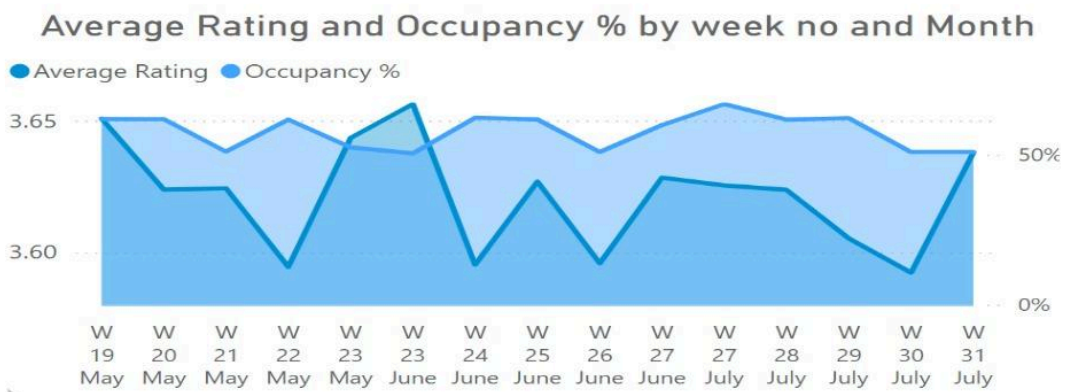
Average Rating By City

CLUSTERED BAR CHART:



Revenue By City

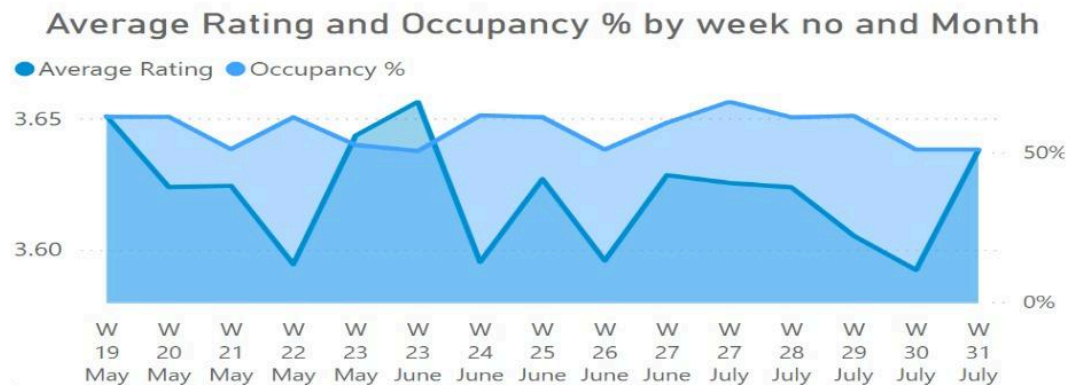
AREA CHART



Average Rating and Occupancy Percentage by Week Number And Month

An area chart, also known as an area graph, is a type of chart that displays quantitative data by filling the area between the line and the axis. It is a variation of the line chart and is commonly used to compare two or more quantities over time. The area between the line and the axis is often emphasized with colors, textures, and hatchings. Area charts are useful for showing trends, changes, and volume over time and are often used in business and scientific applications to analyze and present complex data effectively.

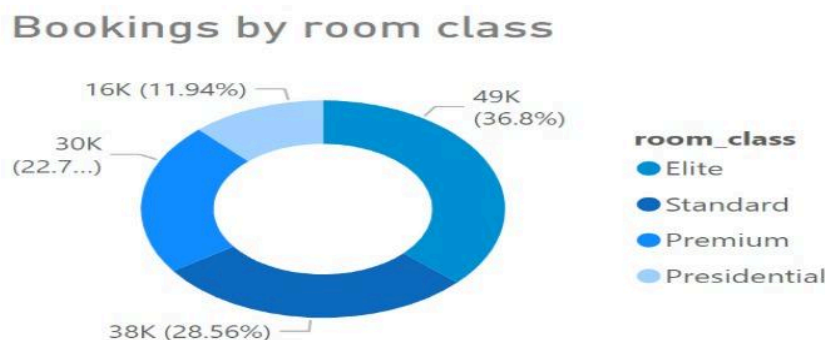
CLUSTERED COLUMN CHART



Average Rating and Occupancy Percentage by Week Number and Month

A clustered column chart is a type of bar chart that displays multiple data series in a single chart. Each data series is represented by a cluster of bars, with each bar in the cluster corresponding to a specific category. This visualization is useful for comparing multiple data series across different categories. The chart can be used to show changes over time, compare different groups, or illustrate the relationship between two variables. Clustered column charts are particularly effective for analyzing data with multiple dimensions and are commonly used in business and scientific applications to present complex data effectively.

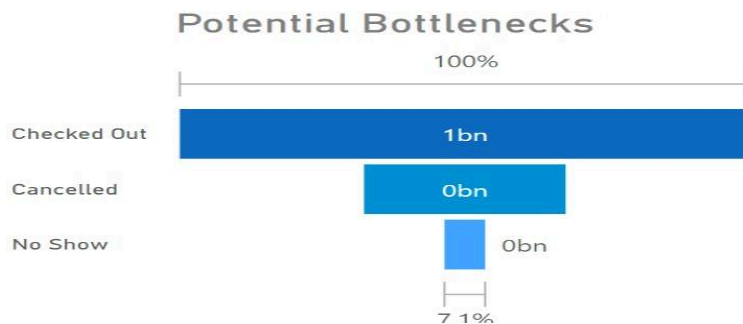
DONUT CHART



Bookings by Room Class

A donut chart, also known as a doughnut chart, is a variation of a pie chart that features a hole in the center. This empty space can be used to display additional data. To create a donut chart, a pie chart is modified by setting the innerRadius to a value or percentage. This setting can be configured to create a donut chart with a specific radius. Donut charts are useful for comparing categories and emphasizing the length of arcs, making them a popular choice for business and scientific applications.

FUNNEL CHART:



Potential Bottlenecks

A funnel chart is a data visualization tool used to represent the progressive reduction of data as it moves through different stages of a process. It is similar to a stacked percent column chart and is often used to illustrate stages in a sales process, such as sales leads to closed deals. Each stage is represented by a proportion of the total, with the size of the area determined by the series value as a percentage of the total. The chart helps identify potential problem areas and bottlenecks in the process, making it a valuable tool for executives and sales teams.

TABLE:

Property by Key Metrics						
property_id	property_name	city	Average Rating	Occupancy %	Cancellation %	Revenue
16559	Atliq Exotica	Mumbai	4.32	65.9%	24.6%	117040988
17559	Atliq Exotica	Mumbai	4.32	66.1%	24.0%	92890448
18562	Atliq Bay	Hyderabad	4.31	65.8%	24.7%	68445240
17561	Atliq Blu	Mumbai	4.30	66.2%	24.4%	73068992
17563	Atliq Palace	Mumbai	4.29	66.1%	24.4%	100245277
19560	Atliq City	Bangalore	4.28	65.5%	26.5%	80764995
19562	Atliq Bay	Bangalore	4.28	65.7%	24.3%	81353820
16561	Atliq Blu	Delhi	4.28	65.7%	25.6%	57206940
16563	Atliq Palace	Delhi	4.27	66.3%	25.2%	88037068
18560	Atliq City	Hyderabad	4.26	66.1%	24.1%	60192060
16558	Atliq Grands	Delhi	4.25	65.8%	25.1%	35610540
18561	Atliq Blu	Hyderabad	4.25	65.5%	24.3%	55300710
19561	Atliq Blu	Bangalore	3.08	53.2%	24.6%	72121440
18563	Atliq Palace	Hyderabad	3.07	52.9%	26.0%	44261060
16562	Atliq Bay	Delhi	3.07	53.4%	25.2%	55771464
18558	Atliq Grands	Hyderabad	3.06	53.4%	25.1%	45660890
17558	Atliq Grands	Mumbai	3.05	53.6%	25.7%	73822772
19559	Atliq Exotica	Bangalore	3.04	53.7%	24.5%	59229150
17560	Atliq City	Mumbai	3.04	53.1%	25.1%	86894072
19563	Atliq Palace	Bangalore	3.02	53.4%	25.4%	67748625
16560	Atliq City	Delhi	3.01	53.6%	24.0%	54296298
19558	Atliq Grands	Bangalore	2.37	44.3%	24.5%	53812500
17562	Atliq Bay	Mumbai	2.37	44.9%	25.4%	51384880
18559	Atliq Exotica	Hyderabad	2.33	44.6%	24.3%	47306410
17564	Atliq Seasons	Mumbai	2.30	44.6%	24.8%	65294229

Property by Key Metrics

In Power BI, a table visualization is a grid that displays related data in a logical series of rows and columns. It is useful for quantitative comparisons where you're looking at many values for a single category. Tables can be used to represent numerical data by category with multiple measures, display data as a matrix or in a tabular format, and review detailed data and exact values rather than visual representations.

