Eco Industry
Oriented University

sinarmas

SAMPLING AND SURVEY TECHNIQUES

Study Case Using R and Python

Writer:

Bakti Siregar, M.Sc., CDS.

D'SCIENCELABS
Smart Idea

Kampus Merdeka
INDONESIA JAYA

First Edition

Sampling and Survey Techniques

Study Case Using R and Python

Bakti Siregar, M.Sc.,CDS

Table of contents

In today's data-driven world, the ability to collect, analyze, and interpret information accurately is more important than ever. Sampling and survey techniques are essential tools in research, business, social sciences, and public policy, allowing researchers to gather representative data, identify patterns, and make well-informed decisions. A well-structured survey, combined with appropriate sampling methods, enhances data reliability while minimizing bias and errors.

The foundation of effective surveys lies in selecting the right sampling strategy. Probability-based sampling methods, such as simple random sampling, stratified sampling, and cluster sampling, provide statistically valid insights, whereas non-probability techniques, including convenience sampling and quota sampling, offer practical advantages in specific research contexts. Understanding these methods allows researchers to optimize data collection while ensuring the accuracy and credibility of their findings.

Beyond data collection, survey methodology involves data validation, analysis, and interpretation. Statistical techniques such as confidence intervals, regression analysis, and hypothesis testing enhance the quality of survey results, allowing researchers to derive meaningful insights. Additionally, modern digital tools and automation have transformed survey research, improving efficiency and accessibility.

This book provides a comprehensive and practical guide to sampling and survey techniques, covering the fundamental principles of data collection, survey design, statistical analysis, and visualization. By mastering these concepts, researchers, analysts, and decision-makers will be better equipped to design effective surveys, interpret survey data accurately, and apply the results to real-world decision-making processes.

Preface

About the Writer

Bakti Siregar, M.Sc., CDS works as a Lecturer at the ITSB Data Science Program. He earned his Master's degree from the Department of Applied Mathematics at National Sun Yat Sen University, Taiwan. In addition to teaching, Bakti also works as a Freelance Data Scientist for leading companies such as JNE, Samora Group, Pertamina, and PT. Green City Traffic.

He has a strong enthusiasm for projects (and teaching) in the fields of Big Data Analytics, Machine Learning, Optimization, and Time Series Analysis, particularly in finance and investment. His core expertise lies in statistical programming languages such as R Studio and Python. He is also experienced in implementing database systems like MySQL/NoSQL for data management and is proficient in using Big Data tools such as Spark and Hadoop. Some of his projects can be viewed here: Rpubs, Github, Website, and Kaggle.

Acknowledgments

First and foremost, I would like to express my deepest gratitude to God Almighty for granting me the strength, wisdom, and perseverance to complete this ebook on Sampling and Survey Techniques. Without His blessings, this endeavor would not have been possible.

I extend my sincere appreciation to my mentors, colleagues, and friends who have provided invaluable guidance, encouragement, and constructive feedback throughout the

process. Your support has been instrumental in shaping the content of this ebook and ensuring its relevance and clarity.

A special thanks to my family for their unwavering patience, love, and motivation. Their belief in my abilities has been a constant source of inspiration, and their support has been indispensable in completing this work.

I am also grateful to all the researchers, authors, and experts whose work has contributed to the knowledge base from which this ebook draws. Their insights have greatly influenced the quality and depth of this publication.

Lastly, I would like to extend my appreciation to my readers. Your interest and engagement motivate me to continue exploring and sharing knowledge. I hope this ebook on Sampling and Survey Techniques serves as a valuable resource for you in your learning journey.

Feedback & Suggestions

Your feedback is essential in improving this book. We invite all readers to share their thoughts on the content, structure, and clarity of the materials. Suggestions for additional topics or areas requiring further explanation are highly appreciated.

With your support and contributions, we aim to refine this book, making it a more comprehensive resource for **Sampling and Survey Techniques**. Thank you for your participation!

For feedback and suggestions, feel free to contact:

- dsciencelabs@outlook.com
- siregarbakti@gmail.com
- siregarbakti@itsb.ac.id

Chapter 1

Principles of Sampling

1.1 What is Sampling?

Sampling is the process of selecting a subset of individuals, items, or observations from a larger population to estimate characteristics of the whole population. It is widely used in research, business, and public policy to make data-driven decisions efficiently.

1.2 Population vs. Sample

In statistics, understanding the distinction between **population** and **sample** is crucial for data analysis, inference, and decision-making.

1.2.1 Population

The **population** (N) is the **entire group** of individuals, objects, or events that a researcher is interested in studying. It includes **all possible observations** relevant to the research. **Examples:**

- All residents of a city when studying voting behavior.
- Every manufactured smartphone from a factory when analyzing defect rates.
- Every student in a university when measuring average exam scores.

Types of Populations:

- **Finite Population:** A population with a fixed number of elements (e.g., employees in a company).
- **Infinite Population:** A population with an uncountable number of elements (e.g., bacteria in a petri dish).
- **Target Population:** The specific population a researcher wants to study.
- *Accessible Population:** The portion of the target population available for study.

1.2.2 Sample

A **sample** (n) is a **subset** of the population, selected for analysis. Since studying an entire population is often impractical due to cost, time, or accessibility, a sample is used to make inferences about the population. **Examples:**

- Surveying 1,000 residents of a city to estimate public opinion.
- Inspecting 500 randomly chosen smartphones to assess defect rates.
- Analyzing exam scores from 200 randomly selected students.

Characteristics of a Good Sample:

- **Representative:** Accurately reflects the population.
- **Random:** Selected without bias.
- **Sufficiently Large:** Ensures reliable estimates.
- **Minimally Biased:** Avoids systematic errors.

1.2.3 Key Differences

When conducting research or statistical analysis, it is essential to distinguish between population and sample. The population refers to the entire group of interest in a study, while the sample is a smaller subset selected from that population for analysis. Understanding their differences is crucial for making accurate inferences and ensuring the validity of conclusions.

Here are the key differences between a population and a sample:

Feature	Population (N)	Sample (n)
Definition	Entire group of interest	A subset selected for study
Size	Large or infinite	Smaller, manageable portion
Notation	Uses uppercase letters (e.g., N, μ, σ)	Uses lowercase letters (e.g., n, $\bar{x}$, s)
Parameters	True values (e.g., population mean μ, standard deviation σ)	Estimates (e.g., sample mean $\bar{x}$, standard deviation s)
Cost & Time	High	Lower
Accuracy	Provides exact information	Provides an estimate with some margin of error

1.3 Why Use a Sample?

In research and data collection, studying an entire population is often impractical or impossible. Instead, researchers use a **sample**, which is a smaller, manageable subset of the population. Below are the key reasons for using a sample:

- **Cost-Effectiveness**
  Collecting data from an entire population requires significant financial resources. A sample reduces costs associated with data collection, processing, and analysis.

- **Time Efficiency**
  Studying an entire population is time-consuming. A well-chosen sample allows for quicker data collection and analysis.

- **Feasibility**
  Some populations are too large or inaccessible to study completely. A sample makes research possible when population-wide data collection is impractical.

- **Accuracy and Reliability**
  When selected properly, a sample can provide highly accurate and reliable insights. Statistical techniques ensure that the sample represents the entire population effectively.

- **Reduced Data Management Complexity**
  Handling vast amounts of data can be challenging. A sample simplifies data management while still providing meaningful conclusions.

- **Ethical Considerations**
  Some research (e.g., medical trials) may involve risks, making it unethical to test on an entire population. A sample allows for controlled and ethical experimentation.

1.4 Avoiding Sampling Bias

Sampling bias occurs when certain members of the population are systematically **excluded** or **overrepresented** in the sample.

This leads to inaccurate and unrepresentative results, potentially skewing conclusions and reducing the validity of a study.There are some sauses of sampling bias:

Aspect	Description	How to Overcome
Undercoverage	Some groups in the population are not included in the sampling frame.	Use a **representative sampling frame** to ensure all groups are covered.
Overrepresentation	Certain groups have a disproportionately higher chance of being selected.	Use **stratified sampling** to maintain balanced proportions.
Self-Selection Bias	Participants voluntarily choose to take part, leading to a non-random sample.	Use **randomized invitations** and consider **incentives** to attract a more diverse group of respondents.

Minimizing sampling bias is essential for producing valid, reliable, and generalizable research findings. By ensuring a well-constructed sampling frame, applying random selection methods, and reducing self-selection effects, researchers can improve the quality and accuracy of their studies.

1.5 Randomization in Sampling

Randomization is a process that ensures every member of a population has an **equal chance** of being selected. This reduces **sampling bias** and enhances the **generalizability** of research findings.

1.5.1 Simple Random Sampling

A method where each element in the population has an equal probability of selection, ensuring a truly random sample. Here, how it works:

- Assign a unique number to each member of the population.
- Use a random number generator or lottery system to select participants.

Example: A company wants to survey 500 employees from a workforce of 5,000. Each employee is assigned a number, and 500 are randomly chosen using a lottery system.

1.5.2 Systematic Sampling

A method where elements are selected at regular intervals from an ordered list. Here, how it works:

- Determine the sample size (e.g., selecting 100 people from a list of 1,000).
- Calculate the sampling interval: **Population Size ÷ Sample Size** (e.g., 1,000 ÷ 100 = 10).
- Randomly select a starting point and then pick every 10th person.

Example: A researcher wants to survey every 5th customer from a list of 1,000 shoppers. If the starting point is 3, the selected individuals will be 3rd, 8th, 13th, etc.

1.5.3 Stratified Sampling

A method that divides the population into **subgroups (strata)** based on a shared characteristic, then randomly selects a proportional number of participants from each stratum. Here, how it works:

- Identify relevant **strata** (e.g., age groups, income levels, education).
- Determine the proportion of each stratum in the population.
- Conduct **random sampling** within each stratum.

Example: A university wants to survey students from different academic years. If 40% of students are freshmen, 30% are sophomores, 20% are juniors, and 10% are seniors, then the sample will reflect these proportions.

Using random sampling methods like **SRS, systematic sampling, and stratified sampling** helps ensure a **fair, unbiased**, and **representative sample**. This improves the reliability and validity of research findings, making them more generalizable to the entire population.

1.6 Challenges in Sampling

Sampling is a critical process in research, but it comes with several challenges that can impact accuracy and reliability. Below is an overview of key sampling challenges along with their causes and possible solutions.

Challenge	Causes	Solutions
Non-Response Bias	Participants unwilling or unable to respond. Surveys too long or complex. Certain groups less likely to participate.	Send follow-up reminders. Offer incentives. Simplify survey format.
Sampling Frame Errors	Outdated or incomplete lists. Incorrect classification. Duplicate or ineligible participants included.	Keep the sampling frame updated. Cross-check data sources. Use stratified sampling.
Inadequate Sample Size	Limited resources for large samples. Miscalculated sample size. High dropout rates in longitudinal studies.	Use statistical methods to determine the correct sample size. Account for potential dropouts.
Cost and Time Constraints	High costs for data collection. Delays in reaching participants. Need for specialized tools or personnel.	Use cost-effective methods like online surveys. Automate data collection. Optimize resources.

Addressing these challenges ensures that the sampling process is more reliable, efficient, and representative of the target population. By implementing effective solutions, researchers can minimize errors and improve the overall quality of their studies.

1.7 Applications in Industry

Sampling plays a crucial role across various industries, allowing organizations to gather insights, make informed decisions, and optimize processes. Below are key areas where sampling is widely used:

Industry	Application	Purpose
Market Research	Conducting surveys and focus groups.	Understanding customer preferences, trends, and behaviors.
Healthcare	Studying patient data and clinical trials.	Estimating disease prevalence, treatment effectiveness, and public health trends.
Quality Control	Inspecting a subset of products in manufacturing.	Ensuring product quality and compliance with industry standards.
Finance	Analyzing financial transactions and market trends.	Assessing risks, detecting fraud, and making investment decisions.

By applying proper sampling techniques, industries can obtain **accurate and reliable insights** while minimizing errors and biases. This ensures better decision-making, cost savings, and improved operational efficiency.

Chapter 2

Probability Distributions

2.1 Probability in Sampling

Sampling is the process of selecting a subset of individuals from a population to make inferences about the entire population. It is widely used in statistics, surveys, and research studies to obtain insights without having to analyze every member of the population.

The two primary categories of sampling methods are **probability sampling** and **non-probability sampling**.

2.2 Probability vs. Non-Probability

In research and data collection, sampling methods are broadly categorized into probability sampling and non-probability sampling. The choice between these methods depends on the research goals, available resources, and the level of accuracy needed.

2.2.1 Probability Sampling

Probability sampling ensures that every element in the population has a known, non-zero chance of being selected. It allows for statistical inference and generalization to the entire population. Common probability sampling methods include:

- **Simple Random Sampling (SRS)**: Each element in the population has an equal chance of being selected. This can be done using random number generators or lottery methods.
- **Stratified Sampling**: The population is divided into strata (subgroups) based on certain characteristics (e.g., age, income), and random samples are drawn from each stratum proportionally.
- **Cluster Sampling**: The population is divided into clusters (e.g., geographic regions), and entire clusters are randomly selected. This is useful for large populations.
- **Systematic Sampling**: A starting point is randomly chosen, and subsequent selections follow a fixed interval (e.g., selecting every 10th person).

2.2.2 Non-Probability Sampling

Non-probability sampling does not guarantee every individual in the population has a chance of being selected. It is often used when probability sampling is impractical or too expensive. Common non-probability sampling methods include:

- **Convenience Sampling**: Selecting individuals based on availability or accessibility.
- **Judgmental (Purposive) Sampling**: Selecting individuals based on researcher judgment and expertise.
- **Quota Sampling**: Ensuring specific subgroups are represented in the sample based on predetermined quotas.
- **Snowball Sampling**: Participants recruit other participants, often used in hard-to-reach populations.

2.3 Types of Sampling Distributions

A **sampling distribution** refers to the probability distribution of a statistic (such as the mean, proportion, variance, or standard deviation) obtained from multiple random samples of the same size from a population. These distributions are essential in inferential statistics, as they help estimate population parameters and test hypotheses.

2.3.1 Mean

This distribution consists of the means of all possible random samples of a given size from a population.

Key Properties:

- The mean of the sampling distribution ($\mu_{\bar{x}}$) is equal to the population mean (μ).
- The standard deviation of the sampling distribution (Standard Error of the Mean, **SEM**) is given by:

$$\sigma_{\bar{x}} = \frac{\sigma}{\sqrt{n}}$$

where σ is the population standard deviation and n is the sample size.

- If the population is normally distributed, the sampling distribution is also normal for any n.
- If the population is not normal, the **Central Limit Theorem (CLT)** states that the sampling distribution of the mean will be approximately normal if $n \geq 30$.

Example: A population has a mean of 100 and a standard deviation of 15. If we take random samples of size 36, the standard error of the mean will be:

$$\sigma_{\bar{x}} = \frac{15}{\sqrt{36}} = \frac{15}{6} = 2.5$$

If the population follows a normal distribution, the sample means will also follow a normal distribution with a mean of 100 and a standard deviation of 2.5.

2.3.2 Proportion

This distribution describes the possible values of sample proportions from a population.

Key Properties:

- The mean of the sampling distribution of proportions is equal to the population proportion (p).
- The standard deviation (Standard Error of the Proportion, **SEP**) is:

$$\sigma_{\hat{p}} = \sqrt{\frac{p(1-p)}{n}}$$

- If $np \geq 5$ and $n(1-p) \geq 5$, the sampling distribution of the proportion is approximately normal (by the normal approximation to the binomial).

Example: If 40% ($p = 0.4$) of a population supports a certain policy, and a random sample of 100 is taken:

$$\sigma_{\hat{p}} = \sqrt{\frac{0.4(1-0.4)}{100}} = \sqrt{\frac{0.24}{100}} = \sqrt{0.0024} \approx 0.049$$

The sample proportions will follow an approximately normal distribution with a mean of 0.4 and a standard deviation of 0.049.

2.3.3 Variance

This distribution describes the variability of sample variances.

Key Properties:

- The mean of the sampling distribution of variance is equal to the population variance (σ^2).
- The sampling distribution follows a **chi-square distribution** with $(n-1)$ degrees of freedom.
- The formula for the sample variance is:

$$s^2 = \frac{\sum(x_i - \bar{x})^2}{n-1}$$

Example: If a population has a variance of 25 and a sample size of 10, the sample variances will follow a chi-square distribution with 9 degrees of freedom.

2.3.4 Standard Deviation

Since the sample variance follows a chi-square distribution, the standard deviation is derived from it.

Key Properties:

- The sampling distribution of standard deviation does not follow a normal distribution.
- It is often estimated using the chi-square distribution.

2.3.5 Difference Between Two Means

Used when comparing two independent sample means.

Key Properties:

- If $\bar{x}_1$ and $\bar{x}_2$ are the means from two independent samples, the mean of their sampling distribution is:

$$\mu_{\bar{x}_1-\bar{x}_2} = \mu_1 - \mu_2$$

- he standard deviation (Standard Error) is:

$$\sigma_{\bar{x}_1-\bar{x}_2} = \sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}$$

- If both populations are normal or the sample sizes are large, the distribution of $\bar{x}_1 - \bar{x}_2$ is approximately normal.

Example: If two populations have means of 50 and 55 with variances of 16 and 25, and sample sizes of 30 each:

$$\sigma_{\bar{x}_1-\bar{x}_2} = \sqrt{\frac{16}{30} + \frac{25}{30}} = \sqrt{0.533 + 0.833} = \sqrt{1.366} \approx 1.17$$

2.3.6 Difference Between Two Proportions

Used when comparing proportions from two independent samples.

Key Properties:

- The mean of the sampling distribution is:

$$\mu_{\hat{p}_1-\hat{p}_2} = p_1 - p_2$$

- The standard deviation (Standard Error) is:

$$\sigma_{\hat{p}_1-\hat{p}_2} = \sqrt{\frac{p_1(1-p_1)}{n_1} + \frac{p_2(1-p_2)}{n_2}}$$

- If sample sizes are large, the distribution is approximately normal.

Example: If $p_1 = 0.6$ and $p_2 = 0.5$ with sample sizes of 100 each:

$$\sigma_{\hat{p}_1-\hat{p}_2} = \sqrt{\frac{0.6(0.4)}{100} + \frac{0.5(0.5)}{100}} = \sqrt{0.0024 + 0.0025} = \sqrt{0.0049} = 0.07$$

2.3.7 Student's t-Distribution

Used when estimating the mean of a normally distributed population with an unknown variance, especially for small samples ($n < 30$).