CLUSTERED BAR CHART:



Booking Percentage by Room Class

A clustered bar chart is a type of bar chart that displays multiple data series in a single chart. Each data series is represented by a cluster of bars, with each bar in the cluster corresponding to a specific category. This visualization is useful for comparing multiple data series across different categories, such as sales revenue of various departments over several years. Clustered bar charts can be created using various tools like Excel, Google Sheets, and ChartExpo, and are commonly used in business and scientific applications to analyze and present complex data effectively

CLUSTERED COLUMN CHART:

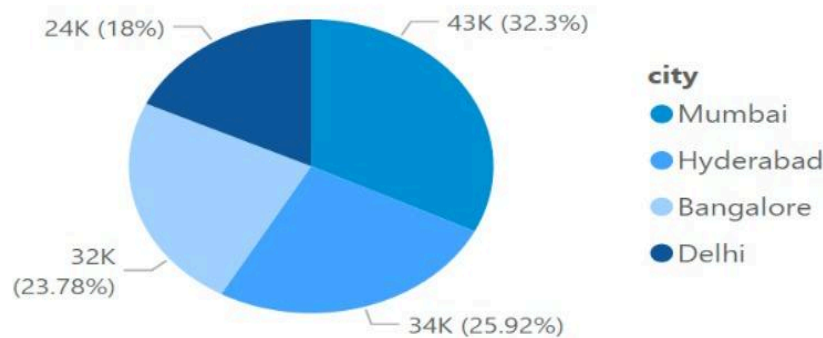


Revenue by Property Name

A clustered column chart is a type of bar chart that displays multiple data series in a single chart. Each data series is represented by a cluster of bars, with each bar in the cluster corresponding to a specific category. This visualization is useful for comparing multiple data series across different categories. The chart can be used to show changes over time, compare different groups, or illustrate the relationship between two variables. Clustered column charts are particularly effective for analyzing data with multiple dimensions and are commonly used in business and scientific applications to present complex data effectively

PIE CHART:

Successful Bookings by city

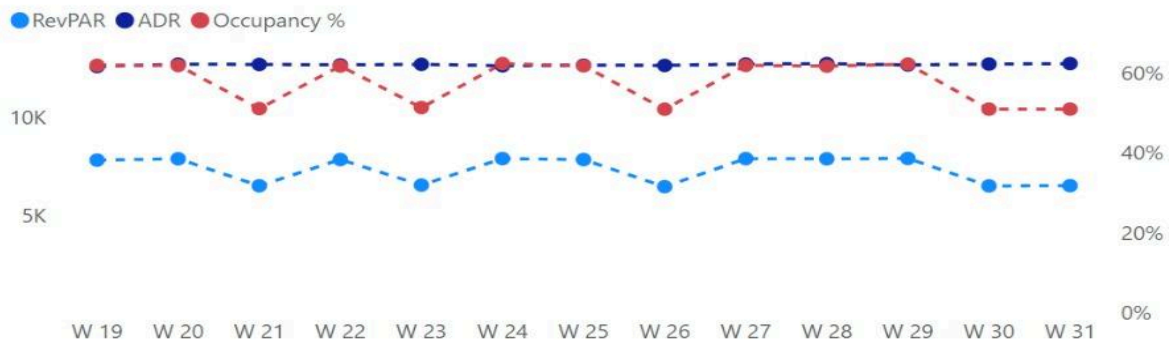


Successful Bookings by City

A pie chart is a circular statistical graphic that is divided into slices to illustrate numerical proportion. Each slice's arc length, central angle, and area are proportional to the quantity it represents. Pie charts are widely used in business and media, but have been criticized as it can be difficult to compare different sections or data across multiple pie charts. Pie charts can be replaced in most cases by other plots such as bar charts, box plots, or dot plots

LINE CHART:

Trend by Key Metrics

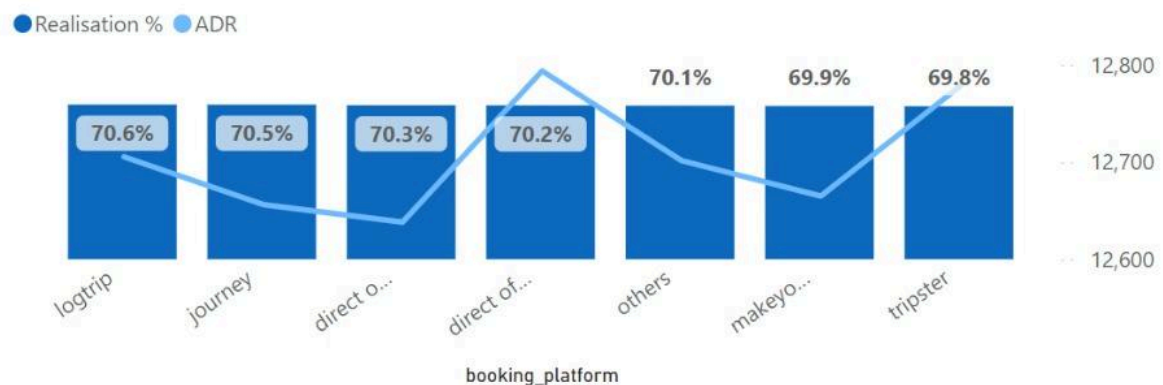


Trend by Key Metrics

A line chart is a graphical representation of data that connects data points with a continuous line. It is commonly used to show trends and patterns in numerical data, particularly in time series analysis. Line charts can be used to compare multiple data sets and identify trends. They can be customized with various options such as colors, labels, and gridlines to enhance their effectiveness in visualizing complex data.

LINE AND STACKED COLUMN CHART:

Realisation % and ADR by Platform



Realisation Percentage aADR by Platform

A line and stacked column chart combines a line chart and a stacked column chart into a single visualization. This allows for quick comparison between two sets of measures, such as

total sales and profit. The chart can have one or two y-axes, enabling the display of measures with different value ranges. Key features include the ability to customize the x-axis, y-axis, legend, gridlines, and data labels. This chart type is useful for analyzing trends and comparing performance across different categories or time periods.

Data Visualisation with Python

Data Collection:

Data Sources:

The data was collected from several CSV files, including `dim_date.csv`, `dim_hotels.csv`, `dim_rooms.csv`, `fact_aggregated_bookings.csv`, and `fact_bookings.csv`. These datasets provide comprehensive information about dates, hotels, room types, aggregated bookings, and detailed booking transactions.

Libraries Used:

- numpy: For numerical operations.
- pandas: For data manipulation and analysis.
- seaborn: For plotting and visualization.
- plotly.graph_objs: For creating complex and interactive plots.
- plotly.express: For creating high-level plots with minimal code.
- plotly.io: For input and output operations with Plotly.
- statsmodels.tsa.arima.model: For time series analysis using ARIMA models.
- statsmodels.graphics.tsaplots: For plotting autocorrelation and partial autocorrelation

Code for Data Collection:

The necessary libraries were imported, and the data was read from the CSV files using `pd.read_csv()` to load them into pandas DataFrames. This setup enables efficient data manipulation and analysis, providing a solid foundation for the subsequent steps of data exploration, cleaning, transformation, and visualization.

Data exploration:

Objective:

The aim of this phase is to understand the structure and content of the data, identify key features, and detect any anomalies or patterns that may inform subsequent steps of the analysis.

dim_date.csv:

The `dim_date.csv` dataset contains information on dates, including the month, year, week number, and whether the day is a weekday or weekend. The dataset has 92 entries and 4 columns with no missing values. The columns include `date`, `mmm yy`, `week no`, and `day_type`. After converting the `date` column to datetime format and extracting the week number, we updated the `day_type` column to accurately reflect weekends and weekdays based on the datetime conversion.

dim_hotels.csv:

The `dim_hotels.csv` dataset includes information about hotel properties, with 25 entries and 4 columns: `property_id`, `property_name`, `category`, and `city`. This dataset also has no missing values. We visualized the value counts for `property_name`, `category`, and `city` to understand the distribution of hotel properties across different categories and locations.

dim_rooms.csv:

The `dim_rooms.csv` dataset provides details on room types, including `room_id` and `room_class`. This dataset helps us understand the variety of room categories available in different hotels.

fact_aggregated_bookings.csv:

The `fact_aggregated_bookings.csv` dataset contains aggregated booking information, with 9200 entries and 5 columns: `property_id`, `check_in_date`, `room_category`, `successful_bookings`, and `capacity`. There are no missing values in this dataset. We examined the value counts for `property_id` and `room_category` to identify the most popular properties and room types.

fact_bookings.csv:

The `fact_bookings.csv` dataset provides detailed booking records, with 53535 entries and 12 columns, including `booking_id`, `property_id`, `booking_date`, `check_in_date`, `checkout_date`, `no_guests`, `room_category`, `booking_platform`, `ratings_given`, `booking_status`, `revenue_generated`, and `revenue_realized`. This dataset contains some missing values in the `ratings_given`, `booking_status`, `revenue_generated`, and `revenue_realized` columns. We visualized the value counts for `ratings_given`, `booking_status`, and `booking_platform` to understand customer feedback, booking statuses, and the popularity of different booking platforms.

Summary of Findings:

1. **dim_date.csv:** All entries are complete with no missing values. The day_type column was updated to accurately reflect weekends and weekdays.
2. **dim_hotels.csv:** No missing values. The distribution of properties across different categories and cities was visualised.
3. **dim_rooms.csv:** Information on different room types was collected.
4. **fact_aggregated_bookings.csv:** No missing values. Popular properties and room categories were identified.
5. **fact_bookings.csv:** Some missing values were identified in ratings and revenue columns. The distribution of ratings, booking statuses, and booking platforms was visualized.

These findings provide a foundational understanding of the data, allowing for more informed data cleaning, transformation, and visualization steps in the subsequent phases of the project.

Data Cleaning

fact_bookings DataFrame

The `fact_bookings` DataFrame contains information on bookings, including booking ID, property ID, booking date, checkin and checkout dates, the number of guests, and room category, among other variables. The data cleaning process for this DataFrame involved several key steps:

1. Filtering Out Invalid Guest Numbers:

- The dataset was filtered to identify and remove rows where the number of guests was zero or negative, as such values are logically invalid for booking records. This step ensured the integrity of the data, leaving us with a more accurate representation of guest bookings.

2. Revenue Analysis and Outlier Removal:

- The mean and standard deviation of the revenue generated were calculated. These values were used to determine the bounds for valid revenue entries. The higher limit for revenue was defined as the mean plus three times the standard deviation, while the lower limit was the mean minus three times the standard deviation.
- Rows where the revenue generated was below the lower limit or above the higher limit were identified as outliers. Specifically, bookings with revenue less than the lower limit were scrutinized, revealing several entries significantly deviating from typical revenue figures.
- Rows with revenue exceeding the higher limit were then removed from the dataset to minimize the impact of outliers on subsequent analyses. This step further refined the dataset by excluding anomalous entries, resulting in a cleaner dataset with reduced noise.

3. Handling Missing Values:

- The 'ratings_given' column had missing values, which were imputed using the mean of the nonmissing ratings. This approach ensured that the imputed values were representative of the existing data, maintaining the overall distribution and preventing the distortion of analytical outcomes.
- After imputation, a check for missing values across all columns confirmed that there were no remaining null values in the dataset. This validation step ensured completeness and reliability of the data for further analysis.

fact_aggregated_bookings DataFrame

The `fact_aggregated_bookings` DataFrame contains aggregated information on properties, such as the number of successful bookings and the capacity of each property. The cleaning process for this DataFrame involved:

- 1. Descriptive Statistics:** The `fact_aggregated_bookings` DataFrame was described to provide a statistical summary of the data. This summary included the count, mean, standard deviation, minimum, 25th percentile, median, 75th percentile, and maximum values for each column, offering insights into the distribution and central tendencies of the data.
- 2. Checking for Null Values:** An assessment of the DataFrame for null values indicated that there were no missing entries across all columns. This ensured the dataset's completeness and prepared it for further analysis without the need for additional imputation.
- 3. Logical Consistency Check:** The dataset was filtered to identify any instances where the number of successful bookings exceeded the capacity of a property. Such cases would indicate logical inconsistencies in the data, as it is not feasible for bookings to surpass the available capacity. This step was crucial for maintaining the data's logical integrity and ensuring that subsequent analyses were based on realistic and accurate information.

By completing these data cleaning steps, both `fact_bookings` and `fact_aggregated_bookings` DataFrames were prepared for robust and reliable data analysis. The removal of invalid and anomalous entries, along with the imputation of missing values and verification of logical consistency, ensured that the datasets were clean, accurate, and ready for further exploratory and inferential analyses.

Data transformation:

The data transformation process involved enhancing and integrating the datasets to derive additional insights and metrics necessary for detailed analysis. Key steps included computing new metrics, merging data from multiple DataFrames, and adding calculated fields to the main dataset.

Occupancy Calculation

The `fact_aggregated_bookings` DataFrame was transformed to include a new column, `occupancy%`, which represents the occupancy rate as a percentage. This was calculated by dividing the total number of successful bookings by the capacity for each property and then multiplying by 100.

Merging DataFrames

Multiple DataFrames (`dim_dates`, `dim_hotels`, `dim_rooms`, and `fact_aggregated_bookings`) were merged to create a comprehensive dataset. The merging process involved:

- **Merging `fact_aggregated_bookings` and `dim_rooms`:** The merge was performed on the `room_category` column from `fact_aggregated_bookings` and `room_id` from `dim_rooms`.
- **Merging the Result with `dim_hotels`:** The resulting DataFrame from the previous step was further merged with `dim_hotels` on the `property_id` column.
- **Converting Date Columns:** The `check_in_date` column was converted to datetime format to facilitate time-based calculations and merging.
- **Merging with `dim_date`:** The DataFrame was merged with `dim_date` on the `check_in_date` column, with `dim_date` being merged on the `date` column. This added temporal information to the dataset.
- **Dropping Redundant Columns:** The `room_id` column was dropped from the merged DataFrame to remove redundancy.

Enhanced Merged DataFrame (`df_2`)

A similar process was applied to the `fact_bookings` DataFrame, resulting in `df_2`, which includes comprehensive booking information merged with details from `dim_rooms`, `dim_hotels`, and `dim_date`. The following transformations were performed on `df_2`:

1. **Datetime Conversion:** The `booking_date` and `checkout_date` columns were converted to datetime format.
2. **Additional Temporal Columns:** New columns were added to capture the month and year of booking (`booking_month` and `booking_year`), the day of the week (`booking_day_of_week`), and the lead time (days between booking date and check-in date).
3. **Booking Duration:** A new column, `booking_duration`, was created to represent the number of days between check-in and checkout.
4. **Revenue Per Guest:** The `revenue_per_guest` column was calculated by dividing the `revenue_generated` by the number of guests.

Hotel Metrics Calculation

Various metrics were calculated to provide insights into the performance of the properties:

1. **Total Revenue:** Sum of revenue_realized across all bookings.
2. **Total Bookings:** Total number of bookings in the dataset.
3. **Total Successful Bookings:** Count of bookings with status 'Successful'.
4. **Average Rating:** Mean of the ratings_given column.
5. **Number of Days:** Total number of unique days covered in the booking dates.
6. **Total Cancelled Bookings:** Count of bookings with status 'Cancelled'.
7. **Cancellation Percentage:** Proportion of cancelled bookings out of total bookings.
8. **Total Checked Out:** Count of bookings with status 'Checked Out'.
9. **Total No Show Bookings:** Count of bookings with status 'No Show'.
10. **No Show Rate Percentage:** Proportion of no show bookings out of total bookings.
11. **Average Daily Rate (ADR):** Total revenue divided by total successful bookings.
12. **Realisation Percentage:** Proportion of checked out bookings out of total bookings.
13. **Daily Booked Room Nights (DBRN):** Total successful bookings divided by the number of days.
14. **Daily Utilized Room Nights (DURN):** Total checked out bookings divided by the number of days.
15. **Week-on-Week Revenue Change:** Revenue changes between current and previous weeks were calculated to assess trends over time.

The computed metrics were added as new columns to the final integrated DataFrame, which included all relevant data from the merged fact_bookings, dim_rooms, dim_hotels, and dim_date DataFrames. This enriched dataset facilitated comprehensive analysis and visualization of booking and property performance.

The transformation steps ensured that the data was well-structured, comprehensive, and ready for detailed exploratory and inferential analysis, providing a solid foundation for generating actionable insights.

Data Cleaning:

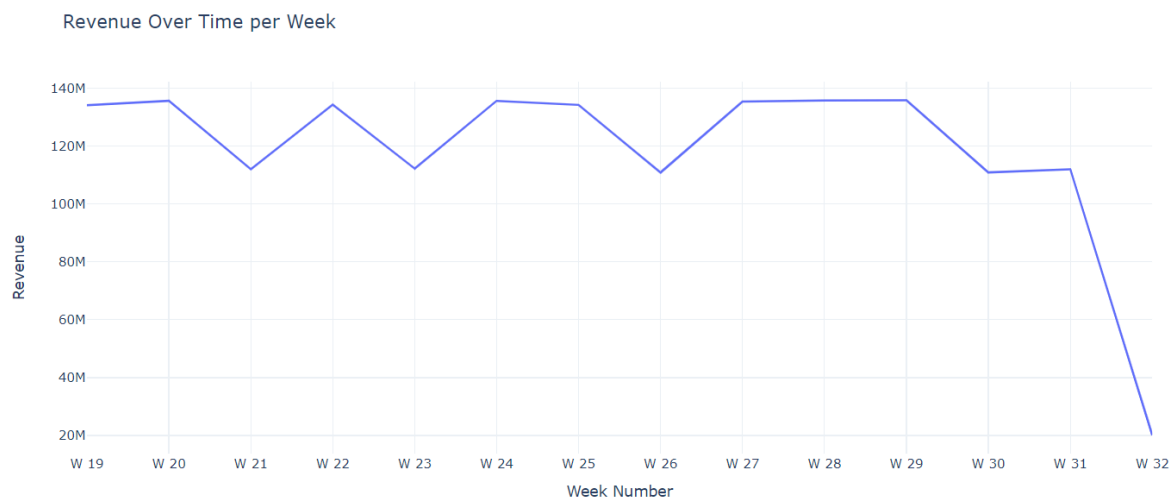
Before concluding, the code drops the redundant 'room_id' column from the `df` DataFrame to streamline the dataset. This step ensures that only relevant columns are retained for subsequent analyses, reducing redundancy and improving data clarity.