Key Properties:

- The shape is similar to a normal distribution but has heavier tails (greater variability for small samples).
- The formula for the **t-statistic** is:

$$t = \frac{\bar{x} - \mu}{\frac{s}{\sqrt{n}}}$$

- Follows a **t-distribution** with $n - 1$ degrees of freedom.

Example: If $\bar{x} = 52$, $\mu = 50$, $s = 10$, and $n = 9$, the t-score is:

$$t = \frac{52 - 50}{\frac{10}{\sqrt{9}}} = \frac{2}{\frac{10}{3}} = \frac{2}{3.33} = 0.6$$

Each type of sampling distribution serves a specific purpose in statistical inference, from estimating means and proportions to comparing groups. Understanding these distributions is crucial for hypothesis testing, constructing confidence intervals, and making data-driven decisions.

2.4 Standard Normal Distribution

The **Z-distribution** (or **standard normal distribution**) is a normal distribution with a mean of **0** and a standard deviation of **1**. It is used for standardizing data, hypothesis testing, and confidence intervals.

Key Properties:

- The **mean** (μ) is **0** and the **standard deviation** (σ) is **1**.
- The **total area** under the curve is **1**.
- The **Z-score formula** converts raw values into standard normal values:

$$Z = \frac{X - \mu}{\sigma}$$

where:

- X = observed value
- μ = population mean
- σ = population standard deviation

Empirical Rule (68-95-99.7 Rule):

- About **68%** of values fall within ± 1 **standard deviation**.

- About **95%** of values fall within ± 2 **standard deviations**.
- About **99.7%** of values fall within ± 3 **standard deviations**.

Example: If a test score is $X = 85$, the population mean is $\mu = 75$, and the standard deviation is $\sigma = 10$, then:

$$Z = \frac{85 - 75}{10} = \frac{10}{10} = 1.0$$

This means the test score is **1 standard deviation above the mean**.

The Z-distribution is widely used in **Z-tests**, probability calculations, and constructing **confidence intervals** for population parameters.

2.5 Central Limit Theorem (CLT)

The **Central Limit Theorem (CLT)** states that for a sufficiently large sample size (typically $n > 30$), the sampling distribution of the mean will be approximately normal, regardless of the original population distribution.

Implications of CLT:

- Allows normal approximation even for skewed population distributions.
- Enables hypothesis testing and confidence interval estimation using normal-based methods.

2.6 Law of Large Numbers

The **Law of Large Numbers (LLN)** states that as the sample size increases, the sample mean approaches the population mean.

- **Weak Law of Large Numbers**: The probability that the sample mean deviates significantly from the population mean decreases as sample size increases.
- **Strong Law of Large Numbers**: The sample mean converges almost surely to the population mean as the sample size grows.

2.7 Confidence Intervals

A **Confidence Interval (CI)** provides a range of values that likely contain the true population parameter. The formula for a confidence interval for a population mean is:

$$CI = \bar{X} \pm Z_{\alpha/2} \times \frac{\sigma}{\sqrt{n}}$$

where:

- $Z_{\alpha/2}$ is the critical value from the standard normal table.
- σ is the population standard deviation.
- n is the sample size.

Common confidence levels:

- **90% CI**: $Z = 1.645$
- **95% CI**: $Z = 1.96$
- **99% CI**: $Z = 2.576$

2.8 Hypothesis Testing in Surveys

Hypothesis testing is used to make inferences about population parameters based on sample data. The general steps include:

- **Define Hypotheses:**
  - Null Hypothesis (H_0): Assumes no effect or no difference.
  - Alternative Hypothesis (H_1): Assumes an effect or difference exists.
- **Select a Significance Level (α)**
  - Common choices: 0.05, 0.01, or 0.10.
- **Compute the Test Statistic**
  - For mean: $Z = \frac{\bar{X} - \mu}{\sigma/\sqrt{n}}$
  - For proportion: $Z = \frac{\hat{p} - p_0}{\sqrt{p_0(1-p_0)/n}}$
- **Determine the p-value**
  - If $p < \alpha$, reject H_0; otherwise, fail to reject H_0.
- **Draw a Conclusion**
  - If H_0 is rejected, there is sufficient evidence to support H_1.
  - If H_0 is not rejected, there is insufficient evidence to support H_1.

Understanding these fundamental statistical concepts allows researchers to design better experiments, analyze survey data effectively, and make informed conclusions based on sampled data.

Chapter 3

Sampling Methods

Sampling is the process of selecting a subset of individuals from a larger population to make statistical inferences. It can be broadly categorized into Probability Sampling and Non-Probability Sampling.

3.1 Probability Sampling

Probability sampling ensures that every individual in the population has a known, nonzero chance of being selected. This allows for generalizable and unbiased results.

3.1.1 Simple Random Sampling

Simple Random Sampling is a method where each individual in the population has an **equal chance** of being chosen. This technique ensures that the sample is **random and unbiased** by using random selection methods.

Characteristics:

- **Equal Probability** → Every individual has the same chance of being selected.
- **No Specific Pattern** → The selection process is entirely **random**.
- **Objective Representation** → The method avoids bias and ensures a fair representation of the population.

There are two primary ways to perform random sampling:

Using a Random Number Generator

Suppose we have 1000 students in a university, and we need a random sample of 100 students. The steps are:

- Assign numbers from 1 to 1000 to each student.
- Use a **random number generator** to select 100 unique numbers.

- The students corresponding to those numbers will be included in the sample.

Python Code:

```python
import pandas as pd

# Create a dataset (example: student data)
students = pd.DataFrame({'ID': range(1, 1001),
                         'Name': ['Student ' + str(i) for i in range(1, 1001)]})

# Set seed for reproducibility
random_state = 123

# Randomly select 100 students from the dataset
sample_students = students.sample(n=100, random_state=random_state)

# Print the selected sample
print(sample_students)
```

```
      ID         Name
131  132  Student 132
203  204  Student 204
50    51   Student 51
585  586  Student 586
138  139  Student 139
..   ...          ...
938  939  Student 939
814  815  Student 815
994  995  Student 995
805  806  Student 806
558  559  Student 559

[100 rows x 2 columns]
```

R Code:

```r
# Load dataset (contoh: data mahasiswa)
students <- data.frame(ID = 1:1000,
                       Name = paste("Student", 1:1000))

# Set seed for reproducibility
set.seed(123)

# Randomly select 100 students from the dataset
sample_students <- students[sample(nrow(students), 100, replace = FALSE), ]

# Print the selected sample
print(head(sample_students))
```

```
     ID        Name
415 415 Student 415
463 463 Student 463
```

```
179 179 Student 179
526 526 Student 526
195 195 Student 195
938 938 Student 938
```

Lottery Method

The Lottery Method is one of the Simple Random Sampling techniques where each individual in the population has an equal chance of being selected. This method is called "Lottery" because it resembles a lottery system, such as a raffle or prize draw, where names or **numbers are placed in a container**, **shuffled**, and **randomly drawn**.

Python Code Lottery Method:

```python
import random

# List of students (example names)
students = ["Syifa","Nabila", "Alya", "Isnaini", "Bagas",
              "Alfayed","Shalfa","Olivia","Nabila", "Fika",
              "Luthfi","Nabil", "Joans", "Riyadh", "Rachelia",
              "Nova", "Zain", "Ragil", "Dadan", "Dwi", "Chello", "Siti"]

# Number of samples to draw
num_samples = 5

# Shuffle the list (simulating shuffling the papers)
random.shuffle(students)

# Randomly draw the required number of samples
selected_students = random.sample(students, num_samples)

# Print the selected names
print("Selected students:", selected_students)
```

```
Selected students: ['Zain', 'Siti', 'Riyadh', 'Dwi', 'Ragil']
```

R Code Lottery Method:

```r
# List of students (example names)
students <- c("Syifa","Nabila", "Alya", "Isnaini", "Bagas",
              "Alfayed","Shalfa","Olivia","Nabila", "Fika",
              "Luthfi","Nabil", "Joans", "Riyadh", "Rachelia",
              "Nova", "Zain", "Ragil", "Dadan", "Dwi", "Chello", "Siti")

# Number of samples to draw
num_samples <- 5

# Shuffle the list (simulating shuffling the papers)
students <- sample(students)

# Randomly draw the required number of samples
```

```r
selected_students <- sample(students, num_samples)

# Print the selected names
print(selected_students)
```

```
[1] "Nabila"  "Luthfi"  "Syifa"   "Alya"    "Alfayed"
```

Here is the advantages and disadvantages in **Simple Random Sampling**:

Advantages	Disadvantages
Minimizes Bias → Every individual has an equal chance, making the process **fair**.	**Requires a Complete Population List** → A full database of individuals is needed.
Simple to Implement → Especially with software tools.	**Inefficient for Large Populations** → If done manually, it can be time-consuming.
Applicable to Large Populations → Works well with technology-assisted selection.	**Might Not Ensure Proportional Representation** → Some subgroups may be underrepresented by pure randomness.

Simple Random Sampling is a **fair, easy, and objective** method for selecting representative samples in research, surveys, and experiments. It is highly effective when a **complete list of the population** is available.

3.1.2 Systematic Sampling

Systematic Sampling is a probabilistic sampling technique where elements are selected from a population at fixed intervals (k) after choosing a random starting point. Instead of selecting samples purely at random, this method follows a structured approach, making it more efficient and easier to implement than Simple Random Sampling.

Python Code: Systematic Sampling

```python
import numpy as np
import pandas as pd

# Create a sample population dataset
data = pd.DataFrame({'Student_ID': np.arange(1, 101),
                     'Name': ['Student_' + str(i) for i in range(1, 101)]})

# Define sample size and interval
N = len(data)  # Population size
n = 10         # Desired sample size
k = N // n     # Sampling interval

# Randomly choose a starting point
np.random.seed(42)
start = np.random.randint(0, k)
```

```
# Select every k-th element
systematic_sample = data.iloc[start::k]

# Display results
print("Selected Sample:")
```

```
Selected Sample:
```

```
print(systematic_sample)
```

```
    Student_ID        Name
6            7   Student_7
16          17  Student_17
26          27  Student_27
36          37  Student_37
46          47  Student_47
56          57  Student_57
66          67  Student_67
76          77  Student_77
86          87  Student_87
96          97  Student_97
```

R Code: Systematic Sampling

```
set.seed(42)

# Create a sample population dataset
data <- data.frame(
  Student_ID = 1:100,
  Name = paste("Student", 1:100, sep = "_")
)

# Define sample size and interval
N <- nrow(data)  # Population size
n <- 10          # Desired sample size
k <- N %/% n     # Sampling interval

# Randomly choose a starting point
start <- sample(1:k, 1)

# Select every k-th element
systematic_sample <- data[seq(start, N, by = k), ]

# Display results
print(systematic_sample)
```

```
   Student_ID       Name
1           1  Student_1
11         11 Student_11
21         21 Student_21
```

```
31          31 Student_31
41          41 Student_41
51          51 Student_51
61          61 Student_61
71          71 Student_71
81          81 Student_81
91          91 Student_91
```

Key Concern: The Risk of Hidden Patterns

If the population follows a specific pattern or cycle, Systematic Sampling may introduce bias. For example, if an employee work schedule is arranged in a repeating morning-afternoon-night shift pattern and we select every 3rd employee, we might only sample morning-shift workers, leading to biased results. To mitigate this risk, researchers should check for patterns in the population before applying Systematic Sampling.

3.1.3 Stratified Sampling

Stratified Sampling is a **probability sampling technique** in which the population is divided into **subgroups (strata)** based on shared characteristics. A sample is then drawn **proportionally** from each stratum to ensure that all groups are adequately represented.

This method is particularly useful when the population is **heterogeneous** and contains **distinct categories**, such as gender, age groups, income levels, or education levels.

Example Scenario:

Imagine a university with **10,000 students** divided into **three faculties**:

Faculty	Population	Proportion (%)	Sample Size (out of 500)
Science	5,000	50%	250
Arts	3,000	30%	150
Business	2,000	20%	100

The **total sample size is 500 students**, and the number of students from each faculty is selected **proportionally** to its representation in the population.

Python Code: Stratified Sampling

```python
import pandas as pd
from sklearn.model_selection import train_test_split

# Dataset with the given names
data = {'Name': ['Syifa', 'Nabila', 'Alya', 'Isnaini', 'Rizky', 'Alfayed',
                 'Whirdyana', 'Olivia', 'Nabila A', 'Fika',
                 'Luthfi', 'Nabil', 'Joans', 'Riyadh', 'Rachelia', 'Nova',
                 'Zain', 'Ragil', 'Dadan', 'Dwi', 'Chello', 'Siti'],
                 'Faculty': ['Science', 'Arts', 'Science', 'Business',
                 'Science', 'Arts', 'Business', 'Arts', 'Science', 'Business',
```

```
                'Science', 'Arts', 'Business', 'Arts', 'Science', 'Business',
                'Science', 'Arts', 'Business', 'Arts', 'Science', 'Business']}

df = pd.DataFrame(data)

# Display initial group sizes
print("Original data distribution:")
```

```
Original data distribution:
```

```
print(df['Faculty'].value_counts())
```

```
Faculty
Science     8
Arts        7
Business    7
Name: count, dtype: int64
```

```
# Stratified Sampling (30% from each group)
stratified_sample, _ = train_test_split(df, test_size=0.7,
                                        stratify=df['Faculty'], random_state=42)

# Display sample group sizes
print("\nSampled data distribution (should be ~30% of each group):")
```

```
Sampled data distribution (should be ~30% of each group):
```

```
print(stratified_sample['Faculty'].value_counts())
```

```
Faculty
Science     2
Business    2
Arts        2
Name: count, dtype: int64
```

```
# Show sampled data
print("\nStratified Sample:")
```

```
Stratified Sample:
```

```
print(stratified_sample)
```

```
        Name   Faculty
16      Zain   Science
18     Dadan  Business
5    Alfayed      Arts
14  Rachelia   Science
17     Ragil      Arts
3    Isnaini  Business
```

R Code: Stratified Sampling

```r
library(dplyr)
library(purrr)

# Dataset with given names
data <- data.frame(
  Name = c("Syifa", "Nabila", "Alya", "Isnaini", "Rizky", "Alfayed",
           "Whirdyana", "Olivia", "Nabila A", "Fika",
           "Luthfi", "Nabil", "Joans", "Riyadh", "Rachelia",
           "Nova", "Zain", "Ragil", "Dadan", "Dwi", "Chello", "Siti"),
  Faculty = c("Science", "Arts", "Science", "Business",
              "Science", "Arts", "Business", "Arts", "Science", "Business",
              "Science", "Arts", "Business", "Arts", "Science",
              "Business", "Science", "Arts", "Business", "Arts",
              "Science", "Business")
)

# Show original data distribution
cat("Original data distribution:\n")
```

Original data distribution:

```r
print(table(data$Faculty))
```

```
    Arts Business  Science
       7        7        8
```

```r
# Determine sample size per strata (30% per group, rounded down)
sample_sizes <- data %>%
  count(Faculty) %>%
  mutate(sample_size = floor(n * 0.3))

# Perform stratified sampling with exact count per group
set.seed(42)

stratified_sample <- sample_sizes %>%
  split(.$Faculty) %>%
  map2(.x = ., .y = sample_sizes$sample_size, ~ data %>%
         filter(Faculty == .x$Faculty) %>%
         slice_sample(n = .y)) %>%
  bind_rows() %>%
  select(Name, Faculty)

# Show sampled data distribution
cat("\nSampled data distribution (should be exactly 30% of each group):\n")
```

Sampled data distribution (should be exactly 30% of each group):

```r
print(table(stratified_sample$Faculty))
```

```
    Arts Business  Science
       2        2        2
```

```r
# Display the sampled data
print(stratified_sample)
```

```
      Name  Faculty
1   Nabila     Arts
2   Riyadh     Arts
3  Isnaini Business
4     Siti Business
5     Alya  Science
6 Nabila A  Science
```

3.1.4 Cluster Sampling

Cluster Sampling is a probabilistic sampling technique where instead of selecting individuals randomly, we select entire groups (clusters). Once a cluster is selected, all individuals within that cluster are included in the sample.

This method is widely used for large populations where individual random selection is costly or impractical. It is especially useful in geographically spread-out populations or organizational structures like schools, hospitals, or companies.

Python Code: Cluster Sampling

```python
import pandas as pd
import numpy as np

# Sample dataset with 3 clusters (3 Schools)
data = pd.DataFrame({
    'Name': ['Syifa', 'Nabila', 'Alya', 'Isnaini', 'Rizky',
             'Alfayed', 'Whirdyana', 'Olivia', 'Nabila A', 'Fika',
              'Luthfi', 'Nabil', 'Joans', 'Riyadh', 'Rachelia',
              'Nova', 'Zain', 'Ragil', 'Dadan', 'Dwi', 'Chello', 'Siti'],
    'School': ['School A', 'School A', 'School A', 'School A',
               'School A', 'School B', 'School B', 'School B',
               'School B', 'School B','School B', 'School C',
               'School C', 'School C', 'School C', 'School C',
               'School C', 'School C', 'School C', 'School C',
               'School C', 'School C']
})

# Show original data distribution
print("Original Data Distribution:")
```

```
Original Data Distribution:
```

```
print(data['School'].value_counts())
```

```
School
School C    11
School B     6
School A     5
Name: count, dtype: int64
```

```
# Randomly select clusters (e.g., choose 1 out of 3 schools)
np.random.seed(42)
selected_clusters = np.random.choice(data['School'].unique(),
                                     size=1, replace=False)

# Select all individuals from the chosen clusters
cluster_sample = data[data['School'].isin(selected_clusters)]

# Display results
print("\nSelected Cluster(s):", selected_clusters)
```

```
Selected Cluster(s): ['School A']
```

```
print("\nCluster Sample:")
```

```
Cluster Sample:
```

```
print(cluster_sample)
```

```
      Name    School
0    Syifa  School A
1   Nabila  School A
2     Alya  School A
3  Isnaini  School A
4    Rizky  School A
```

R Code: Cluster Sampling

```
library(dplyr)

# Sample dataset with 3 clusters (3 Schools)
data <- data.frame(
  Name = c("Syifa", "Nabila", "Alya", "Isnaini", "Rizky", "Alfayed",
           "Whirdyana", "Olivia", "Nabila A", "Fika",
           "Luthfi", "Nabil", "Joans", "Riyadh", "Rachelia",
           "Nova", "Zain", "Ragil", "Dadan", "Dwi", "Chello", "Siti"),
  School = c("School A", "School A", "School A", "School A", "School A",
             "School B", "School B", "School B", "School B", "School B",
             "School B", "School C", "School C", "School C", "School C",
             "School C", "School C", "School C", "School C", "School C",
              "School C", "School C")
```

```
)

# Show original data distribution
cat("Original Data Distribution:\n")
```

```
Original Data Distribution:
```

```
print(table(data$School))
```

```
School A School B School C
       5        6       11
```

```
# Randomly select clusters (e.g., choose 1 out of 3 schools)
set.seed(42)
selected_clusters <- sample(unique(data$School), size = 1)

# Select all individuals from the chosen clusters
cluster_sample <- data %>% filter(School %in% selected_clusters)

# Display results
cat("\nSelected Cluster(s):", selected_clusters, "\n")
```

```
Selected Cluster(s): School A
```

```
cat("\nCluster Sample:\n")
```

```
Cluster Sample:
```

```
print(cluster_sample)
```

```
     Name   School
1   Syifa School A
2  Nabila School A
3    Alya School A
4 Isnaini School A
5   Rizky School A
```

3.2 Non-Probability Sampling

Non-probability sampling does not provide every individual with a known chance of selection, making it prone to bias but useful in exploratory research.