Data Visualization:

Finally, the code provides information about the DataFrame `df`, including column names, data types, and non-null counts. This summary helps in verifying the integrity of the transformed dataset and ensures that it is ready for exploratory analysis and further processing.

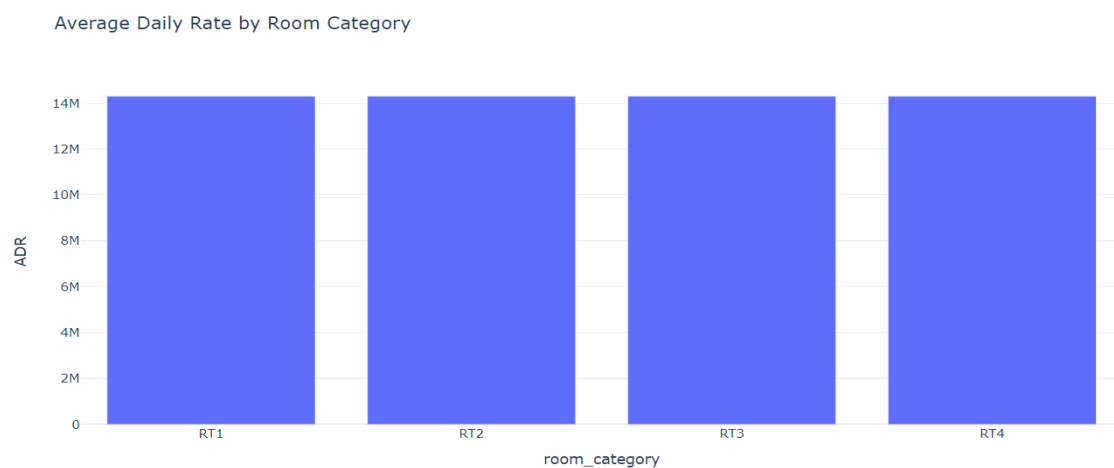
By performing these data transformation steps, the code prepares the AtliQ Hospitality dataset for in-depth analysis, enabling stakeholders to derive actionable insights and make data-driven decisions to enhance operational efficiency and customer satisfaction within the hospitality industry.

Line Plot



A Line plot is used to visualize revenue trends over time on a weekly basis. Line plots are useful for showing changes in a continuous variable over intervals of time. The plot displays the total revenue realized each week, allowing for easy identification of trends and patterns in the data. The x-axis represents the week number, while the y-axis shows the corresponding revenue. This visualization helps in understanding how revenue fluctuates over different weeks.

Bar chart

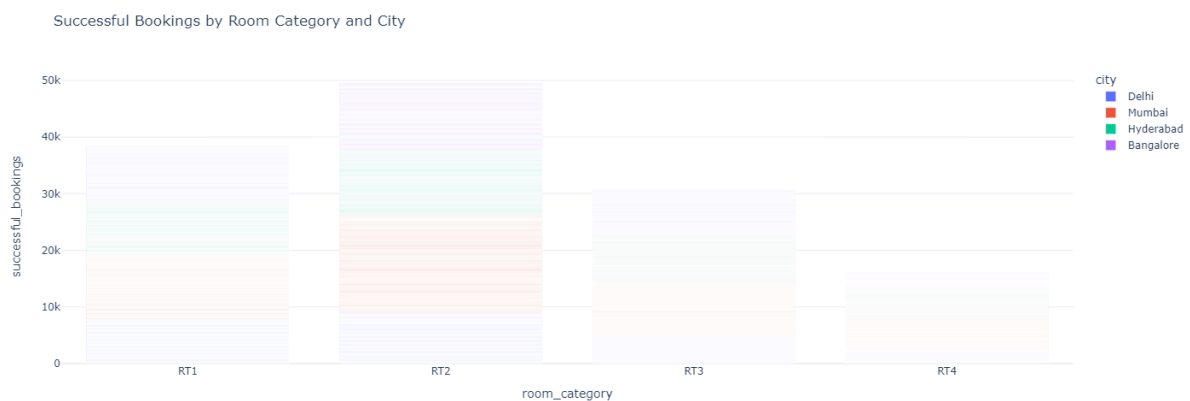


A bar chart is a visualization that displays categorical data with rectangular bars. The length of each bar represents the value associated with a category, making it easy to compare different categories side by side. In this visualization, the x-axis represents the different room categories, while the y-axis shows the Average Daily Rate (ADR) for each category. The ADR is calculated by dividing the total revenue by the number of days and averaging these values for each room category. The bar chart helps to compare the ADR across various room categories, highlighting which categories generate higher or lower daily rates on average.



A bar chart is used to represent categorical data with rectangular bars, where the length of each bar indicates the count or value associated with each category. In this visualization, the x-axis represents different room classes, and the y-axis shows the number of bookings for each class. The bars indicate the count of bookings for each room class, providing a clear comparison of the popularity of different room classes. The chart helps identify which room classes are booked more frequently.

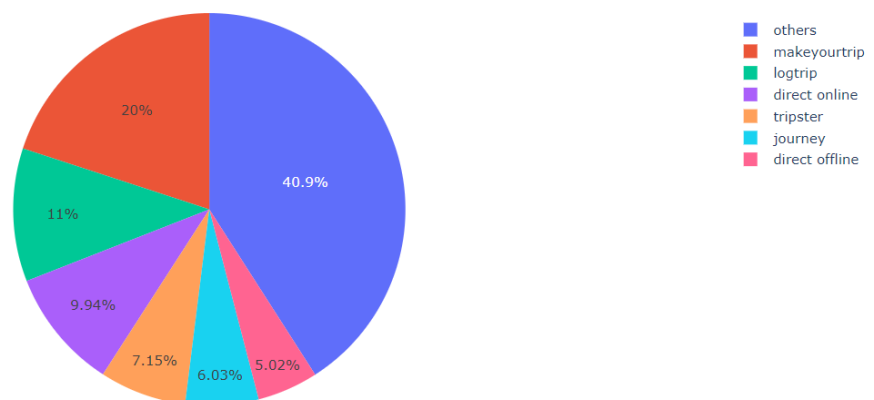
Bar chart



A bar chart is a graphical representation of categorical data, where each category is represented by a bar, and the height or length of the bar corresponds to the value of the category. In the visualization, the x-axis represents the different room categories, while the y-axis shows the number of successful bookings. The bars are color-coded by city, providing a breakdown of successful bookings for each room category across different cities. This helps in comparing the popularity or demand for various room categories within and across cities, highlighting trends and differences in booking patterns.

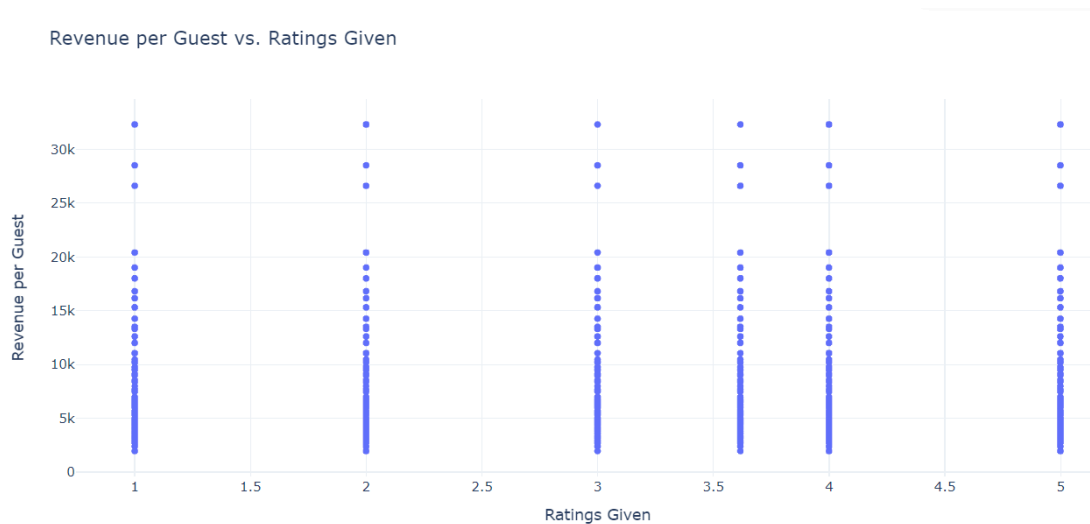
Pie chart

Revenue Distribution by Booking Platform

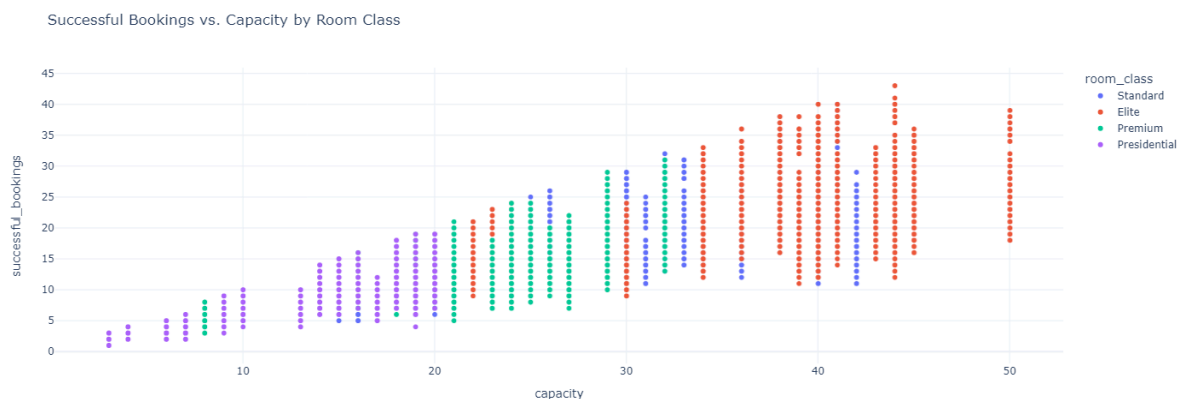


A pie chart is a circular statistical graphic divided into slices to illustrate numerical proportions. Each slice represents a category's contribution to the whole. In this visualization, each slice of the pie chart represents the total revenue generated by a specific booking platform. The size of each slice is proportional to the revenue amount, allowing for a visual comparison of the revenue distribution across different booking platforms. This chart helps to quickly understand which platforms are the most or least significant revenue sources.

Scatter plot

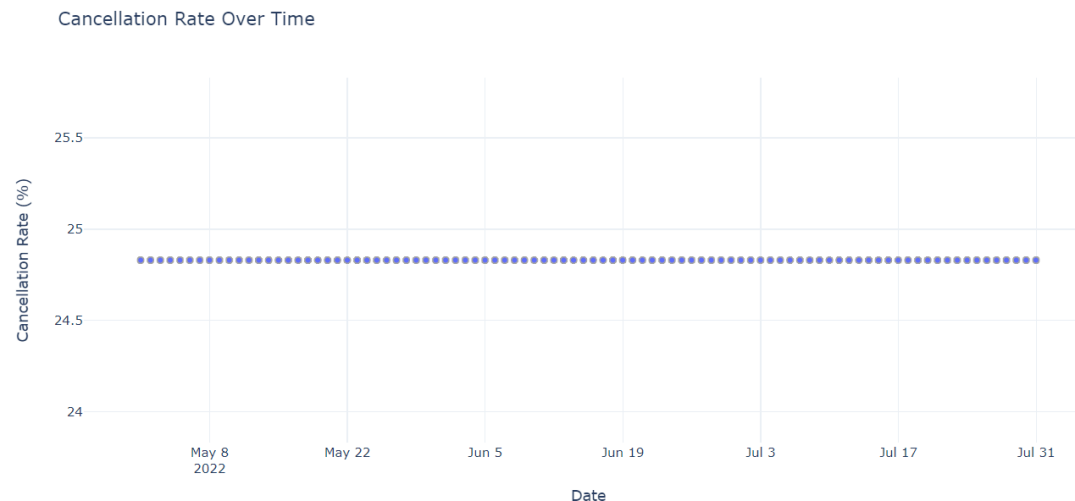


A scatter plot visualizes the relationship between two continuous variables by displaying individual data points on a two-dimensional plane. In this visualization, the x-axis represents the ratings given by guests, and the y-axis shows the revenue per guest. Each point corresponds to a booking, indicating the revenue generated by a guest and the rating they provided. The scatter plot helps to identify any correlations between higher ratings and revenue per guest, revealing potential trends or patterns in guest behavior and satisfaction.



A scatter plot is a graphical representation that uses dots to display the relationship between two continuous variables. Each dot represents an individual data point, showing how one variable is affected by the other. In the visualization, the x-axis represents the capacity of rooms, and the y-axis represents the number of successful bookings. The dots are color-coded by room class, indicating how different room classes perform in terms of bookings relative to their capacities. This scatter plot helps in identifying patterns or correlations between room capacity and successful bookings, and it can reveal trends specific to each room class, such as whether larger capacities are associated with more bookings for certain classes.

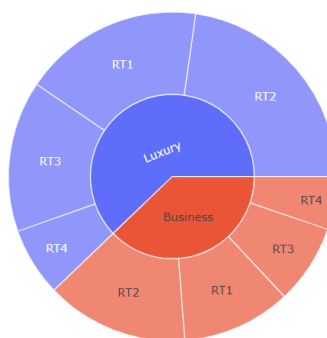
Scatter plot



A scatter plot is a visualization that displays individual data points on a two-dimensional plane, showing the relationship between two variables. Each point represents an observation in the dataset. In this visualization, the x-axis represents the date, and the y-axis shows the cancellation rate percentage. Each point on the scatter plot corresponds to the cancellation rate on a specific date. This plot helps to identify trends and patterns in cancellations over time, making it easier to observe periods with higher or lower cancellation rates.

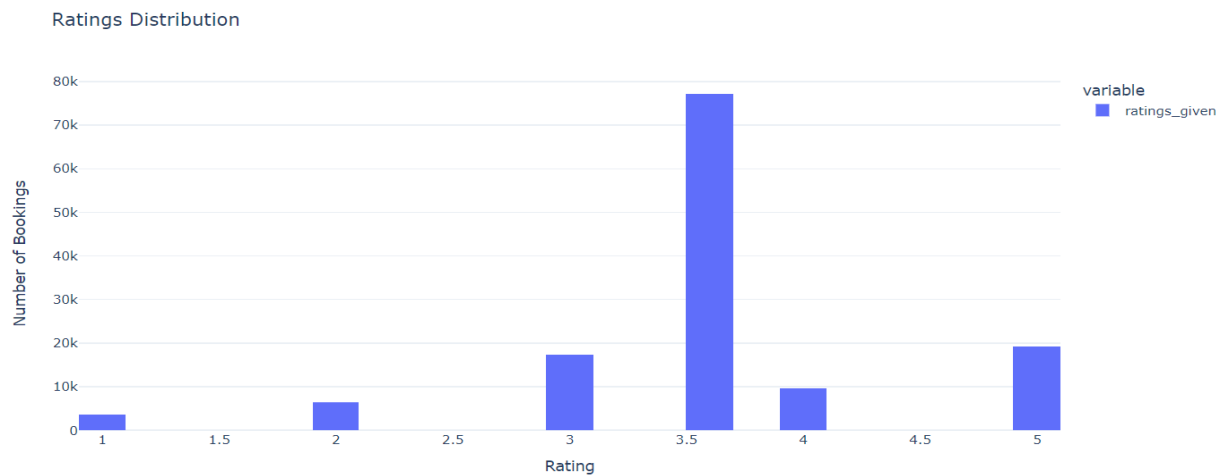
Sunburst chart

Booking Success Rates by Room Category

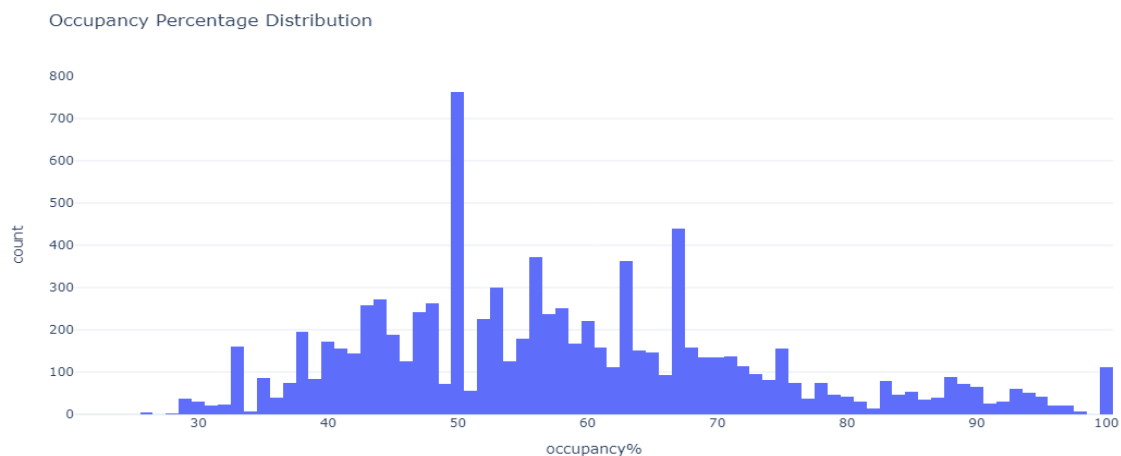


This sunburst chart visualizes booking success rates categorized by room category. Each segment in the chart represents a specific room category nested within its broader category. The size of each segment corresponds to the total number of successful bookings for that room category. This hierarchical representation allows for a clear understanding of which room categories have higher or lower booking success rates, providing insights into booking performance across different types of accommodations. The title of the sunburst chart is "Booking Success Rates by Room Category", with a color scale disabled for clarity.