3.2.1 Convenience Sampling

Convenience Sampling is a non-probability sampling method where subjects are selected based on ease of access, availability, and proximity rather than randomness. It is commonly used in exploratory research, pilot studies, or situations where time and resources are limited.

Instead of carefully choosing a representative sample, researchers select participants who are easiest to reach—such as nearby students, colleagues, or online survey respondents.

Python Code: Convenience Sampling

```python
import pandas as pd

# Example dataset of students
data = pd.DataFrame({
    'Student_ID': range(1, 21),
    'Name': ['Student_' + str(i) for i in range(1, 21)],
    'Location': ['Campus'] * 10 + ['Online'] * 10}) # 10 from campus, 10 online

# Selecting the first 5 students available (e.g., from campus)
convenience_sample = data.head(5)

# Display selected sample
print(convenience_sample)
```

```
   Student_ID       Name Location
0           1  Student_1   Campus
1           2  Student_2   Campus
2           3  Student_3   Campus
3           4  Student_4   Campus
4           5  Student_5   Campus
```

R Code: Convenience Sampling

```r
# Create a sample dataset
data <- data.frame(
  Student_ID = 1:20,
  Name = paste("Student", 1:20, sep = "_"),
  Location = c(rep("Campus", 10), rep("Online", 10)) # 10 from campus, 10 online
)

# Selecting the first 5 students available (e.g., from campus)
convenience_sample <- data[1:5, ]

# Display selected sample
print(convenience_sample)
```

```
  Student_ID      Name Location
1          1 Student_1   Campus
2          2 Student_2   Campus
3          3 Student_3   Campus
4          4 Student_4   Campus
5          5 Student_5   Campus
```

3.2.2 Quota Sampling

Quota Sampling is a non-probability sampling method where researchers divide the population into subgroups (quotas) based on specific characteristics (e.g., age, gender, occupation) and select participants non-randomly to meet a predefined quota for each subgroup.

Unlike stratified random sampling, where individuals are randomly selected within each subgroup, quota sampling allows researchers to handpick individuals within quotas based on convenience or judgment, which introduces potential bias.

Python Code: Quota Sampling

```python
import pandas as pd

# Creating a dataset with 100 individuals (50 males, 50 females)
data = pd.DataFrame({
    'ID': range(1, 101),
    'Name': ['Person_' + str(i) for i in range(1, 101)],
    'Gender': ['Male'] * 50 + ['Female'] * 50,
})

# Defining quotas: 5 males and 5 females
quota_male = data[data['Gender'] == 'Male'].head(5)
quota_female = data[data['Gender'] == 'Female'].head(5)

# Combining quota-based sample
quota_sample = pd.concat([quota_male, quota_female])

# Displaying the selected sample
print(quota_sample)
```

```
    ID       Name  Gender
0    1   Person_1    Male
1    2   Person_2    Male
2    3   Person_3    Male
3    4   Person_4    Male
4    5   Person_5    Male
50  51  Person_51  Female
51  52  Person_52  Female
52  53  Person_53  Female
53  54  Person_54  Female
54  55  Person_55  Female
```

R Code: Quota Sampling

```r
# Creating a dataset
data <- data.frame(
  ID = 1:100,
  Name = paste("Person", 1:100, sep = "_"),
```

```r
  Gender = c(rep("Male", 50), rep("Female", 50))
)

# Defining quotas: 5 males and 5 females
quota_male <- head(subset(data, Gender == "Male"), 5)
quota_female <- head(subset(data, Gender == "Female"), 5)

# Combining quota-based sample
quota_sample <- rbind(quota_male, quota_female)

# Displaying the selected sample
print(quota_sample)
```

```
   ID      Name Gender
1   1  Person_1   Male
2   2  Person_2   Male
3   3  Person_3   Male
4   4  Person_4   Male
5   5  Person_5   Male
51 51 Person_51 Female
52 52 Person_52 Female
53 53 Person_53 Female
54 54 Person_54 Female
55 55 Person_55 Female
```

3.2.3 Judgmental Sampling

Researchers use their expertise to select the most relevant subjects. While it ensures focus, it introduces potential researcher bias.

Python Code: Judgmental Sampling

```python
import pandas as pd

# Creating a dataset
data = pd.DataFrame({
    'Name': ['Syifa', 'Nabila', 'Alya', 'Isnaini', 'Rizky',
             'Alfayed', 'Whirdyana', 'Olivia'],
    'Faculty': ['Science', 'Arts', 'Science', 'Business',
                'Science', 'Arts', 'Business', 'Arts'],
    'Experience (Years)': [5, 3, 10, 2, 7, 4, 8, 6]  # Experience in years
})

# Researcher selects only experienced Science faculty members (Judgement Sampling)
selected_sample = data[(data['Faculty'] == 'Science') &
                       (data['Experience (Years)'] > 5)]

# Displaying selected individuals
print(selected_sample)
```

```
    Name  Faculty  Experience (Years)
2   Alya  Science                  10
4  Rizky  Science                   7
```

R Code: Judgmental Sampling

```r
# Load necessary library
library(dplyr)

# Creating a dataset
data <- data.frame(
  Name = c("Syifa", "Nabila", "Alya", "Isnaini", "Rizky",
           "Alfayed", "Whirdyana", "Olivia"),
  Faculty = c("Science", "Arts", "Science",
              "Business", "Science", "Arts", "Business", "Arts"),
  Experience_Years = c(5, 3, 10, 2, 7, 4, 8, 6)  # Experience in years
)

# Researcher selects only experienced Science faculty members
selected_sample <- data %>%
  filter(Faculty == "Science", Experience_Years > 5)

# Displaying selected individuals
print(selected_sample)
```

```
   Name Faculty Experience_Years
1  Alya Science               10
2 Rizky Science                7
```

3.2.4 Snowball Sampling

Snowball Sampling is a non-probability sampling method used to study hard-to-reach or hidden populations (e.g., drug users, undocumented immigrants, people with rare diseases).

Instead of selecting participants randomly, researchers start with a small group of known individuals (seeds), who then recruit others from their social networks, creating a "snowball" effect.

Python Code: Snowball Sampling

```python
import pandas as pd
import random

# Creating a dataset of 100 individuals
data = pd.DataFrame({
    'ID': range(1, 101),
    'Name': ['Person_' + str(i) for i in range(1, 101)],
    'Group': ['Hidden Population'] * 100
})
```

```
# Start with 2 "seed" participants, 42 is just a common convention.
initial_sample = data.sample(n=2, random_state=42) # Any number can be used

# Snowball effect: Each seed recruits 2 more participants
snowball_sample = initial_sample.copy()
for _ in range(3):  # Repeat recruitment process
    new_recruits = data.sample(n=len(snowball_sample) * 2,
    random_state=random.randint(1, 100))
    snowball_sample = pd.concat([snowball_sample,
    new_recruits]).drop_duplicates()

# Displaying the selected sample
print(snowball_sample.head())
```

```
    ID       Name              Group
83  84  Person_84  Hidden Population
53  54  Person_54  Hidden Population
20  21  Person_21  Hidden Population
2    3   Person_3  Hidden Population
15  16  Person_16  Hidden Population
```

R Code: Snowball Sampling

```
# Load necessary library
library(dplyr)

# Creating a dataset
data <- data.frame(
  ID = 1:100,
  Name = paste("Person", 1:100, sep = "_"),
  Group = rep("Hidden Population", 100)
)

# Start with 2 "seed" participants
set.seed(42)
initial_sample <- sample_n(data, 2)

# Snowball effect: Each seed recruits 2 more participants
snowball_sample <- initial_sample
for (i in 1:3) {  # Repeat recruitment process
  new_recruits <- sample_n(data, nrow(snowball_sample) * 2)
  snowball_sample <- distinct(bind_rows(snowball_sample, new_recruits))
}

# Displaying the selected sample
print(head(snowball_sample))
```

```
  ID      Name              Group
1 49 Person_49 Hidden Population
```

```
2 65 Person_65 Hidden Population
3 25 Person_25 Hidden Population
4 74 Person_74 Hidden Population
5 18 Person_18 Hidden Population
6 47 Person_47 Hidden Population
```

3.3 Hybrid Sampling

In some research scenarios, Probability Sampling (random selection) and Non-Probability Sampling (subjective selection) can be combined to balance representation and practicality, it is called as a **Hybrid Sampling**.

3.3.1 Python Code: Hybrid Sampling

Here, we randomly select faculties (Probability Sampling), then select experienced members within each faculty (Judgement Sampling).

```python
import pandas as pd
import random

# Creating a dataset
data = pd.DataFrame({
    'Name': ['Syifa', 'Nabila', 'Alya', 'Isnaini', 'Rizky',
             'Alfayed', 'Whirdyana', 'Olivia', 'Nabil', 'Joans'],
    'Faculty': ['Science', 'Arts', 'Science', 'Business', 'Science',
                'Arts', 'Business', 'Arts', 'Science', 'Business'],
    'Experience (Years)': [5, 3, 10, 2, 7, 4, 8, 6, 12, 9]
})

# 1  Probability Sampling: Randomly select 2 faculties
random_faculties = random.sample(list(data['Faculty'].unique()), 2)

# 2  Non-Probability Sampling: Select experienced individuals from chosen faculties
hybrid_sample = data[(data['Faculty'].isin(random_faculties)) &
                     (data['Experience (Years)'] > 5)]

# Display results
print(hybrid_sample)
```

```
     Name  Faculty  Experience (Years)
2    Alya  Science                  10
4   Rizky  Science                   7
7  Olivia     Arts                   6
8   Nabil  Science                  12
```

3.3.2 R Code: Hybrid Sampling

We randomly select faculties (Probability Sampling) and then apply Judgment Sampling for experienced individuals.

```r
# Load necessary library
library(dplyr)

# Creating a dataset
data <- data.frame(
  Name = c("Syifa", "Nabila", "Alya", "Isnaini", "Rizky", "Alfayed",
           "Whirdyana", "Olivia", "Nabil", "Joans"),
  Faculty = c("Science", "Arts", "Science", "Business",
              "Science", "Arts", "Business", "Arts", "Science", "Business"),
  Experience_Years = c(5, 3, 10, 2, 7, 4, 8, 6, 12, 9)
)

# 1  Probability Sampling: Randomly select 2 faculties
set.seed(42)
random_faculties <- sample(unique(data$Faculty), 2)

# 2  Non-Probability Sampling: Select experienced individuals from chosen faculties
hybrid_sample <- data %>%
  filter(Faculty %in% random_faculties, Experience_Years > 5)

# Display results
print(hybrid_sample)
```

```
       Name  Faculty Experience_Years
1      Alya  Science               10
2     Rizky  Science                7
3 Whirdyana Business                8
4     Nabil  Science               12
5     Joans Business                9
```

3.4 Strengths & Limitations

The following table compares different sampling methods based on their strengths and limitations:

Sampling Method	Category	Strengths	Limitations
Simple Random Sampling	Probability	Unbiased, easy to implement	Requires full population list
Stratified Sampling	Probability	Ensures subgroup representation	Complex and time-consuming
Cluster Sampling	Probability	Cost-effective for large populations	Risk of sampling error
Systematic Sampling	Probability	More efficient than SRS	Can introduce bias if patterns exist
Convenience Sampling	Non-Probability	Quick and inexpensive	High risk of bias

Sampling Method	Category	Strengths	Limitations
Quota Sampling	Non-Probability	Ensures subgroup representation	Non-random selection introduces bias
Snowball Sampling	Non-Probability	Useful for niche populations	Limited generalizability
Judgmental Sampling	Non-Probability	Focused and expert-driven	Subject to researcher bias

3.5 Real-World Applications

- **Market Research:** Stratified sampling ensures different demographics are represented in surveys.
- **Medical Studies:** Random sampling helps in clinical trials to obtain unbiased results.
- **Social Sciences:** Snowball sampling is used for studying hidden populations like drug users or marginalized groups.
- **Business Analytics:** Convenience sampling is commonly used in quick customer feedback surveys.

Understanding and choosing the appropriate sampling method is crucial for obtaining valid and reliable research outcomes.

Chapter 4

Margin of Error

Margin of error (MoE) is a statistical concept that quantifies the uncertainty in survey results or sample-based estimates. It provides a range within which the true population parameter is likely to fall.

4.1 Why is MoE Important?

When conducting surveys or experiments, we rarely measure an entire population. Instead, we take a sample and use it to infer information about the whole group. However, because a sample is only a subset of the population, there is always some level of error. The margin of error helps account for this uncertainty.

4.2 Importance of Sample Size

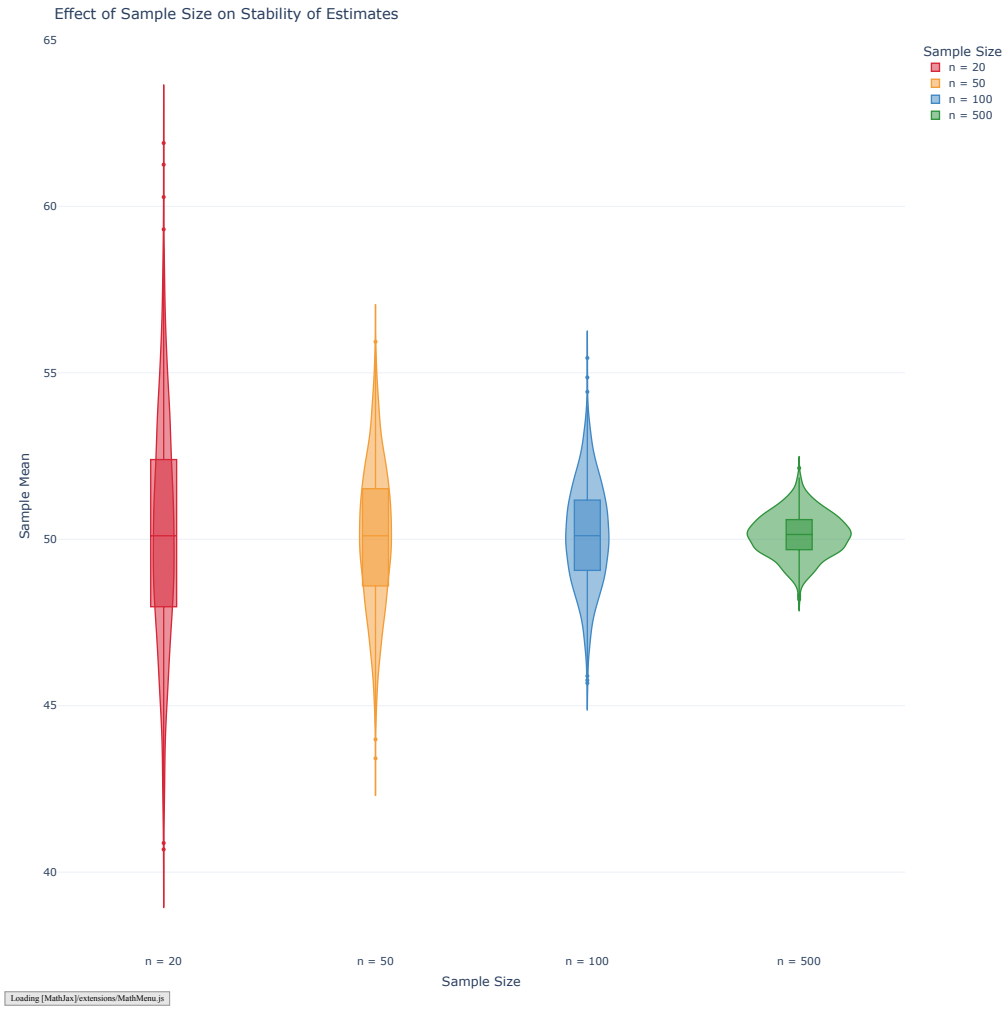
In statistics, **sample size** plays a crucial role in determining the **reliability** and **stability** of an estimate. A **larger sample size** leads to more **precise** and **consistent** estimates of the population parameters. Here's why:

- **Reduces Variability:** Smaller samples tend to exhibit greater fluctuations in their estimates, whereas larger samples provide more **stable** and **reliable** results.
- **Improves Accuracy:** A small sample may fail to accurately represent the overall population. However, as the sample size increases, the **sample mean converges toward the true population mean** (*Law of Large Numbers*).
- **Minimizes Sampling Error:** Larger samples result in **smaller sampling errors**, making conclusions **more generalizable** to the broader population.
- **Enhances Statistical Power:** A greater sample size **increases the statistical power** of tests, improving the ability to detect **true differences** or **effects**.