Histogram

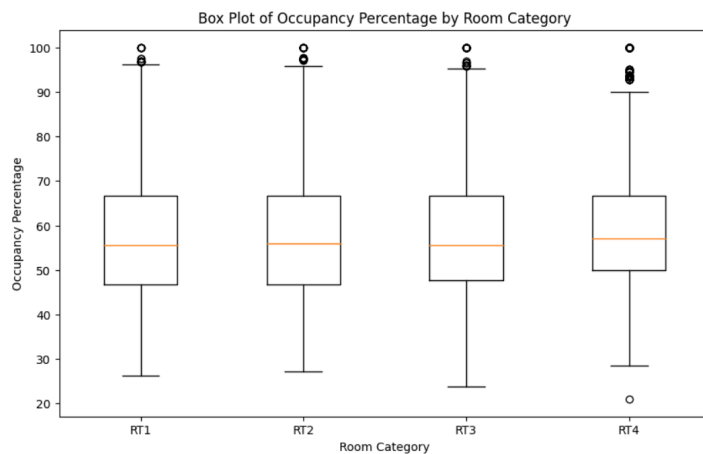


A histogram is a visualization that displays the distribution of a continuous variable by dividing it into bins and showing the frequency of observations within each bin. In this visualization, the x-axis represents the ratings given by guests, divided into bins. The y-axis shows the count of bookings that fall within each rating bin. The histogram helps to understand the overall distribution of guest ratings, identifying areas with high and low frequencies, and providing insight into the typical satisfaction levels among guests.

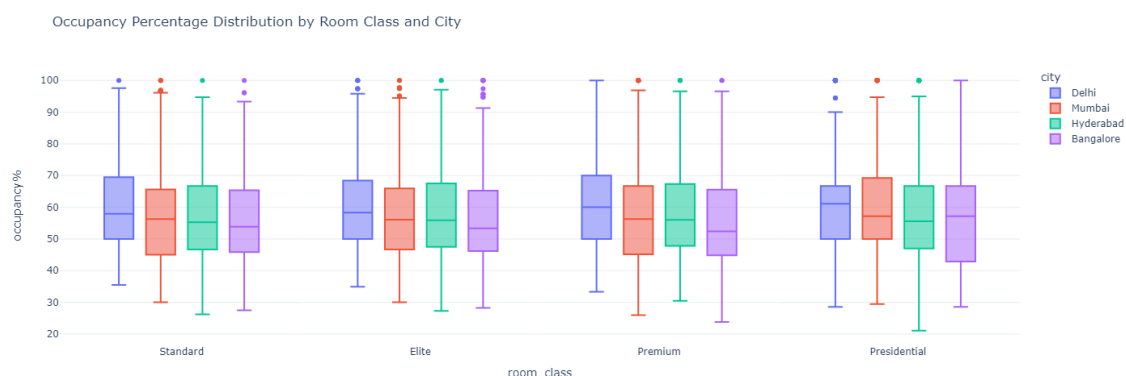


Histogram is a visualization, which displays the distribution of occupancy percentages across the dataset. Histograms are useful for understanding the frequency distribution of a continuous variable. In the visualization, the x-axis represents the occupancy percentages, divided into bins. The y-axis shows the count of occurrences within each bin. The histogram helps to identify the overall shape of the distribution, such as whether it is skewed, uniform, or follows a normal distribution. Additionally, it highlights areas with high and low frequencies, providing insight into the typical occupancy percentages in the dataset.

Box plot

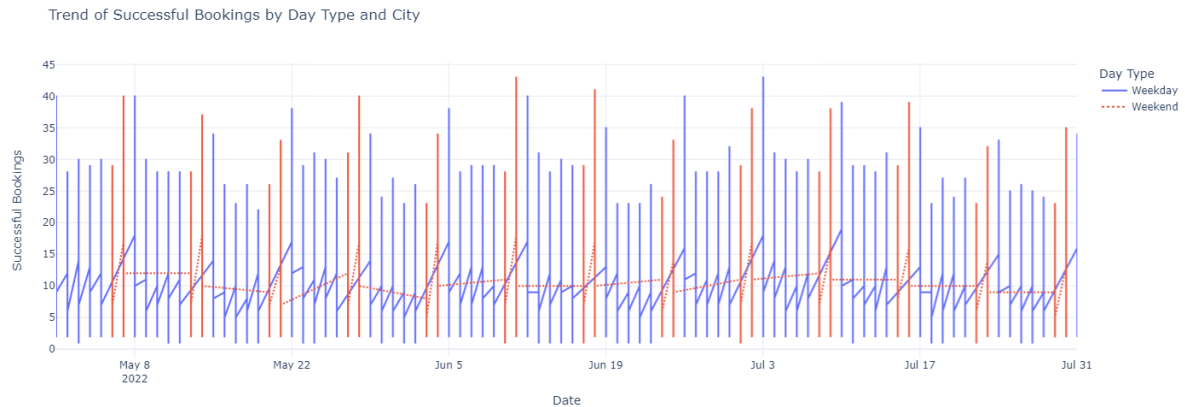


This visualization is a box plot, which displays the distribution of occupancy percentages across different room categories. Box plots are useful for identifying the central tendency, dispersion, and potential outliers within each category. In the visualization, each box represents the interquartile range (IQR) of occupancy percentages for a specific room category, with the median shown as a line within the box. The whiskers extend to the smallest and largest values within 1.5 times the IQR from the lower and upper quartiles, respectively. Outliers beyond this range are displayed as individual points, highlighting any extreme values in occupancy percentages for each room category.



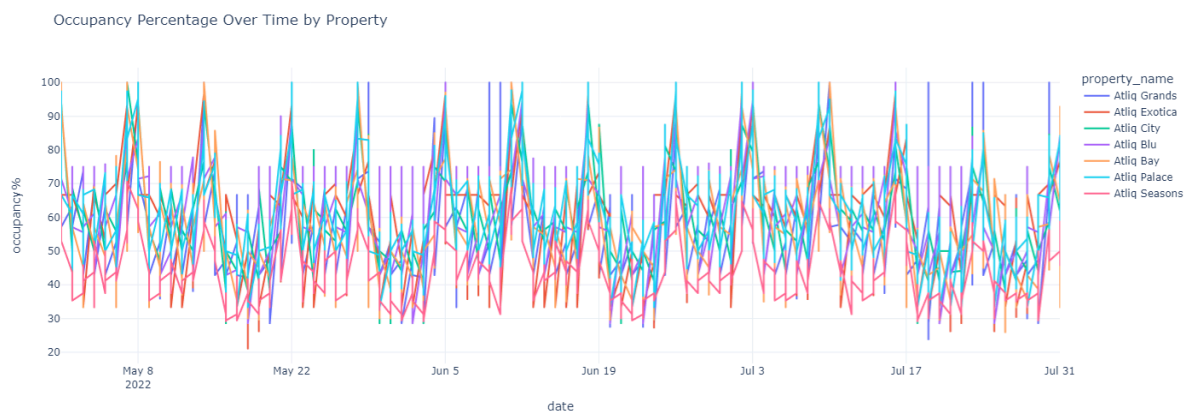
A box plot is a graphical representation that displays the distribution of a dataset based on a five-number summary: minimum, first quartile, median, third quartile, and maximum. It is useful for identifying the central tendency, dispersion, and potential outliers within different groups. In the visualization, the x-axis represents the room class, while the y-axis shows the occupancy percentage. The boxes are color-coded by city, providing a comparison of occupancy percentage distributions across different room classes and cities. Each box represents the interquartile range (IQR) of occupancy percentages for a specific room class in a city, with the median shown as a line within the box. Outliers beyond this range are displayed as individual points, highlighting any extreme values in occupancy percentages for each combination of room class and city.

Line graph



This line graph illustrates the trend of successful bookings over time, categorized by weekday and weekend, across different cities. Each line represents either weekday or weekend bookings, distinguished by line style (solid vs. dashed). The x-axis displays dates, while the y-axis represents the count of successful bookings. This visualization helps in comparing booking patterns between weekdays and weekends within each city, highlighting any variations or trends over time.

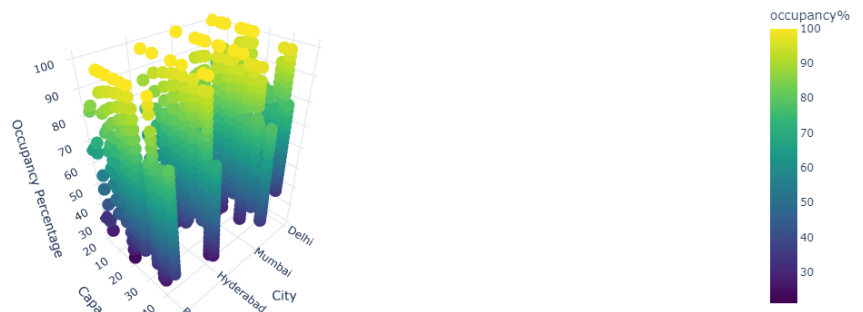
Line chart



A line chart is a graphical representation that displays information as a series of data points connected by straight line segments. It is useful for showing trends over time. In the visualization, the x-axis represents the date, while the y-axis shows the occupancy percentage. Each line corresponds to a different property, color-coded for distinction. This line chart helps in tracking how the occupancy percentage changes over time for each property, making it easier to identify trends, patterns, or seasonal variations in occupancy rates across different properties.

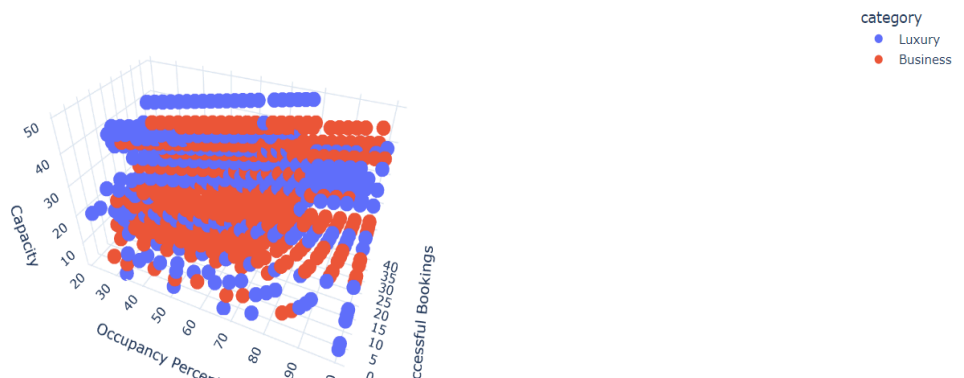
3D Scatter plot

Occupancy Rates vs. Capacity Across Cities



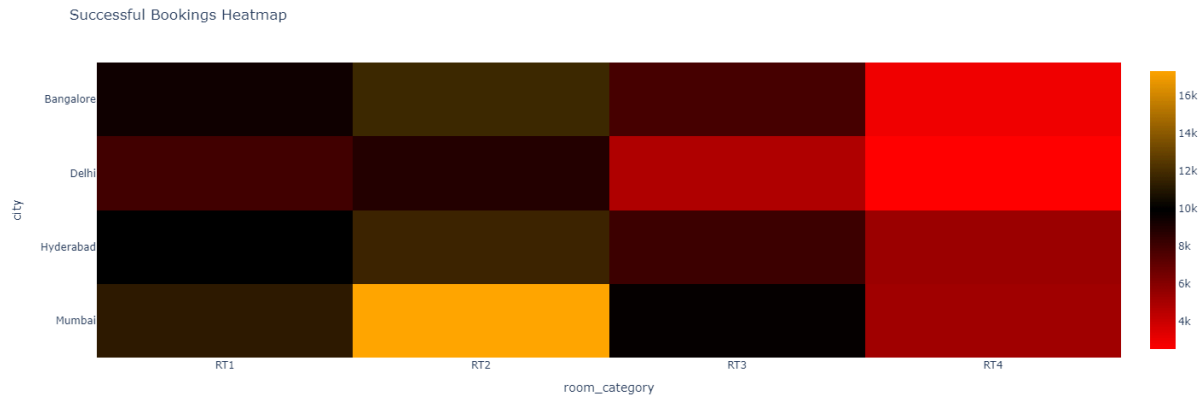
This visualization is a 3D scatter plot, which depicts relationships among three variables using points in a three-dimensional space. In this 3D scatter plot, each point represents a city positioned based on its room capacity (y-axis) and occupancy percentage (z-axis). The x-axis categorizes cities, providing a clear view of how occupancy rates vary with capacity across different urban areas. The color of each point indicates occupancy percentage intensity, ranging from low (purple) to high (yellow) using the Viridis color scale. This visualization enables immediate comparisons of occupancy trends and capacity utilization across various cities, helping identify potential areas for optimization and improvement.

Booking Success by Occupancy, Capacity, & Category



This 3D scatter plot visualizes booking success by occupancy percentage (x-axis), number of successful bookings (y-axis), and room capacity (z-axis), differentiated by room category (colour-coded). Each point in the plot represents a specific combination of these variables for different categories of accommodations. The visualisation enables the exploration of how occupancy rates, successful bookings, and capacity interrelate across different room categories. It provides insights into which combinations of occupancy, capacity, and category tend to have higher or lower booking success rates. The title of the plot is "Booking Success by Occupancy, Capacity, & Category", with axes labelled for clarity to facilitate interpretation of the data.

Heatmap



A heatmap visually represents the matrix of successful bookings across different cities and room categories. Each cell's color intensity corresponds to the number of successful bookings, using a custom color scale defined as red, black, and orange. Darker colors indicate higher booking counts. This visualization allows for easy identification of patterns, such as which room categories and cities have higher booking numbers.

Python Visualisation:

Python offers more flexibility and customization options for creating visualisations using libraries like Matplotlib and Seaborn.

It is preferred for in-depth statistical analysis, custom analytical methods, complex calculations, and advanced machine learning techniques that may not be feasible in Power BI or Tableau.

Python is helpful when data analysis requires specialised statistical analyses, predictive modelling, or advanced machine learning techniques.

Python visualisations are static images and cannot be easily shared with others who don't have Python installed.

Python is better suited for smaller to medium-scale projects and is more challenging to use for creating interactive dashboards compared to Power BI or Tableau.

When to Choose Power BI vs Python for Visualization

Choose Power BI when you need to create interactive dashboards, track real-time metrics, and share insights with non-technical users.

Choose Python when you require specialised calculations, custom analytical methods, complex statistical models, or advanced machine learning techniques.

Python is preferred for in-depth statistical analysis, hypothesis testing, and handling diverse data sources with complex transformations.

A combination of both tools is often used, with Python for data cleaning, preprocessing, and building advanced models, and Power BI for creating visualisations and sharing insights.

In summary, Power BI is a powerful tool for creating interactive dashboards and communicating insights to stakeholders, while Python offers more flexibility and customization options for advanced data analysis and visualisation tasks.

2. Getting Started

Setting Up Power BI:

- ***Download Power BI Desktop***
 4. Go to the official Power BI website and click on the "Download Free" button.
 5. Select the appropriate version (32-bit or 64-bit) based on your system architecture.
 6. Click "Next" to download the Power BI Desktop setup file to your computer.

- ***Install Power BI Desktop***
 7. Open the downloaded setup file.
 8. Click "Next" in the installation wizard to proceed.
 9. Accept the licence agreement by checking the box and clicking "Next".
 10. Choose the installation folder or use the default location, then click "Next".
 11. Review the installation settings and click "Install" to begin the installation process.
 12. Wait for the installation to complete, then click "Finish" to launch Power BI Desktop.

- ***Set Up Power BI Desktop***
 9. When prompted, you can fill out the form with your name, company, email, and job title, or simply close the window.
 10. If asked to sign in, you can skip this step for now.
 11. You will now see the Power BI Desktop interface, which includes a ribbon with tabs like "Home," "View," "Modelling," and "Help".
 12. On the left side, you'll find three layouts: "Report," "Data," and "Model".
 13. In the "Report" layout, you can build your reports and dashboards.
 14. The "Data" layout shows your uploaded or created data tables.
 15. The "Model" layout allows you to create relationships between multiple tables based on common columns.
 16. On the right side of the "Report" layout, you'll find sections for "Filters," "Visualisations," and "Fields".

Installing Python and Necessary Libraries

To install Python and essential libraries like Pandas, NumPy, Matplotlib, Seaborn, and Plotly for data analysis and visualisation, follow these detailed steps:

- **Installing Python**

4. Visit the official Python website and download the latest version of Python for your operating system.
5. Run the downloaded Python installer and follow the installation wizard instructions.
6. During installation, ensure to check the box that adds Python to your system's PATH to make it easier to run Python from the command line.