To gain a clearer understanding of this concept, let's visualize how different sample sizes impact the **distribution of sample means**:

4.2.1 Python code

```
import numpy as np
import pandas as pd
import plotly.express as px

# Set random seed for reproducibility
np.random.seed(123)

# Generate a normally distributed population
population = np.random.normal(loc=50, scale=15, size=10000)  # Mean=50, SD=15

# Function to take samples and compute mean
def sample_means(sample_size, n_samples=1000):
    means = [np.mean(np.random.choice(population,
            sample_size, replace=True)) for _ in range(n_samples)]
    return pd.DataFrame({'SampleSize': f"n = {sample_size}", 'Mean': means})

# Create sample distributions for different sizes
samples_20  = sample_means(20)
samples_50  = sample_means(50)
samples_100 = sample_means(100)
samples_500 = sample_means(500)

# Combine all into one dataset
sample_data = pd.concat([samples_20, samples_50, samples_100, samples_500])

# Define high-contrast colors for readability
custom_colors = {
    "n = 20":  "#D72638",  # Bright Red
    "n = 50":  "#F49D37",  # Deep Orange
    "n = 100": "#3F88C5",  # Strong Blue
    "n = 500": "#2E933C"   # Bold Green
}

# Create an interactive violin plot (without jitter)
fig = px.violin(sample_data, x="SampleSize", y="Mean", color="SampleSize",
                box=True, hover_data=["SampleSize"],
                color_discrete_map=custom_colors)

# Update layout for readability
fig.update_layout(
    title="Effect of Sample Size on Stability of Estimates",
    xaxis_title="Sample Size",
    yaxis_title="Sample Mean",
    template="plotly_white",
    font=dict(size=16),
    legend_title_text="Sample Size"
)
```



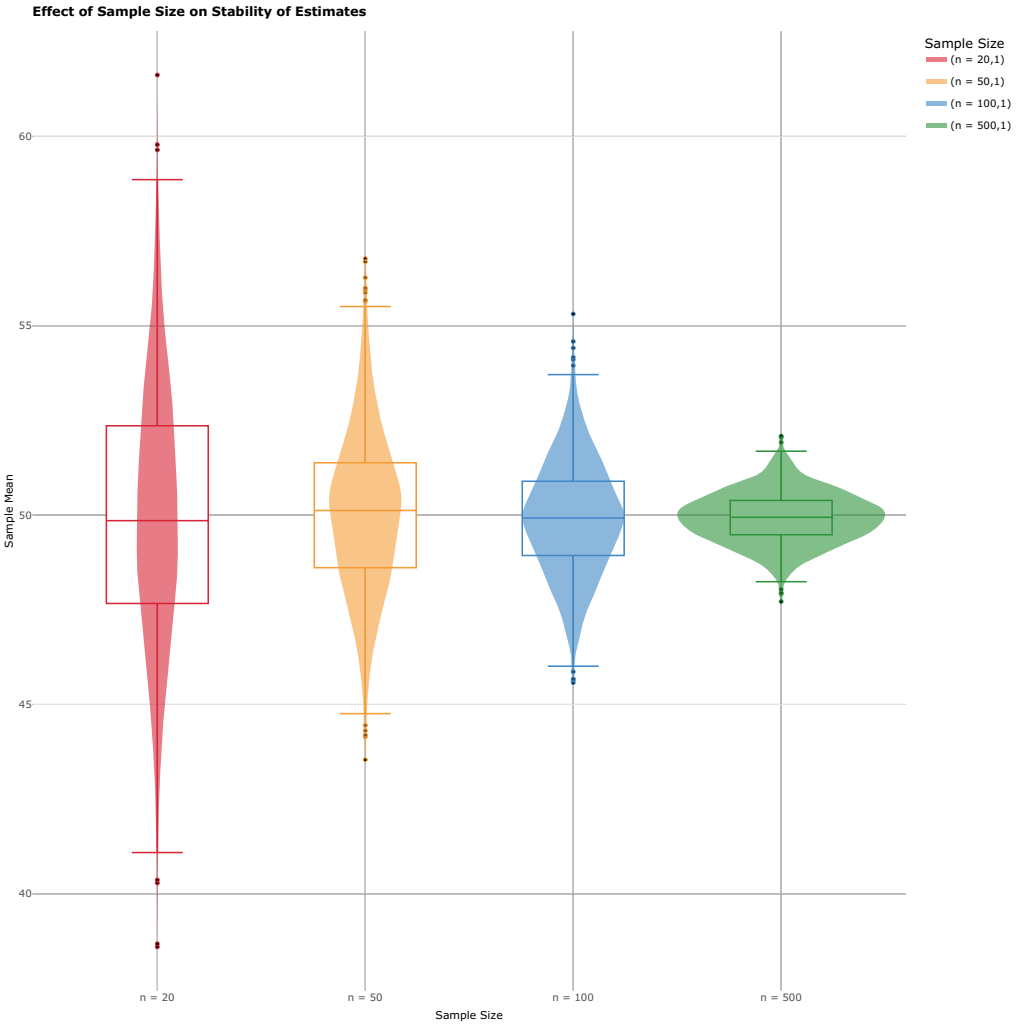

```
# Display the plot
# fig.show()
```

4.2.2 R code

```
# Load required libraries
library(ggplot2)
library(dplyr)
library(plotly)

set.seed(123)  # For reproducibility

# Generate a normally distributed population
population <- rnorm(10000, mean = 50, sd = 15)

# Function to take samples and compute mean
```

```r
sample_means <- function(sample_size, n_samples = 1000) {
  means <- replicate(n_samples, mean(sample(population,
                                            sample_size, replace = TRUE)))
  return(data.frame(SampleSize = paste0("n = ", sample_size), Mean = means))
}

# Create sample distributions for different sizes
samples_20  <- sample_means(20)
samples_50  <- sample_means(50)
samples_100 <- sample_means(100)
samples_500 <- sample_means(500)

# Combine all into one dataset
sample_data <- bind_rows(samples_20, samples_50, samples_100, samples_500)

# Convert SampleSize to a factor for better visualization
sample_data$SampleSize <- factor(sample_data$SampleSize,
                                 levels = c("n = 20",
                                            "n = 50",
                                            "n = 100",
                                            "n = 500"))

# Define high-contrast colors for readability
custom_colors <- c("n = 20"  = "#D72638",  # Bright Red
                   "n = 50"  = "#F49D37",  # Deep Orange
                   "n = 100" = "#3F88C5",  # Strong Blue
                   "n = 500" = "#2E933C")  # Bold Green

# Create violin plot with matching outline, avoiding duplicate legends
p <- ggplot(sample_data, aes(x = SampleSize,
                             y = Mean,
                             fill = SampleSize)) +
  geom_violin(alpha = 0.6, color = NA) +  # Remove outline from legend
  geom_boxplot(width = 0.1, fill = "white",
               outlier.shape = NA, aes(color = SampleSize), size = 0.6) +
  scale_fill_manual(values = custom_colors) +  # Custom fill colors
  scale_color_manual(values = custom_colors, guide = "none") +
  labs(
    title = "Effect of Sample Size on Stability of Estimates",
    x = "Sample Size", y = "Sample Mean",
    fill = "Sample Size"
  ) +
  theme_minimal(base_size = 14) +
  theme(
    plot.title = element_text(size = 14, face = "bold"),
    axis.title = element_text(size = 12),
    legend.position = "top"
  )
```

```
# Convert ggplot to interactive Plotly plot
ggplotly(p)
```



4.3 Factors Affecting Sample Size

Determining the appropriate sample size is crucial for obtaining accurate and meaningful results in statistical analysis. Several key factors influence the required sample size:

- Population Variability (Standard Deviation, SD)

The more diverse a population, the larger the sample needed to capture its full characteristics. If a population has high variability, a small sample may not be representative, leading to unreliable estimates.

- Confidence Level

The confidence level (e.g., 90%, 95%, or 99%) indicates how certain we want to be that the sample accurately represents the population. A higher confidence level requires a

larger sample size to reduce uncertainty.

3. Margin of Error (MoE)

The margin of error measures how much the sample estimate is expected to vary from the true population value. A smaller margin of error (e.g., $\pm 1\%$ instead of $\pm 5\%$) requires a larger sample to ensure higher precision.

4. Population Size

For small populations, a higher percentage needs to be sampled to achieve reliable results. However, beyond a certain point, increasing sample size provides diminishing returns in accuracy.

5. Study Complexity and Statistical Power

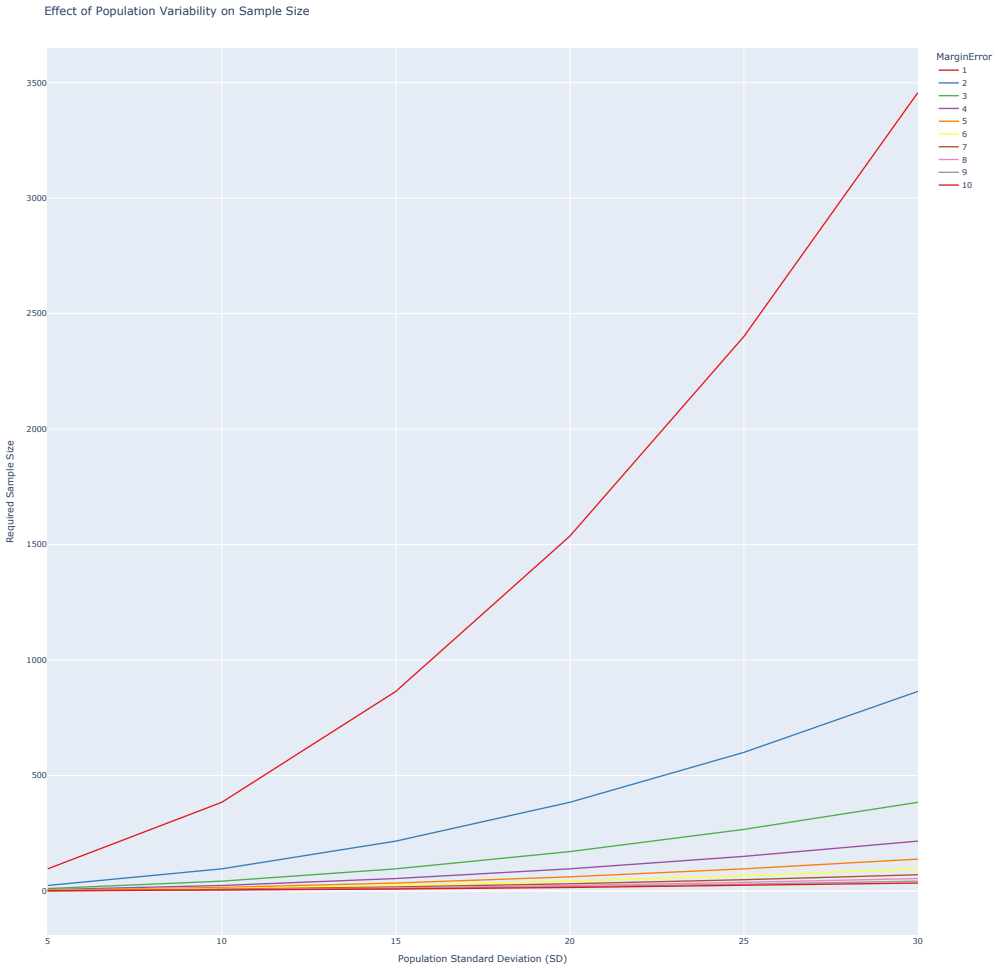
More complex studies (e.g., subgroup analysis, machine learning models) require larger sample sizes to ensure meaningful results. In hypothesis testing, a larger sample size increases statistical power, making it easier to detect true effects.

While a larger sample size improves accuracy, it also requires more time, cost, and resources. The goal is to find the optimal balance that ensures reliable results without unnecessary effort.

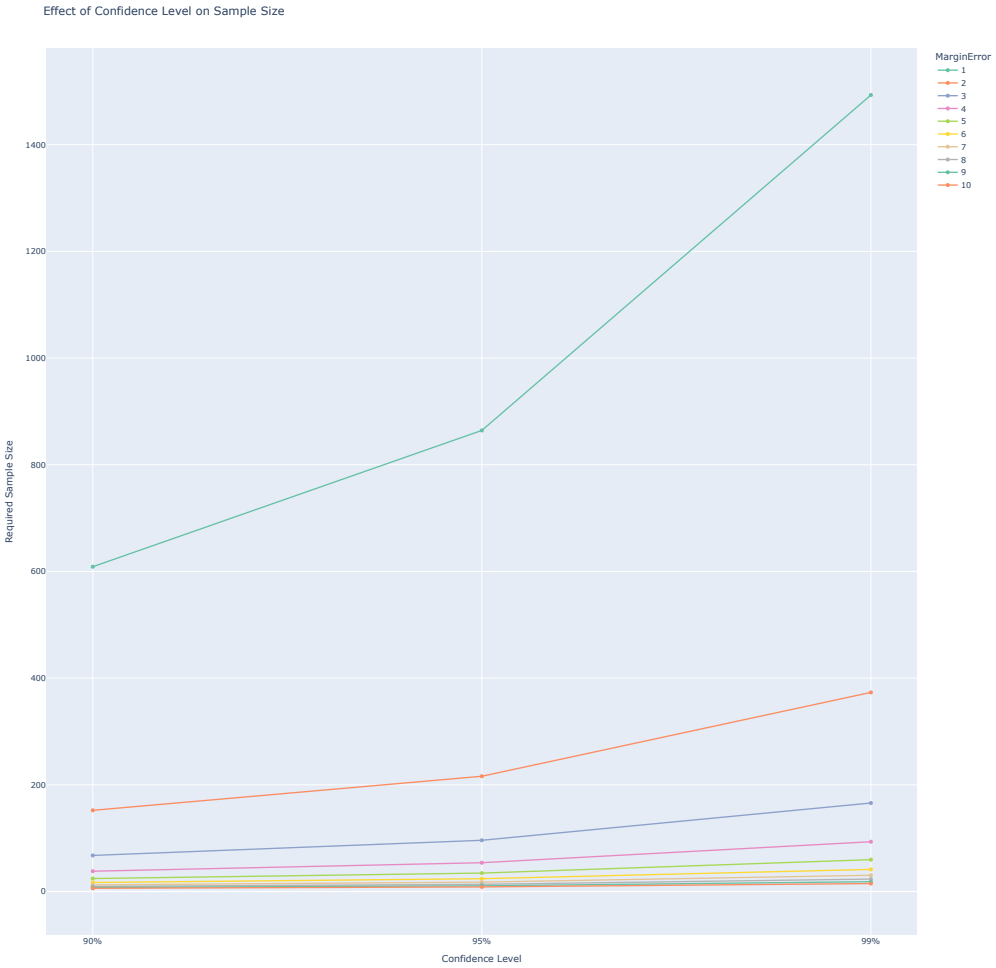
4.3.1 Python Code

```
import numpy as np
import pandas as pd
import plotly.express as px

# Sample Size Calculation Function
def calculate_sample_size(std_dev, confidence_z, margin_of_error):
    return ((confidence_z * std_dev) / margin_of_error) ** 2

# Define Parameters
sd_values = np.arange(5, 35, 5)  # Population standard deviations
confidence_levels = [1.645, 1.96, 2.576]  # 90%, 95%, 99% Z-scores
margin_errors = np.arange(1, 11, 1)  # Different margin of errors

# Generate Data for Population Variability Impact
var_data = pd.DataFrame([(sd, moe, calculate_sample_size(sd, 1.96, moe))
                         for sd in sd_values for moe in margin_errors],
                        columns=["SD", "MarginError", "SampleSize"])
var_data["MarginError"] = var_data["MarginError"].astype(str)

# Generate Data for Confidence Level Impact
conf_data = pd.DataFrame([(z, moe, calculate_sample_size(15, z, moe))
                          for z in confidence_levels for moe in margin_errors],
                         columns=["ConfidenceZ", "MarginError", "SampleSize"])
conf_data["ConfidenceZ"] = conf_data["ConfidenceZ"].map({1.645: "90%",
                           1.96: "95%", 2.576: "99%"})
conf_data["MarginError"] = conf_data["MarginError"].astype(str)
```

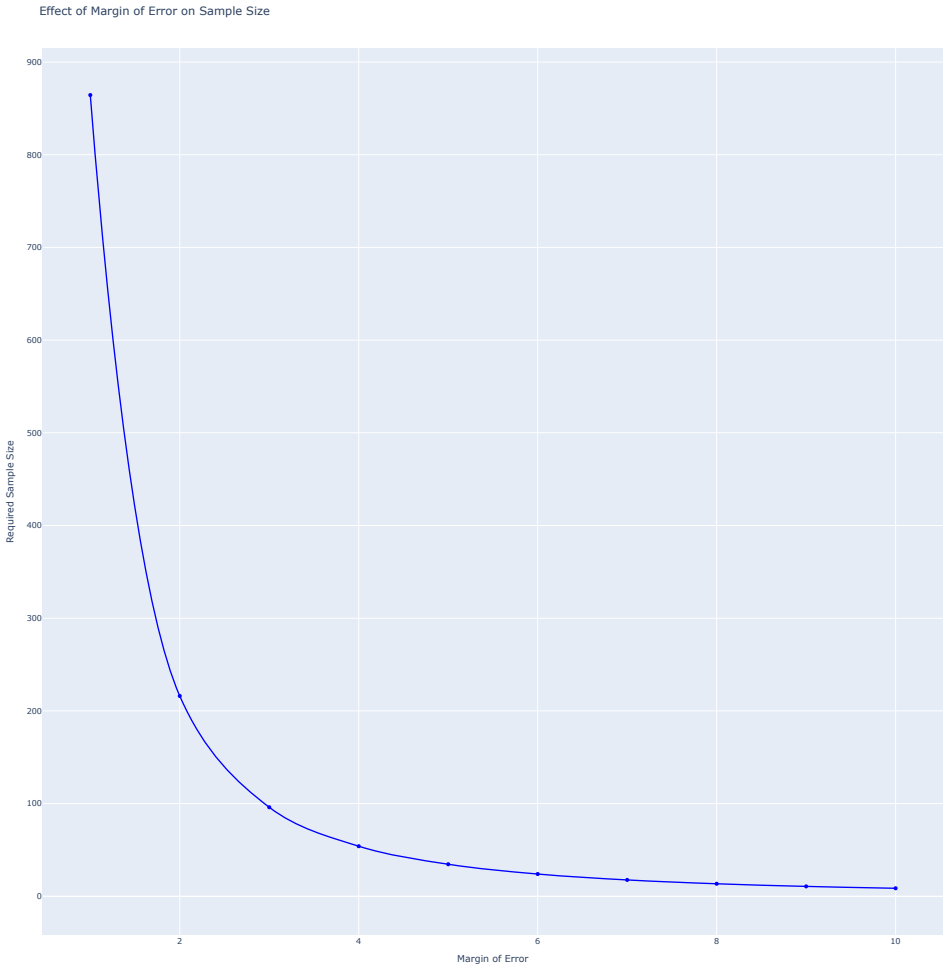
```
# Generate Data for Margin of Error Impact
me_data = pd.DataFrame({"MarginError": margin_errors,
                        "SampleSize": [calculate_sample_size(15, 1.96,
                        moe) for moe in margin_errors]})

# Plot 1: Effect of Population Variability
fig1 = px.line(var_data, x="SD", y="SampleSize", color="MarginError",
               title="Effect of Population Variability on Sample Size",
               labels={"SD": "Population Standard Deviation (SD)",
               "SampleSize": "Required Sample Size"},
               color_discrete_sequence=px.colors.qualitative.Set1)

# Plot 2: Effect of Confidence Level
fig2 = px.line(conf_data, x="ConfidenceZ", y="SampleSize", color="MarginError",
               title="Effect of Confidence Level on Sample Size",
               labels={"ConfidenceZ": "Confidence Level",
               "SampleSize": "Required Sample Size"},
               color_discrete_sequence=px.colors.qualitative.Set2,
               markers=True)

# Plot 3: Effect of Margin of Error
fig3 = px.line(me_data, x="MarginError", y="SampleSize",
               title="Effect of Margin of Error on Sample Size",
               labels={"MarginError": "Margin of Error",
               "SampleSize": "Required Sample Size"},
               markers=True, line_shape="spline",
               color_discrete_sequence=["blue"])  # Single color for clarity

# Show Plots
fig1.show()
```


Effect of Population Variability on Sample Size
MarginError
1
2
3
4
5
6
7
8
9
10
Required Sample Size
0
500
1000
1500
2000
2500
3000
3500
5
10
15
20
25
30
Population Standard Deviation (SD)

```
fig2.show()
```


Effect of Confidence Level on Sample Size
Required Sample Size
Confidence Level
90%
95%
99%
0
200
400
600
800
1000
1200
1400
MarginError
1
2
3
4
5
6
7
8
9
10

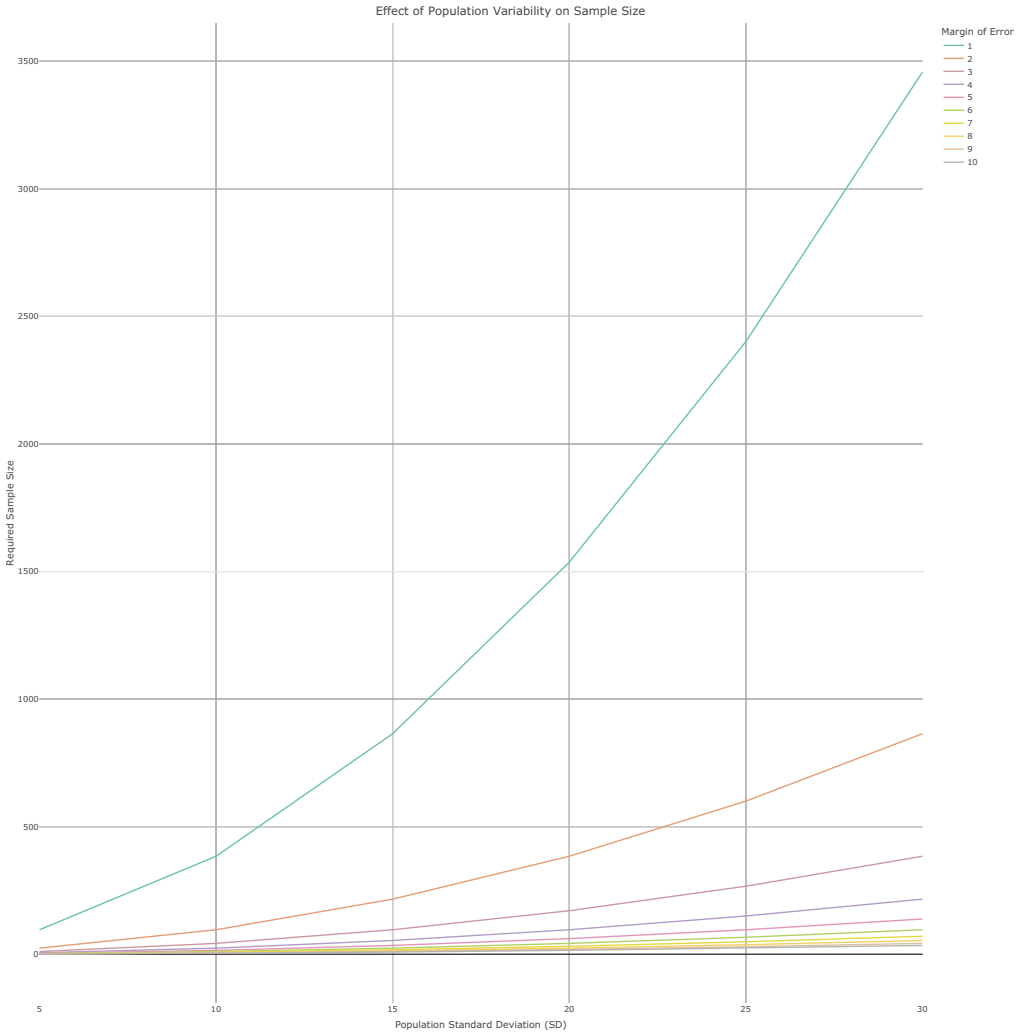
```
fig3.show()
```



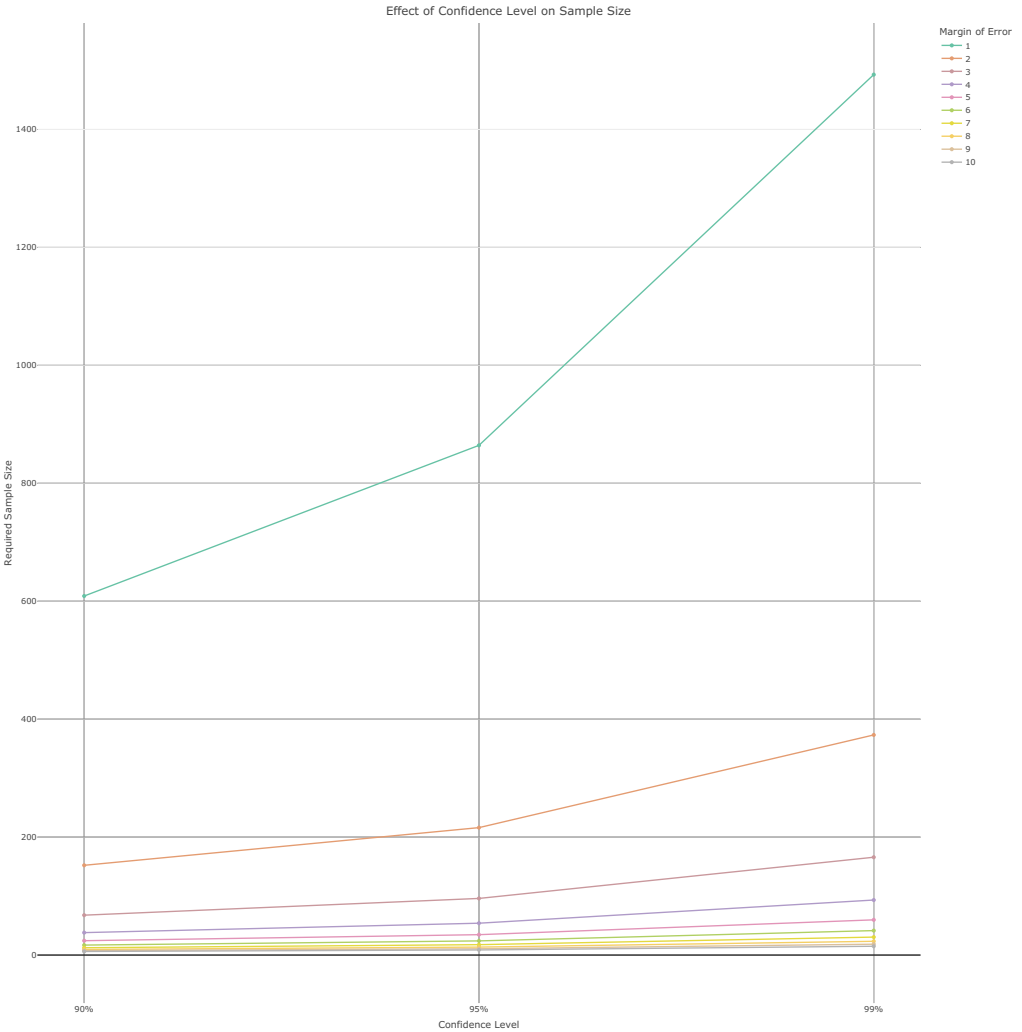
4.3.2 R Code

```r
# Load required libraries
library(plotly)
library(dplyr)

# Sample Size Calculation Function
calculate_sample_size <- function(sd, confidence_z, margin_error) {
  return(((confidence_z * sd) / margin_error)^2)
}

# Define Parameters
sd_values <- seq(5, 30, by = 5)  # Population standard deviations
confidence_levels <- c(1.645, 1.96, 2.576)  # 90%, 95%, 99% Z-scores
margin_errors <- seq(1, 10, by = 1)  # Different margin of errors
```

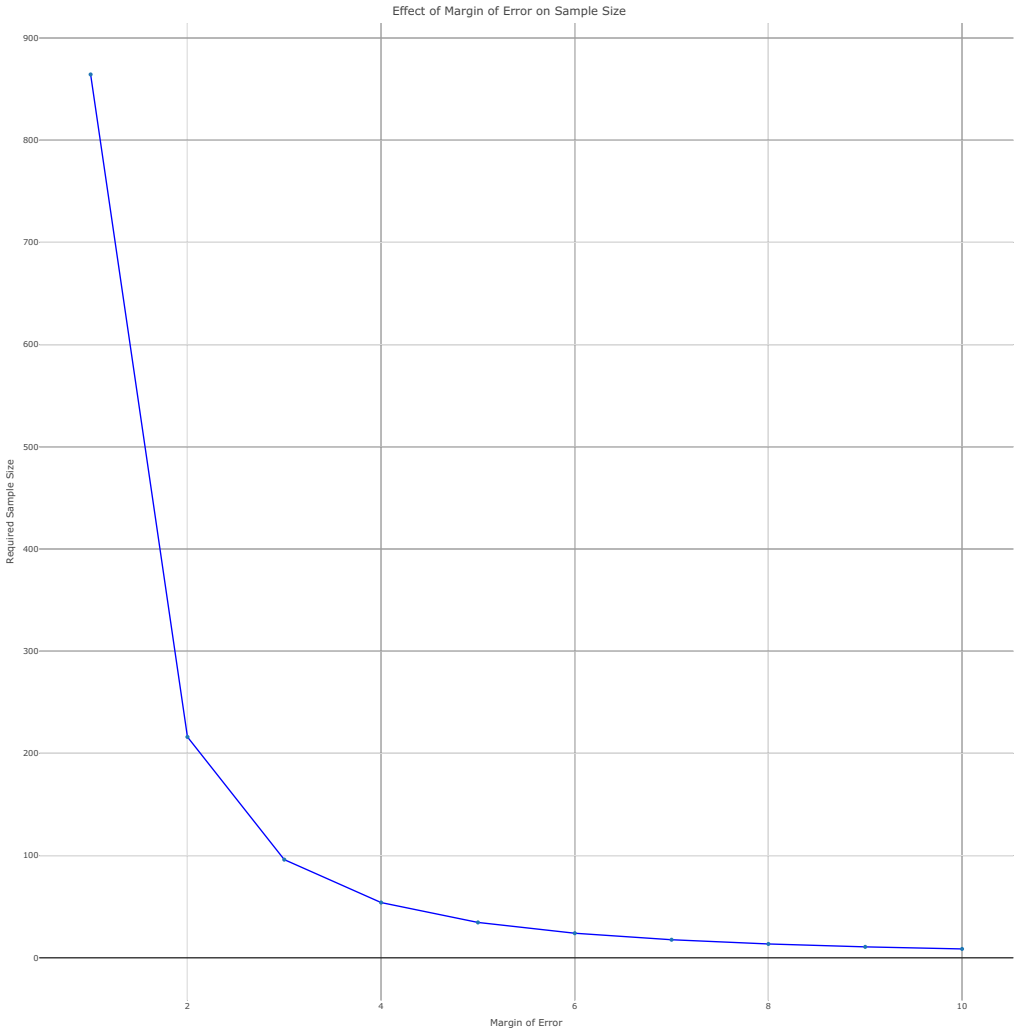
```r
# Generate Data for Variability Impact
var_data <- expand.grid(SD = sd_values, MarginError = margin_errors)
var_data$SampleSize <- mapply(calculate_sample_size, var_data$SD, 1.96, var_data$MarginError)

# Generate Data for Confidence Level Impact
conf_data <- expand.grid(ConfidenceZ = confidence_levels, MarginError = margin_errors)
conf_data$SampleSize <- mapply(calculate_sample_size, 15, conf_data$ConfidenceZ, conf_data$Ma
conf_data$ConfidenceZ <- factor(conf_data$ConfidenceZ, levels = confidence_levels, labels = c

# Generate Data for Margin of Error Impact
me_data <- data.frame(
  MarginError = margin_errors,
  SampleSize = calculate_sample_size(15, 1.96, margin_errors)
)

# Create Plotly Plots

# Plot 1: Effect of Population Variability
p1 <- plot_ly(var_data, x = ~SD, y = ~SampleSize, color = ~as.factor(MarginError),
              type = 'scatter', mode = 'lines', line = list(width = 2)) %>%
  layout(title = "Effect of Population Variability on Sample Size",
         xaxis = list(title = "Population Standard Deviation (SD)"),
         yaxis = list(title = "Required Sample Size"),
         legend = list(title = list(text = "Margin of Error")))

# Plot 2: Effect of Confidence Level
p2 <- plot_ly(conf_data, x = ~ConfidenceZ, y = ~SampleSize, color = ~as.factor(MarginError),
              type = 'scatter', mode = 'lines+markers', line = list(width = 2)) %>%
  layout(title = "Effect of Confidence Level on Sample Size",
         xaxis = list(title = "Confidence Level"),
         yaxis = list(title = "Required Sample Size"),
         legend = list(title = list(text = "Margin of Error")))

# Plot 3: Effect of Margin of Error
p3 <- plot_ly(me_data, x = ~MarginError, y = ~SampleSize,
              type = 'scatter', mode = 'lines+markers', line = list(color = 'blue', width = 2
  layout(title = "Effect of Margin of Error on Sample Size",
         xaxis = list(title = "Margin of Error"),
         yaxis = list(title = "Required Sample Size"))

# Display Interactive Plots
p1
```


Effect of Population Variability on Sample Size
Margin of Error
1
2
3
4
5
6
7
8
9
10
Required Sample Size
3500
3000
2500
2000
1500
1000
500
0
5
10
15
20
25
30
Population Standard Deviation (SD)

p2


Effect of Confidence Level on Sample Size
Margin of Error
1
2
3
4
5
6
7
8
9
10
Required Sample Size
1400
1200
1000
800
600
400
200
0
90%
95%
99%
Confidence Level

p3



4.4 Probability Sample Size

Probability sampling ensures that every member of the population has a known, non-zero chance of being selected. Here are examples of different **probability sampling methods** and how to calculate the required sample size.

4.4.1 Simple Random Sampling (SRS)

Every individual in the population has an **equal** chance of being selected.

Example: Estimating Average Test Scores A school wants to estimate the **average math test score** of students.

- Population size (**N**) = 2,000 students
- Standard deviation ( ) = 15

- Desired confidence level = **95%** (**Z = 1.96**)
- Margin of error (**E**) = 2 points

$$n = \left(\frac{Z \cdot \sigma}{E}\right)^2$$
$$n = \left(\frac{1.96 \times 15}{2}\right)^2$$

$$n = (14.7)^2 = 216.1$$
$$n \approx 217$$

Conclusion: The school should randomly select **217 students** for an accurate estimate.

4.4.2 Stratified Random Sampling

The population is divided into **subgroups (strata)** based on characteristics (e.g., gender, grade level), and a random sample is taken from each.

Example: Employee Satisfaction Survey A company wants to assess employee satisfaction across three departments.

- **Departments:** HR (100 employees), IT (200 employees), Sales (300 employees)
- **Total Population (N) = 600**
- Required Sample Size (**n**) = 200

To ensure proportional representation:

$$n_{HR} = \frac{100}{600} \times 200 = 33$$
$$n_{IT} = \frac{200}{600} \times 200 = 67$$
$$n_{Sales} = \frac{300}{600} \times 200 = 100$$

Conclusion: The company should survey **33 HR, 67 IT, and 100 Sales employees**.

4.4.3 Systematic Sampling

Every **k-th** individual is selected from an ordered list.

Example: Checking Product Quality

A factory produces **10,000** smartphones daily. The quality team inspects **500** of them.

To select samples systematically:

$$k = \frac{10,000}{500} = 20$$

A random starting point is chosen (e.g., **7**), then every **20th** phone is selected:
7, 27, 47, 67, 87...

Conclusion: A total of **500 phones** will be inspected systematically.

4.4.4 Cluster Sampling

The population is divided into **clusters**, and **entire clusters** are randomly selected.

Example: Measuring Household Electricity Consumption

A city has **10 districts**, each with **5,000 households**. Instead of surveying individuals across all districts, researchers randomly select **3 districts** and survey **all households within them**.

Conclusion: If each district has **5,000 households**, then **3 districts** $\times$ **5,000** $=$ **15,000 households** will be surveyed.

4.5 Non-Probability Sample Size

Non-probability sampling is used when random selection is not feasible, often due to time, cost, or accessibility constraints. In these methods, **sample size determination is more subjective**, as it depends on practical considerations rather than statistical formulas. Below are common **non-probability sampling methods** and how sample sizes are determined.

4.5.1 Convenience Sampling

Selection is based on **availability and willingness** of participants.

Example: Coffee Shop Customer Survey

A coffee shop wants to understand customer preferences. The manager surveys **the first 100 customers** who visit in the morning.

Sample Size Consideration:

- No fixed formula, depends on **time constraints** and **available respondents**.
- The manager stops at **100 responses**, assuming this is sufficient for insights.

4.5.2 Purposive (Judgmental) Sampling

Participants are **handpicked** based on specific **criteria**.

Example: Expert Opinion on Climate Change

A researcher wants insights from **climate scientists**. They select **50 experts** based on qualifications and experience.

Sample Size Consideration:

- Depends on the **research objective**.
- Common practice: **10-50 experts** for specialized studies.

4.5.3 Quota Sampling

The researcher **sets quotas** for different subgroups.

Example: Market Research for a New Product

A company surveys **500 people**, ensuring:

- **250 males, 250 females**
- **100 young adults (18-25), 200 middle-aged (26-45), 200 older adults (46+)**

Sample Size Consideration:

- Based on **market representation** and **business needs**.
- Typically uses **proportional allocation**.

4.5.4 Snowball Sampling

Used for **hard-to-reach populations**, where participants **refer others**.

Example: Studying Homeless Individuals

A researcher interviews **10 homeless individuals**, who refer others, growing the sample to **100 participants**.