Installing Essential Libraries

Pandas, NumPy, Matplotlib, Seaborn, and Plotly

3. Open a command prompt or terminal after installing Python.
4. Use the following commands to install each library using Python's package manager, pip:

```
pip install pandas
```

```
pip install numpy
```

```
pip install matplotlib
```

```
pip install seaborn
```

```
pip install plotly
```

3. Verify the installation of each library by importing them in a Python script or Jupyter Notebook.

By following these steps, you will successfully install Python and the essential libraries Pandas, NumPy, Matplotlib, Seaborn, and Plotly for data analysis and visualisation.

3.Data Preparation

- **Data Acquisition**

Databases

4. Connect to databases like SQL Server, MySQL, PostgreSQL, etc. using database connectors or drivers
5. Write SQL queries to extract data from database tables and views
6. Use database APIs or libraries in programming languages like Python, R, Java, etc. to interact with databases

CSV and Excel Files

4. Read CSV files using libraries like pandas in Python or readr in R
5. Import data from Excel spreadsheets using language-specific libraries
6. Specify the file path or URL of the CSV/Excel file to read the data into a data frame or tibble

APIs

4. Make HTTP requests to API endpoints to retrieve data in formats like JSON or XML
5. Use API client libraries in programming languages to simplify API interactions
6. Handle authentication (API keys, OAuth, etc.) and pagination when making API requests

Web Scraping

4. Use web scraping libraries like BeautifulSoup in Python or rvest in R to parse HTML and extract data
5. Identify the relevant HTML elements containing the desired data using CSS selectors or XPath
6. Implement web scraping while respecting robots.txt and avoiding excessive requests to avoid overloading websites

Other techniques include:

5. Connecting to cloud storage services like Google Drive, Dropbox, etc. to access files
6. Reading data from real-time data streams or event logs
7. Importing data from statistical software formats like SPSS, SAS, Stata, etc.
8. The choice of data acquisition method depends on the data source, format, and accessibility. It's important to understand the data structure and ensure data quality during the acquisition process.

● Data Cleaning and Preprocessing:

To handle missing values, duplicates, outliers, and inconsistencies in data during the data cleaning and preprocessing process, several strategies can be employed:

Missing Values:

4. Identification: Use functions like `is.na()` in R or similar methods in Python to detect missing values.
5. Imputation: Fill missing values using techniques like mean, median, mode imputation, or advanced methods like k-Nearest Neighbors (KNN) imputation.
6. Removal: Eliminate rows or columns with excessive missing values using functions like `na.omit()`.

Duplicates:

3. Detection: Identify duplicate entries that overrepresent data by checking for identical records.
4. Handling: Decide whether to remove duplicates or merge conflicting information to ensure data integrity.

Outliers:

3. Detection: Use methods like the IQR method or Z-score method to identify extreme values that deviate significantly from the majority of data points.
4. Handling: Address outliers by removing them, transforming the data, or using robust statistical methods less sensitive to outliers.

Inconsistencies:

3. Identification: Look for inconsistencies in data formats, naming conventions, or variable types that can lead to errors in analysis.
4. Correction: Standardised formats, correct naming discrepancies, and ensure uniformity in data representation to enhance data consistency.

Best Practices:

4. Documentation: Maintain a record of changes made during the cleaning process for transparency and reproducibility.
5. Validation: Validate and test the effectiveness of data cleaning steps iteratively to ensure data quality.
6. Strategic Decisions: Make informed decisions on imputation, removal, or transformation based on the nature of the data and the analysis requirements.

By implementing these strategies, data scientists can ensure that the data is accurate, consistent, and free of errors, leading to more reliable and meaningful analysis results.

- **Data Transformation:**

Data transformation is the process of converting raw data into a structured, standardised format that is suitable for analysis and visualisation. This process involves several techniques to reshape, aggregate, and combine data to make it more meaningful and useful for decision-making. Here are some key methods and techniques used in data transformation:

Reshaping Data

4. Smoothing: Removes noise from the dataset to highlight important features and patterns.
5. Data Generalization: Converts low-level attributes to high-level attributes to create a clear data snapshot.
6. Data Manipulation: Changes or alters data to make it more readable and organised.

Aggregating Data

3. Data Aggregation: Combines data from multiple sources into a single format for accurate analysis and reporting.
4. Data Discretization: Converts continuous data into discrete categories or bins to improve efficiency and easier analysis.

Combining Data

3. Data Integration: Combines data from multiple sources, such as databases and spreadsheets, into a single format.
4. Data Normalisation: Scales the data to a common range of values to facilitate comparison and analysis.

Other Techniques

3. Attribute Construction: Creates new attributes from existing attributes to ease data mining and analysis.
4. Log Transformation: Transforms data by taking the logarithm of the data to stabilise variance and improve normality.

Benefits of Data Transformation

6. Improved Data Quality: Removes errors, inconsistencies, and missing values.
7. Facilitates Data Integration: Enables integration of data from multiple sources.
8. Improves Data Analysis: Prepares data for analysis and modelling by normalising, reducing dimensionality, and discretizing data.
9. Increases Data Security: Masks sensitive data or removes sensitive information to increase security.
10. Enhances Data Mining Algorithm Performance: Reduces dimensionality and scales data to improve algorithm performance.

Challenges and Considerations

6. Time-Consuming: Data transformation can be a time-consuming process, especially for large datasets.
7. Complexity: Data transformation can be complex, requiring specialised skills and knowledge.
8. Data Loss: Data transformation can result in data loss, such as when discretizing continuous data.
9. Biased Transformation: Data transformation can result in bias if not properly understood or used.
10. High Cost: Data transformation can be expensive, requiring significant investments in hardware, software, and personnel.

Tools and Libraries

4. Python Libraries: Pandas, NumPy, Matplotlib, Seaborn, Plotly
5. R Libraries: readr, dplyr, tidyr, ggplot2
6. SQL: SQL Server, MySQL, PostgreSQL

Best Practices

5. Documentation: Maintain a record of changes made during the transformation process for transparency and reproducibility.
6. Validation: Validate and test the effectiveness of transformation steps iteratively to ensure data quality.
7. Strategic Decisions: Make informed decisions on imputation, removal, or transformation based on the nature of the data and the analysis requirements.
- 8.

By understanding these methods and techniques, data scientists can effectively transform raw data into a structured and standardised format, making it suitable for analysis and visualisation.

- **Data Analysis Techniques**

Regression Analysis

4. Analyses the relationship between one or more independent variables and a dependent variable
5. Used to explain how changes in the independent variables influence the dependent variable
6. Types include simple linear regression, multiple linear regression, logistic regression, etc.

Cluster Analysis

4. Groups data points into clusters based on similarity
5. Helps identify patterns and relationships in complex datasets
6. Commonly used for customer segmentation and anomaly detection

Time Series Analysis

4. Tracks data over time to identify trends, patterns, and relationships
5. Used for forecasting, anomaly detection, and understanding cyclical behavior
6. Techniques include moving averages, exponential smoothing, ARIMA models, etc.

Factor Analysis

4. Reduces a large number of variables to a smaller set of underlying factors
5. Uncovers hidden patterns and relationships in complex data
6. Helps with dimensionality reduction and feature extraction

Cohort Analysis

4. Breaks down data into related groups (cohorts) for deeper analysis

5. Allows comparing the behaviour of different cohorts over time
6. Commonly used in marketing and product analytics

Monte Carlo Simulation

4. Models the probability of different outcomes in a process that cannot easily be predicted
5. Incorporates multiple variables and scenarios to assess risk
6. Used for risk analysis, forecasting, and optimization problems

Sentiment Analysis

4. Analyses text data to determine the sentiment (positive, negative, neutral) expressed
5. Helps understand customer opinions, emotions, and attitudes
6. Widely used in social media monitoring and customer service

The choice of data analysis technique depends on the type of data, the research question, and the desired insights. Combining multiple techniques can provide a more comprehensive understanding of the data.

Conclusion :

In conclusion, the integration of Python and Power BI offers a robust and versatile solution for data analytics, combining the strengths of Python's extensive libraries and programming flexibility with Power BI's powerful data visualization and business intelligence capabilities. Python's libraries such as Pandas, NumPy, and Matplotlib enable data analysts to perform complex data manipulations, statistical analysis, and generate intricate visualizations, while machine learning libraries like Scikit-learn and TensorFlow allow for predictive analytics and advanced modeling. On the other hand, Power BI simplifies the process of creating interactive and shareable dashboards that can integrate data from various sources, providing real-time insights and fostering data-driven decision-making across organizations. The synergy between Python's scripting power and Power BI's visualization proficiency allows analysts to seamlessly process and interpret large datasets, uncovering actionable insights that drive strategic initiatives and operational efficiencies. As data continues to grow in volume and complexity, mastering both Python and Power BI equips data professionals with the essential tools to navigate the evolving landscape of data analytics, ultimately contributing to more informed and impactful business decisions.