Sample Size Consideration:

- Sample grows **organically** until data **saturation** (new responses add little value).
- Often **50-200 participants**.

4.6 Real-World Examples

In this study, you will compare **Probability Sampling** and **Non-Probability Sampling** in handling **Margin of Error (MoE)** when estimating **university students' monthly food expenses**. Please, apply **all methods** from both **Probability Sampling** and **Non-Probability Sampling** to gain a comprehensive understanding of their differences and effectiveness.

4.6.1 Selecting Sampling Methods

A. Probability Sampling

1. **Simple Random Sampling (SRS)**
   - Randomly select students from the entire population using a random number generator or a random number table.
2. **Stratified Sampling**
   - Divide the population into groups (strata), such as **faculty or academic year**.
   - Randomly select students from each stratum **proportionally**.
3. **Systematic Sampling**

- Select every **k-th student** from a sorted list of students.

- Example: If there are **10,000 students** and a sample of **200** is needed, select every **50th student** (10,000/200 = 50).

4. **Cluster Sampling**
   - Randomly select **some classes or student groups** and survey all students in those groups.

   - Example: Choose **5 random classes** and interview all students in those classes.
5. **Multi-Stage Sampling**
   - Combine multiple techniques, such as:
     - **Stage 1:** Randomly select **faculties**

     - **Stage 2:** Randomly select **classes within faculties**

     - **Stage 3:** Randomly select **students within those classes**

B. Non-Probability Sampling

1. **Convenience Sampling**
   - Interview students who are **easily accessible**, such as in the cafeteria or library.
2. **Quota Sampling**
   - Ensure a fixed number of students are surveyed in each category (e.g., **50 students per faculty**) without random selection.
3. **Judgmental (Purposive) Sampling**
   - Select students who are **believed to be representative**, such as dormitory residents who may have more consistent food expenses.
4. **Snowball Sampling**
   - Start with a few students and ask them to recommend other students for the survey.

4.6.2 Data Collection

1. **Apply each sampling method to select student samples.**

2. **Record their monthly food expenses.**

3. **Store data in a spreadsheet or statistical software for analysis.**

4.6.3 Calculate Margin of Error

For **Probability Sampling**, calculate the **Margin of Error (MoE)** using the formula:

$$MoE = Z \times \frac{\sigma}{\sqrt{n}}$$

Where:

- Z = **1.96** (for 95% confidence level)
- σ = **Sample standard deviation (calculated from data)**
- n = **Sample size**

Perform **MoE calculations** for each Probability Sampling method and compare the results.

4.6.4 Bias Analysis

- Explain the **sources of bias** in each **Non-Probability Sampling** method.
- Discuss **how this bias affects survey results** and **the difference compared to Probability Sampling**.

4.6.5 Determine Sample Size

Use the following formula to determine required Sample Size for MoE = 5:

$$n = \left(\frac{Z \times \sigma}{MoE}\right)^2$$

Where:
- Z = **1.96**
- σ = **Sample standard deviation**
- $MoE = 5$

Calculate the **minimum required sample size** and interpret the results.

4.6.6 Create a Study Report

The final report should include:

1. **Introduction** – Purpose of the study and importance of MoE in sampling
2. **Sampling Methods Used** – Explanation of all methods applied
3. **MoE Calculations for Probability Sampling**
4. **Bias Analysis in Non-Probability Sampling**
5. **Comparison of All Methods**
6. **Required Sample Size for MoE = 5**
7. **Conclusion and Recommendations**

4.6.7 Additional Instructions

- Use **Excel, R, or Python** to calculate MoE and generate comparison charts.

- If using Python, use **numpy, pandas, and scipy.stats** for analysis.
- If using R, use **qnorm()**, **sqrt()**, and **mean()** functions.
- Ensure all calculations and results are clearly interpreted.
- Submit your answer on RPubs or Google Colab.

Chapter 5

Questionnaire Design

Designing an effective questionnaire is crucial for gathering accurate and meaningful data. A well-structured questionnaire ensures clarity, minimizes bias, and enhances response rates. This chapter covers the key aspects of questionnaire design, including different types of survey questions and best practices for structuring them. Before proceeding to the next level of this topic, please consider watching the following video.

In this session, we'll break down the fundamentals of questionnaire design—covering different question types, structuring techniques, and strategies to improve response rates. Whether you're conducting research, gathering customer feedback, or analyzing trends, a well-designed questionnaire can make all the difference.

5.1 Types of Survey Questions

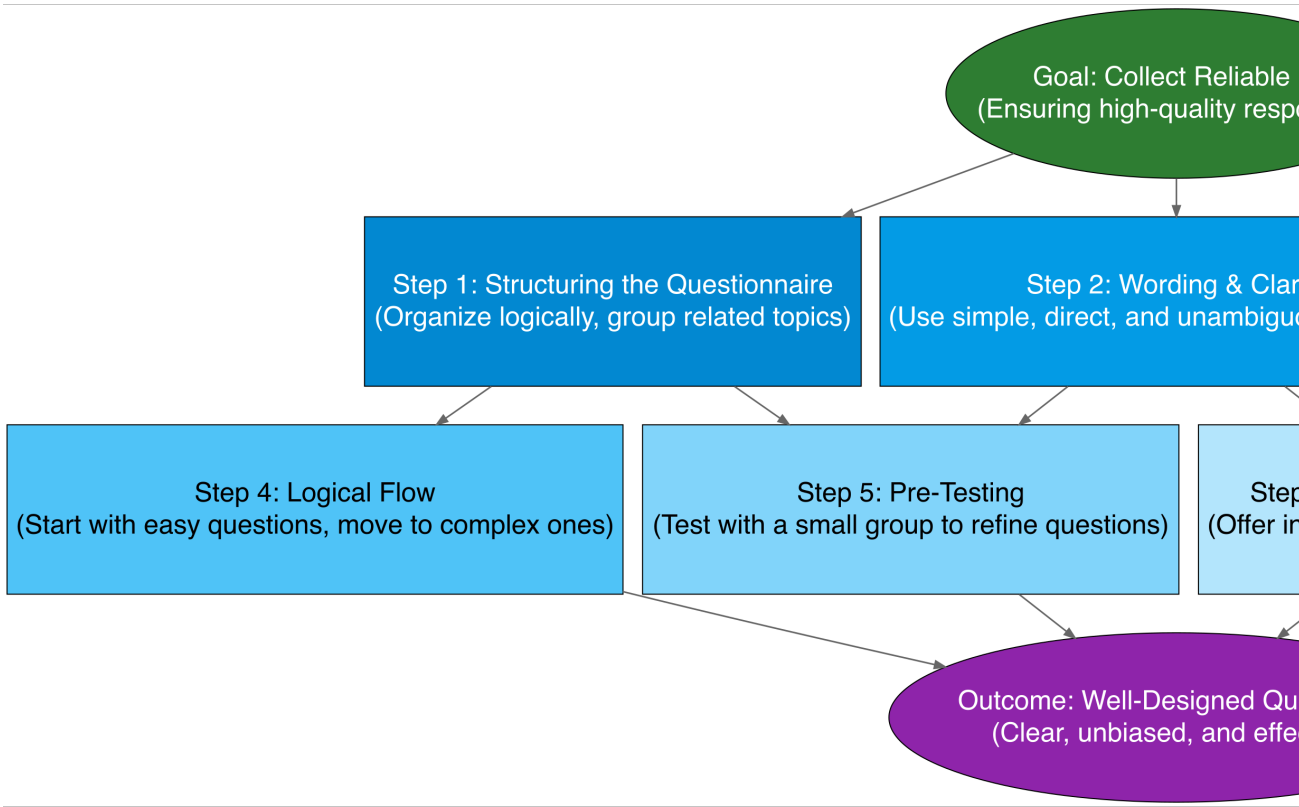
When designing a questionnaire, choosing the right type of questions is essential. Different survey question types serve distinct purposes and influence the quality and reliability of responses. Below are some of the most common types of survey questions, along with examples:

Question Type	Description	Example
Open-Ended Questions	Allow respondents to provide detailed and free-form responses, useful for exploring opinions, experiences, and suggestions.	"What improvements would you like to see in our customer service?"
Closed-Ended Questions	Provide specific response options, making it easier to analyze data quantitatively.	"Which of the following best describes your level of satisfaction with our service?"- Very Satisfied- Satisfied- Neutral- Dissatisfied- Very Dissatisfied

Question Type	Description	Example
Likert Scale Questions	Measure attitudes and perceptions using a scale, typically ranging from agreement to disagreement.	"On a scale of 1 to 5, how strongly do you agree with the following statement: 'The online shopping experience was easy and convenient'?"- 1: Strongly Disagree- 2: Disagree- 3: Neutral- 4: Agree- 5: Strongly Agree
Rating Questions	Allow respondents to evaluate a statement, service, or product on a numerical or graphical scale.	"How would you rate your overall experience with our mobile app on a scale of 1 to 10?"
Multiple-Choice Questions	Provide respondents with a predefined list of answers, allowing single or multiple selections.	"Which of the following features do you use most often? (Select all that apply)"- Live Chat Support- FAQ Section- Email Support- Phone Support
Dichotomous Questions	Offer only two possible answers, such as "Yes" or "No."	"Have you purchased from our store in the past six months?" (Yes/No)

5.2 Structuring a Questionnaire

A well-structured questionnaire follows a logical sequence that flows smoothly for respondents. Begin with easy and engaging questions before moving to more complex or sensitive ones. Group similar topics together for clarity.


Goal: Collect Reliable
(Ensuring high-quality resp
Step 1: Structuring the Questionnaire
(Organize logically, group related topics)
Step 2: Wording & Clar
(Use simple, direct, and unambigu
Step 4: Logical Flow
(Start with easy questions, move to complex ones)
Step 5: Pre-Testing
(Test with a small group to refine questions)
Step
(Offer in
Outcome: Well-Designed Qu
(Clear, unbiased, and effe

5.2.1 Structuring the Questionnaire

A well-structured questionnaire should have a clear sequence to guide respondents smoothly through the survey.

- Organize questions logically to create a natural flow.
- Group related topics together for clarity and coherence.
- Begin with easy and engaging questions to encourage participation before moving to more complex ones.
- Use a mix of question types (multiple-choice, open-ended, Likert scale) to collect diverse data.
- Ensure consistency in question formatting and terminology.

5.2.2 Wording & Clarity

To ensure clarity and avoid confusion:

- Use simple, clear, and concise language that respondents can easily understand.
- Avoid technical jargon unless your audience is highly familiar with it.
- Frame questions in a way that eliminates ambiguity and misinterpretation.
- Use direct wording to improve response accuracy.
- Avoid double-barreled questions that ask about multiple things at once.

Example of a Poor Question: - "How do you perceive the intrinsic value and overall utility of our digital solutions?"

Improved Version: - "How useful do you find our digital solutions?"

5.2.3 Avoiding Bias

To prevent unintentional influence on responses:

- Avoid leading questions that push respondents toward a particular answer.
- Use neutral wording that does not assume a certain opinion.
- Provide a balanced set of response options to prevent bias in answers.
- Use randomized answer choices (where applicable) to reduce order bias.
- Avoid emotionally loaded language that could sway opinions.

Example of a Biased Question: - "Don't you think our product is excellent?"

Unbiased Version: - "How would you rate our product?"

5.2.4 Ensuring Logical Flow

A well-structured sequence of questions improves response quality:

- Start with general and non-threatening questions to engage respondents.
- Move to more specific or sensitive questions gradually.
- Use conditional logic (skip logic) to direct respondents to relevant questions based on their previous answers.
- Place demographic or classification questions at the end to prevent early disengagement.
- Test the question order to ensure it makes sense and is intuitive.

5.2.5 Pre-Testing & Pilot Surveys

Before finalizing the questionnaire:

- Conduct a pilot survey with a small, diverse group representing your target audience.
- Identify ambiguities, unclear wording, and potential biases based on pilot responses.
- Collect feedback on the overall questionnaire experience.
- Adjust question wording, order, or format based on pilot survey findings.
- Use statistical analysis to check for inconsistencies or confusion in responses.

5.2.6 Encouraging Responses

To maximize participation and reduce dropouts:

- Keep the questionnaire short, relevant, and engaging.
- Clearly communicate the purpose of the survey and how the data will be used.
- Offer incentives (e.g., discounts, prize draws, or exclusive content) to encourage participation.
- Send reminders via email or SMS to those who have not completed the survey.
- Ensure confidentiality and anonymity to make respondents feel comfortable sharing honest opinions.
- Optimize the questionnaire for different devices (desktop, mobile, tablet) to enhance accessibility.

5.2.7 Identifying & Fixing Mistakes

Common mistakes that reduce the effectiveness of a questionnaire:

- **Too many questions** – Keep it concise to maintain engagement.
- **Ambiguous wording** – Ensure clarity and precision in every question.
- **Inconsistent scales** – Use uniform response options to avoid confusion.
- **Lack of logical flow** – Arrange questions in a way that makes sense to respondents.
- **Overuse of open-ended questions** – Balance open-ended and close-ended questions for efficiency.
- **Ignoring cultural differences** – Adapt questions to fit the background of diverse respondents.

By applying these best practices, you can design a questionnaire that is clear, unbiased, engaging, and effective in gathering high-quality and reliable data. Ensuring proper structure, clarity, and flow will improve response rates and data accuracy, leading to more insightful and actionable results.

5.3 Study Case Questionnaire

Background:

ABC Online Store, a fast-growing e-commerce platform, has been experiencing a decline in customer retention. Recent data shows that while many users visit the platform and make initial purchases, repeat purchases have decreased significantly. The management

suspects that customer satisfaction issues might be a contributing factor, but they lack specific insights into customer experiences, preferences, and pain points.

To address this issue, the company decides to conduct a structured customer satisfaction survey using a well-designed questionnaire.

Research Objective: The primary goal of this study is to:

- Identify key factors affecting customer satisfaction.
- Understand pain points in the shopping experience.
- Determine how product quality, pricing, and customer support impact retention.
- Gather data to enhance future services and improve customer retention.

5.3.1 Structuring the Questionnaire

- **Logical Flow:** The survey starts with general questions about the shopping experience, then moves to specific issues like website usability, product satisfaction, and customer service.
- **Grouping Topics:** The questionnaire is divided into sections:
  1. Shopping Experience
  2. Product Quality
  3. Pricing and Promotions
  4. Customer Service
  5. Future Expectations
- **Question Types:** The survey uses multiple-choice, Likert scale, and open-ended questions.

5.3.2 Wording & Clarity

- **Example of Poor Question:**
  "Do you find the pricing of our products to be strategically aligned with the industry standards while maintaining high product quality?"
- **Improved Version:**
  "Are our product prices reasonable compared to their quality?"
- **Example of Poor Question:**
  "Did you face any inconveniences due to poor customer service response times, incorrect information, or lack of issue resolution?"
- **Improved Version:**
  "How satisfied are you with the response time of our customer service?"

5.3.3 Avoiding Bias

- **Example of a Biased Question:**
  "Don't you think our product is excellent?"
- **Unbiased Version:**
  "How would you rate our product?"

- **Example of Leading Question:**
  "Would you say our customer service is excellent?"
- **Unbiased Version:**
  "How would you rate your experience with our customer service?"

5.3.4 Ensuring Logical Flow

Question Order Strategy:

- Start with an engaging question: *"How often do you shop online?"*
- Move to usability: *"How easy is it to find what you need on our website?"*
- Then product satisfaction: *"How satisfied are you with the quality of the products?"*
- End with demographics and classification questions.

5.3.5 Pre-Testing & Pilot Survey

- Before launching, ABC Online Store conducts a pilot test with 30 existing customers.
- **Feedback includes:**
  - Some questions felt repetitive.
  - The survey was slightly too long.
  - A few technical terms needed simplification.
- Adjustments were made before rolling it out to the broader audience.

5.3.6 Encouraging Responses

To increase participation, ABC Online Store: - Offers a **10% discount** for completing the survey. - Sends **email reminders** after 3 days. - **Assures anonymity** to encourage honest feedback. - Makes the survey **mobile-friendly** for easy access.

5.3.7 Identifying & Fixing Mistakes

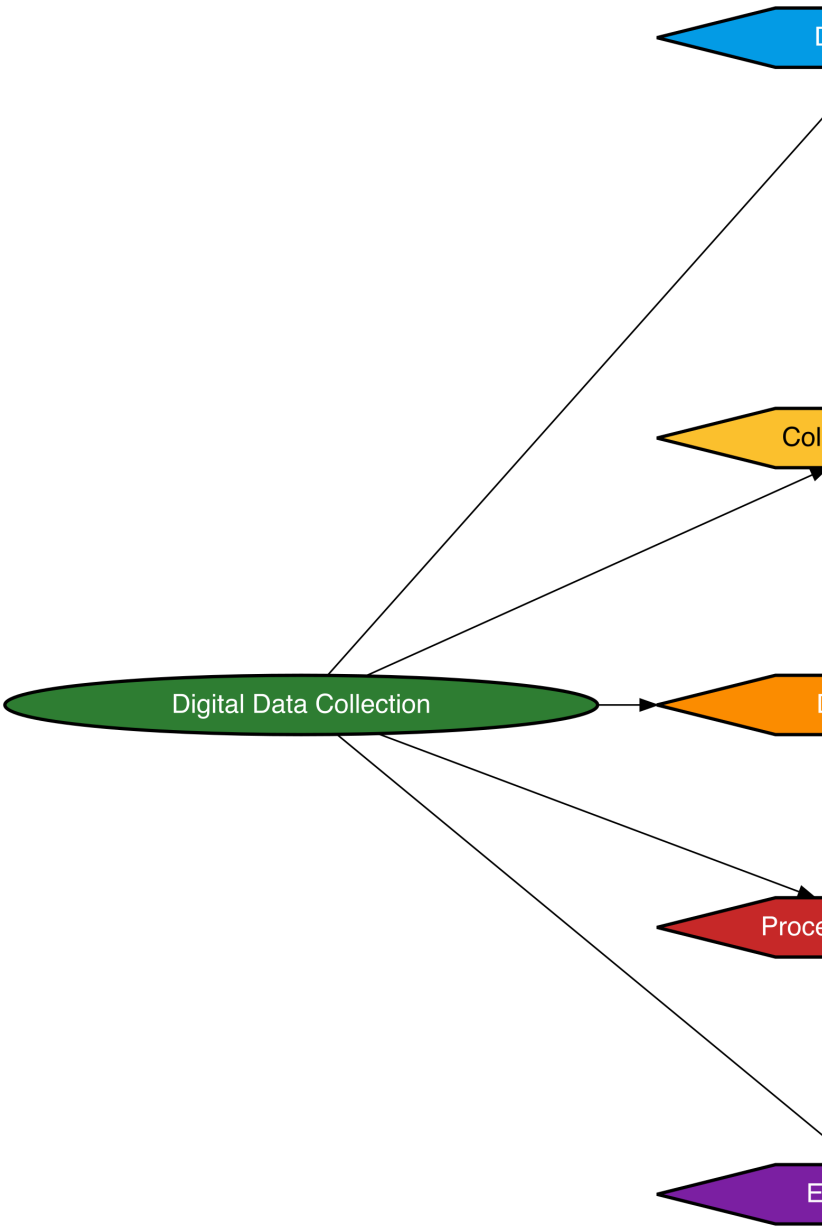
After collecting initial responses, the team notices: - Some respondents skipped open-ended questions, so **wording was adjusted** to encourage short responses. - The customer service satisfaction scale was inconsistent with other rating scales, so it was **standardized**. - A few questions lacked a "Not Applicable" option, which was then added.

By following a structured questionnaire design process, ABC Online Store successfully gathered valuable data that directly improved customer satisfaction and retention.

Chapter 6

Digital Data Collection

Digital data collection refers to the process of gathering, recording, and storing data using digital tools and technologies. Unlike traditional paper-based methods, digital data collection leverages devices such as smartphones, tablets, sensors, and computers to improve efficiency, accuracy, and accessibility of data.


Digital Data Collection
Col
Proce

6.1 Advantages of Digital Surveys

Digital surveys offer numerous benefits over traditional paper-based surveys, making them a popular choice for data collection in research, business, and decision-making. Here are the key advantages:

6.1.1 Cost-Effective

- Eliminates the need for printing, paper, and manual data entry.
- Reduces administrative costs related to survey distribution and processing.

6.1.2 Faster Data Collection

- Responses can be gathered in real-time.
- Automated distribution via email, social media, or websites speeds up participation.

6.1.3 Improved Accuracy and Data Quality

- Reduces human errors in data entry.
- Built-in validation rules help ensure complete and accurate responses.

6.1.4 Easy Customization and Personalization

- Allows logic-based questions (e.g., skip logic) to tailor surveys to respondents.
- Can include multimedia elements like images and videos for better engagement.

6.1.5 Convenience for Respondents

- Participants can complete surveys anytime and anywhere using mobile devices or computers.
- No need for in-person interactions, making participation more accessible.

6.1.6 Automated Data Analysis

- Digital surveys integrate with analytical tools for instant data processing.
- Generates real-time reports, charts, and insights without manual effort.

6.1.7 Higher Response Rates

- Mobile-friendly formats encourage more participation.
- Automatic reminders can be sent to non-respondents.

6.1.8 Environmental Benefits

Reduces paper waste, contributing to sustainability efforts.

6.1.9 Global Reach

- Enables surveys to be distributed across different locations and demographics effortlessly.

- Supports multiple languages for diverse audiences.

6.1.10 Enhanced Security and Data Storage

- Digital platforms offer encryption and secure cloud storage to protect survey data.
- Minimizes the risk of data loss compared to paper-based records.

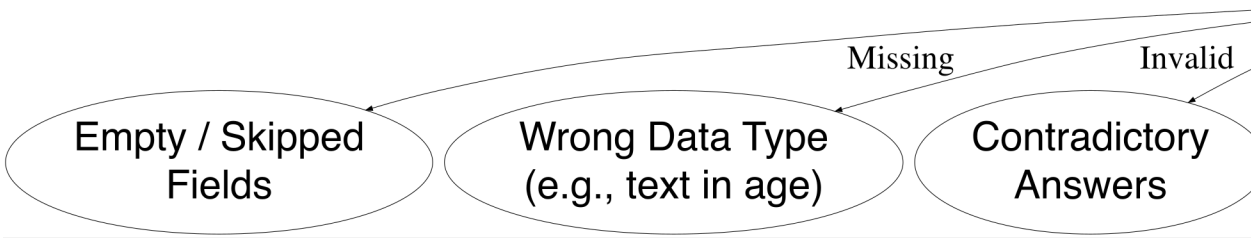
Digital surveys provide a faster, more efficient, and cost-effective way to collect and analyze data. Their flexibility, accuracy, and convenience make them a powerful tool for researchers, businesses, and organizations.

6.2 Online Survey Platforms

Chapter 7

Data Validation

Before analyzing survey data, it's essential to validate that the data collected is accurate, complete, and follows the expected format. This step helps catch simple errors early, such as text in a numeric field, missing entries, or contradictory answers. For example, if someone types "twenty-five" instead of "25" in the "Age" field, the system should detect and correct it.



7.1 Importance of Data Quality

High-quality data forms the foundation of reliable analysis and sound decision-making. If the data contains errors, it can lead to incorrect insights and poor strategic choices. For instance, inaccurate customer satisfaction scores might suggest there's a problem where none exists—or fail to reveal a real issue. Clean, accurate data improves credibility and saves time during analysis.

7.2 Common Survey Data Errors

Survey data is prone to several types of errors that can distort findings if not addressed. Identifying these issues early is critical for maintaining data integrity.

- **Missing responses**: Questions left blank by respondents.
- **Invalid entries**: Incorrect data types, such as entering "abc" for age.
- **Inconsistent answers**: Contradictions within a response, like stating "Not employed" but listing a workplace.
- **Duplicate submissions**: The same respondent completes the survey multiple times.
- **Outliers**: Unusual values that don't align with the overall pattern, like a monthly income of "10,000,000" when most report between "10,000" and "100,000".

7.3 Techniques for Data Cleaning

Cleaning data means correcting or removing errors so that the dataset is ready for analysis. This step is crucial to ensure consistency and reliability.

- **Standardizing formats**: For example, using the same date format such as "YYYY-MM-DD".
- **Fixing typos and inconsistencies**: Like correcting misspellings or varying labels for missing values ("N/A", "None", "NA").
- **Using tools**: Software like Excel, Google Sheets, or programming libraries (e.g., Python's `pandas`) help clean data efficiently.

7.4 Automated vs Manual Validation

Validation can be performed automatically or manually, depending on the type and complexity of data.

- **Automated validation**: Built-in rules or scripts to prevent invalid input, such as allowing only numbers in an age field.
- **Manual validation**: Human review is needed for open-ended or complex responses, such as interpreting the relevance of a text answer.

7.5 Handling Missing Data

Missing data is a common issue that must be handled thoughtfully to avoid biased results. The approach depends on the nature of the data and the research goals.

- **Listwise deletion**: Remove responses with missing data entirely.
- **Imputation**: Estimate the missing value, for example, by using the average value from similar respondents.
- **Flagging**: Mark missing values to exclude them from certain types of analysis.

7.6 Detecting Outliers & Inconsistencies

Outliers and logical inconsistencies can distort results and should be identified early.

- **Outliers**: Extremely high or low values, like someone reporting an age of 150.
- **Inconsistencies**: Conflicting answers, such as saying "No children" but listing "Childcare" as an expense.

These issues can be flagged for review or corrected if they appear to be errors.

7.7 Duplicate Response Detection

Duplicate responses can skew results and must be filtered out. This typically involves looking for patterns or clues that suggest repeated entries.

- **Repeated IP addresses**
- **Identical answers across all questions**
- **Submissions within a short timeframe**

For example, if two identical surveys come from the same IP address five minutes apart, one may be a duplicate and should be excluded.

7.8 Validation Features

Digital survey platforms come with built-in tools to help reduce data entry errors during the collection process. These tools are especially useful for real-time validation.

- **Required fields**: Prevent submission if a key question is skipped.
- **Predefined answer formats**: Like allowing only valid email addresses or numbers.
- **Dropdown menus or multiple-choice fields**: Limit the chance of invalid input.
- **Skip logic**: Hide or show questions based on previous responses to ensure relevance.

7.9 Ensuring Data Accuracy

Maintaining high data accuracy starts from the survey design phase and continues throughout data processing. Below are best practices that can help ensure reliable results:

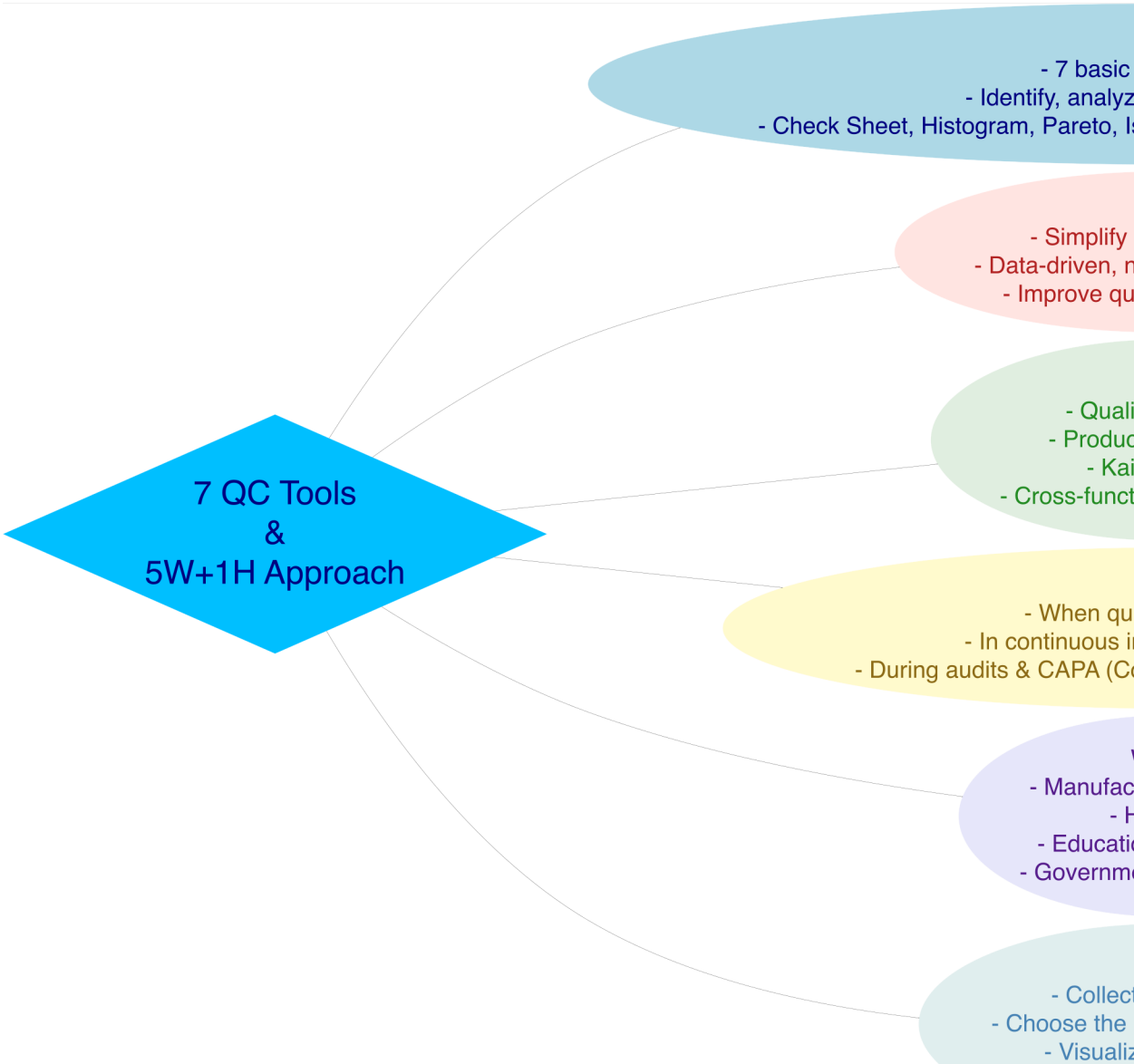
1. **Design clear and simple questions** to minimize misinterpretation.
2. **Pilot test the survey** with a small group to identify confusing items.
3. **Train data collectors** (if any) to avoid input errors.
4. **Use real-time validation tools** such as mandatory fields or input restrictions.
5. **Document all validation and cleaning steps** to support transparency and reproducibility.

Chapter 8

Seven Tools Analysis

8.1 Introduction to 7 QC Tools

Definition: The 7 QC Tools (Seven Quality Control Tools) are simple statistical methods used to systematically identify, analyze, and solve quality-related problems.



8.2 What

What are the 7 QC Tools?

The **7 QC Tools** are seven simple, statistics-based tools used in quality management to systematically identify, analyze, and resolve problems. Here are the seven tools:

1. **Check Sheet** – A structured data collection sheet

2. **Histogram** – A frequency distribution graph
3. **Pareto Chart** – A bar chart based on the 80/20 principle
4. **Cause and Effect Diagram (Fishbone/Ishikawa)** – A diagram to trace root causes
5. **Scatter Diagram** – A plot showing relationships between two variables
6. **Control Chart** – A process monitoring control chart
7. **Flowchart** – A visual representation of a process flow

8.3 Why

Why are the 7 QC Tools important?

- Easy to use by anyone, even those without a statistical background
- Problem-solving is based on data, not assumptions
- Improves process efficiency and effectiveness
- Supports fact-based decision making
- Enhances product/service quality and customer satisfaction

8.4 Who

Who uses the 7 QC Tools?

- Quality managers and quality control staff
- Production operators and supervisors
- Continuous Improvement / Kaizen teams
- Internal and external auditors
- Professionals from various fields: education, healthcare, public services, etc.

8.5 Where

Where are the 7 QC Tools used?

- Manufacturing industries and factories
- Hospitals and healthcare services

- Service and financial companies
- Educational institutions
- Government organizations and public services

8.6 When

When are the 7 QC Tools used?

- When recurring quality issues arise
- During continuous improvement initiatives
- When conducting root cause analysis
- During internal or external quality audits
- When developing corrective and preventive actions (CAPA)

8.7 How

How to use the 7 QC Tools?

- Collect data from relevant processes or activities
- Choose the appropriate QC tool for your analysis goal
- Visualize the data using one of the 7 tools
- Analyze data patterns and trends
- Identify the root cause of the issue
- Develop solutions and improvement plans
- Take follow-up action and monitor improvement results

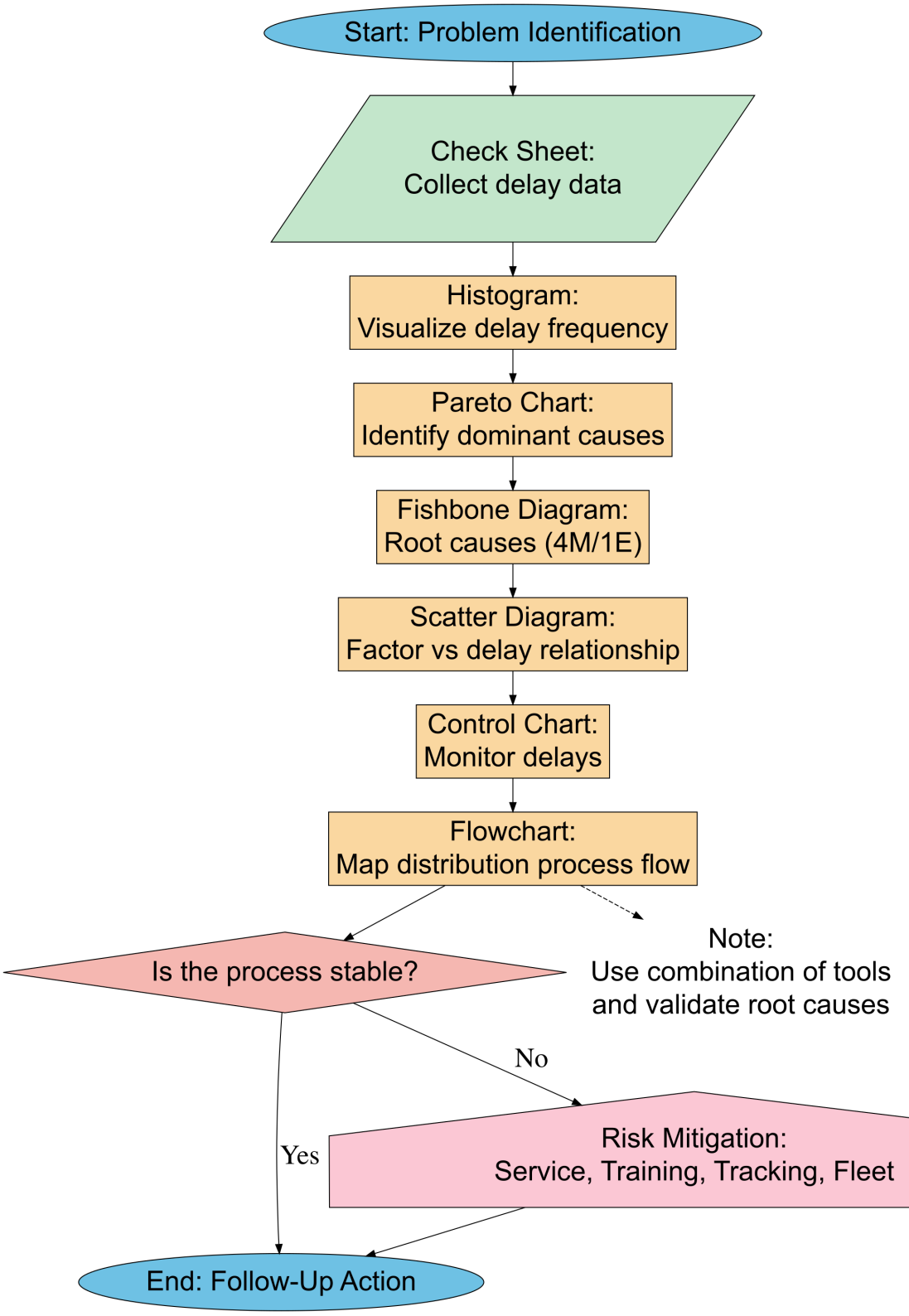
8.8 Applied of 7 QC Tools

The application of the 7 QC Tools is highly effective in helping organizations identify and mitigate potential operational risks.

For example, in the context of **delivery delays**, the process can begin with a **Check Sheet** to collect data on dates, times, and reasons for the delays. The gathered data is then analyzed using a **Histogram** to observe frequency distribution patterns, and a **Pareto Chart** to identify the dominant causes that should be prioritized.

Next, a **Fishbone Diagram** (Cause and Effect) can be used to explore root causes based on factors like Man, Machine, Method, Material, and Environment. Once the root

causes are identified, a **Scatter Diagram** helps to examine the relationship between key variables—such as the link between order quantity and delay frequency. A **Control Chart** is then used to monitor process stability over time, ensuring variations remain within acceptable limits. Finally, a **Flowchart** is used to map the entire logistics process, helping to pinpoint potential risk areas.


Start: Problem Identification
Check Sheet:
Collect delay data
Histogram:
Visualize delay frequency
Pareto Chart:
Identify dominant causes
Fishbone Diagram:
Root causes (4M/1E)
Scatter Diagram:
Factor vs delay relationship
Control Chart:
Monitor delays
Flowchart:
Map distribution process flow
Is the process stable?
Note:
Use combination of tools
and validate root causes
No
Yes
Risk Mitigation:
Service, Training, Tracking, Fleet
End: Follow-Up Action

By using the 7 QC Tools, organizations can make data-driven decisions to formulate risk mitigation strategies, such as staff training, regular fleet maintenance, or improving delivery tracking systems. The systematic implementation of the 7 QC Tools not only enhances service quality but also reduces the potential losses due to unmanaged risks.

In operational and logistics management, risks such as delivery delays, data discrepancies, and process errors often arise and affect service quality. To address this, a data-driven approach becomes crucial, and the 7 QC Tools provide the right solution because they can identify, analyze, and help reduce the root causes of these issues systematically.

8.8.1 Cheeck Sheet

Check Sheet is the first step in identifying a problem. This tool helps gather field data quickly and accurately, especially when we do not yet know the specific problem. In risk mitigation, Check Sheet serves as a tool to **observe and document risk-causing events** systematically.

Aspect	Description
Brief Definition	A simple QC tool used to collect data systematically and in real-time, typically in the form of a table to record the frequency of specific events.
Main Function	- Record data quickly and accurately- Identify patterns or event frequencies- Facilitate preliminary analysis of issues
Case Example	A logistics team creates a check sheet to record the date, time, and reasons for delivery delays over the course of one month. This data becomes the basis for further analysis.

8.8.1.1 Check Sheet in R

1. Data Check Sheet

Copy CSV PDF Print Search:

Check Sheet: Delivery Delay Records

Date	Reason
All	All
2025-04-15	Sudden Request
2025-04-19	Document Error
2025-04-14	Road Closure
2025-04-03	Driver Sick
2025-04-10	Road Closure
2025-04-18	Driver Sick
2025-04-22	Operational Issue
2025-04-11	Technical Disruption
2025-04-05	Late Departure
2025-04-20	Late Departure
2025-04-14	Sudden Request
2025-04-22	Road Closure
2025-04-25	Technical Disruption
2025-04-26	Fleet Issues
2025-04-27	Technical Disruption
2025-04-05	Fleet Issues
2025-04-19	Technical Disruption

Showing 1 to 100 of 100 entries

2. Frequency Tabulation

```
# Count frequency of each delay reason
library(dplyr)
```

```r
library(DT)

reason_summary <- check_sheet %>%
  count(Reason, sort = TRUE) %>%
  rename(Frequency = n)

# Display summary table
datatable(
  reason_summary,
  options = list(
    scrollCollapse = TRUE,
    searching = FALSE,    # Remove search box
    paging = FALSE        # Remove pagination
  ),
  rownames = FALSE,
  caption = htmltools::tags$caption(
    style = 'caption-side: top; text-align: left;
             font-size: 18px; font-weight: bold;',
    'Check Sheet: Summary of Delay Reasons'
  ),
  class = 'stripe hover compact'
)
```

Check Sheet: Summary of Delay Reasons

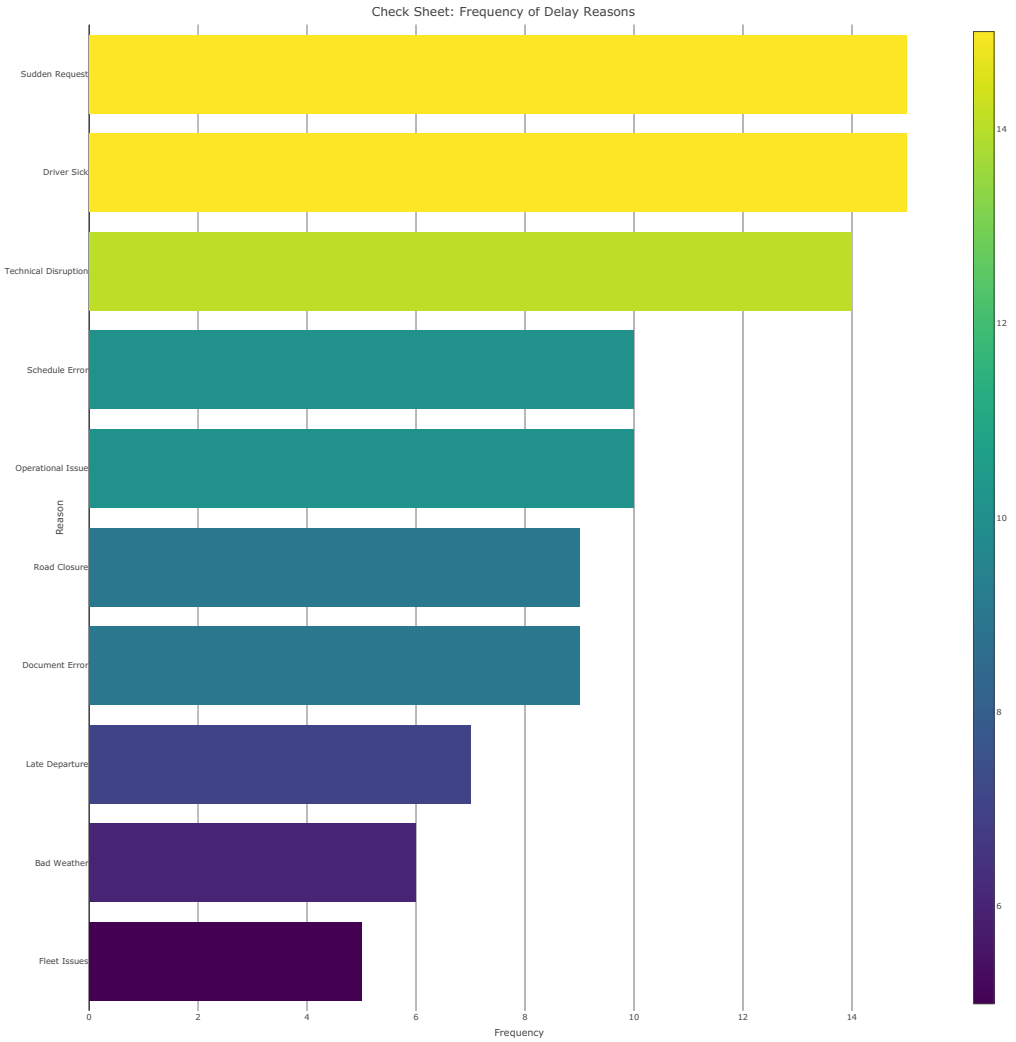
Reason	Frequency
Driver Sick	15
Sudden Request	15
Technical Disruption	14
Operational Issue	10
Schedule Error	10
Document Error	9
Road Closure	9
Late Departure	7
Bad Weather	6
Fleet Issues	5

Showing 1 to 10 of 10 entries

3. Simple Visualization

```r
library(plotly)

# Interactive bar chart using plotly
plot_ly(reason_summary,
        x = ~Frequency,
        y = ~reorder(Reason, Frequency),
        type = 'bar',
        orientation = 'h',
        marker = list(
          color = ~Frequency,
          colorscale = 'Viridis',  # Can also try: 'Bluered', 'Cividis', 'YlOrRd'
          showscale = TRUE
        )
) %>%
  layout(
    title = list(text = "Check Sheet: Frequency of Delay Reasons", font = list(size = 18)),
    xaxis = list(title = "Frequency"),
```

```
    yaxis = list(title = "Reason"),
    margin = list(l = 120)
  )
```



8.8.1.2 Check Sheet in Python

1. Data Check Sheet

```
import pandas as pd
import numpy as np
import dash
from dash import dash_table, html, dcc
import plotly.express as px
from datetime import datetime, timedelta

# Simulate data
np.random.seed(123)
```

```
dates = pd.date_range(start="2025-04-01", end="2025-04-30")
delay_reasons = [
    "Bad Weather",
    "Fleet Issues",
    "Road Closure",
    "Schedule Error",
    "Late Departure",
    "Operational Issue",
    "Driver Sick",
    "Sudden Request",
    "Document Error",
    "Technical Disruption"
]

data = {
    "Date": np.random.choice(dates, 100),
    "Reason": np.random.choice(delay_reasons, 100)
}

check_sheet = pd.DataFrame(data)
```

2. Frequency Tabulation

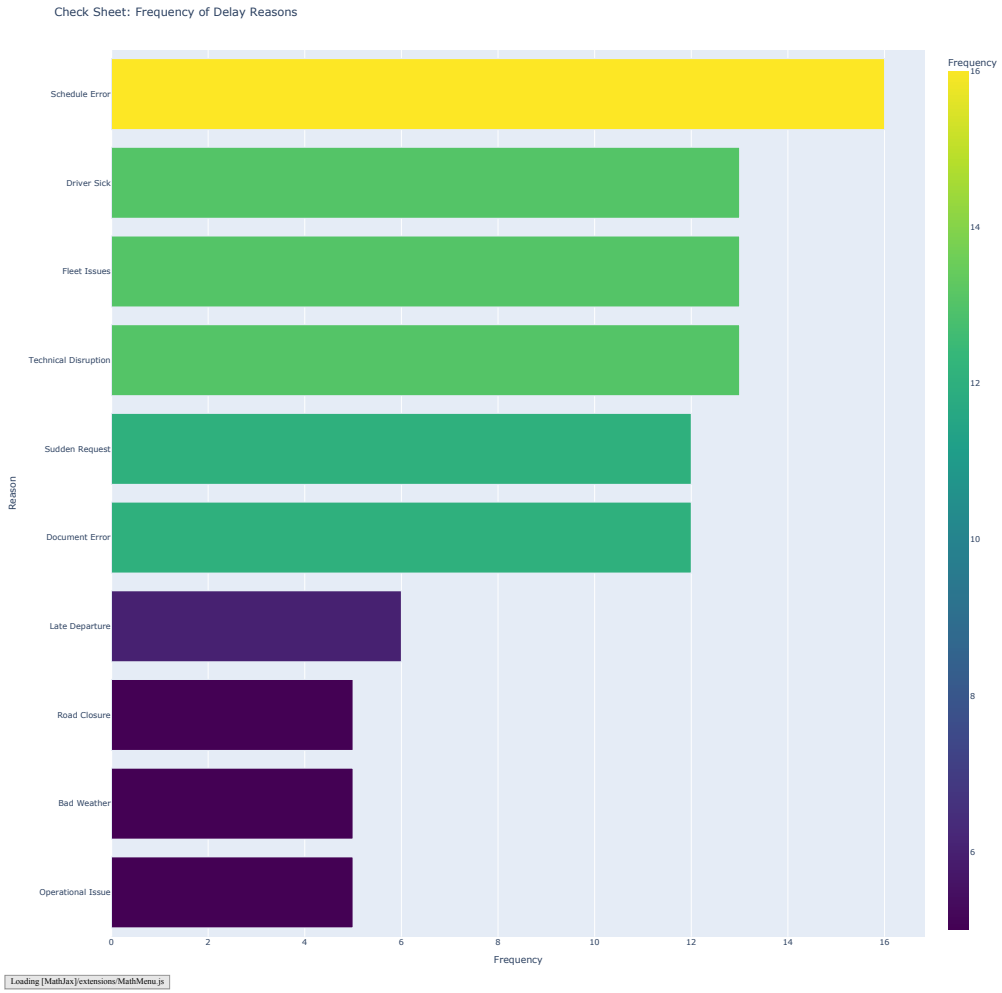
```
# Count frequency of delay reasons
reason_summary = check_sheet['Reason'].value_counts().reset_index()
reason_summary.columns = ['Reason', 'Frequency']
reason_summary
```

```
                Reason  Frequency
0        Schedule Error         16
1  Technical Disruption         13
2          Fleet Issues         13
3           Driver Sick         13
4        Document Error         12
5        Sudden Request         12
6        Late Departure          6
7     Operational Issue          5
8           Bad Weather          5
9          Road Closure          5
```

3. Simple Visualization

```
# Create a horizontal bar plot
fig = px.bar(
    reason_summary.sort_values("Frequency"),
    x="Frequency",
    y="Reason",
    orientation='h',
    title="Check Sheet: Frequency of Delay Reasons",
    color="Frequency",
    color_continuous_scale='Viridis'
```

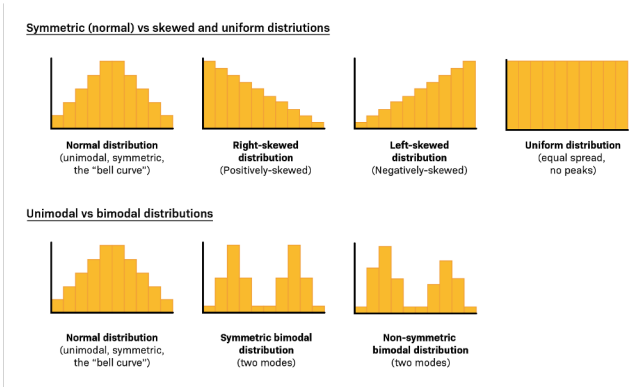
```
)
fig.update_layout(margin=dict(l=120), xaxis_title="Frequency", yaxis_title="Reason")
```



8.8.2 Histogram

Once the data is collected, the next step is to present it in a visual format that is easy to understand. A histogram is used to **see how often a problem or risk occurs**, so we can begin recognizing patterns in the events.

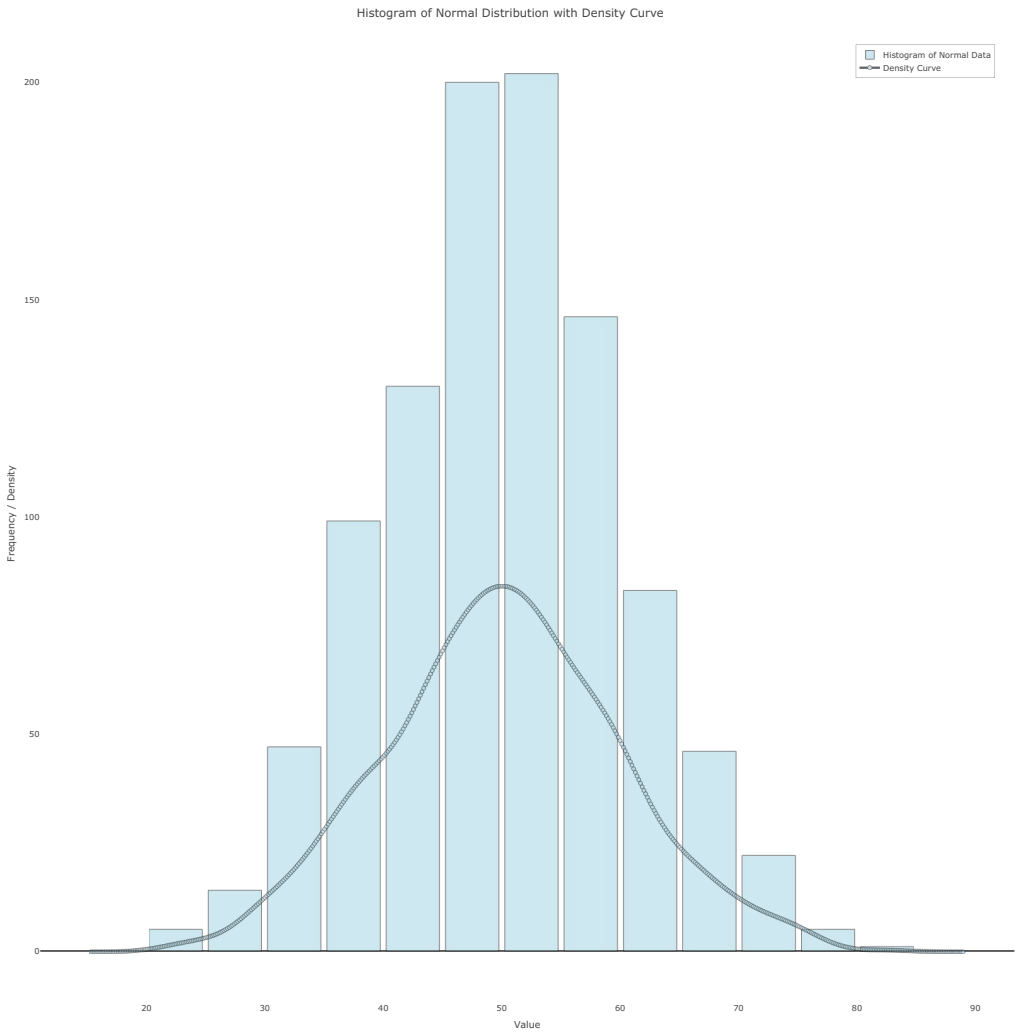
Aspect	Description
Brief Definition	A bar graph that shows the frequency distribution of data within a certain interval.
Main Function	- Shows how often an event occurs- Identifies distribution patterns- Finds process variations
Case Example	The team creates a histogram from the check sheet data to see how many times delays occur at 8 AM, 9 AM, 10 AM, etc.



8.8.2.1 Histogram in R

```
# Install and load necessary libraries
library(plotly)

# Generate normal distribution data (mean = 50, standard deviation = 10)
set.seed(123)
normal_data <- rnorm(1000, mean = 50, sd = 10)  # 1000 random data points

# Calculate the density of the normal data
density_data <- density(normal_data)

# Create a histogram of the normal data using plotly
histogram_plot <- plot_ly(
  x = normal_data,
  type = 'histogram',
  marker = list(color = 'lightblue', line = list(color = 'black', width = 1)),
  name = 'Histogram of Normal Data',
  nbinsx = 30,
  opacity = 0.6,
  showlegend = TRUE
) %>%
  # Add the density curve
  add_trace(
    x = density_data$x,
    y = density_data$y * length(normal_data) * diff(range(normal_data)) / 30,
    type = 'scatter',
    mode = 'lines',
    name = 'Density Curve',
    line = list(color = 'black', width = 3),
    showlegend = TRUE
  ) %>%
  layout(
    title = 'Histogram of Normal Distribution with Density Curve',
    xaxis = list(title = 'Value', showgrid = FALSE),
    yaxis = list(title = 'Frequency / Density', showgrid = FALSE),
```

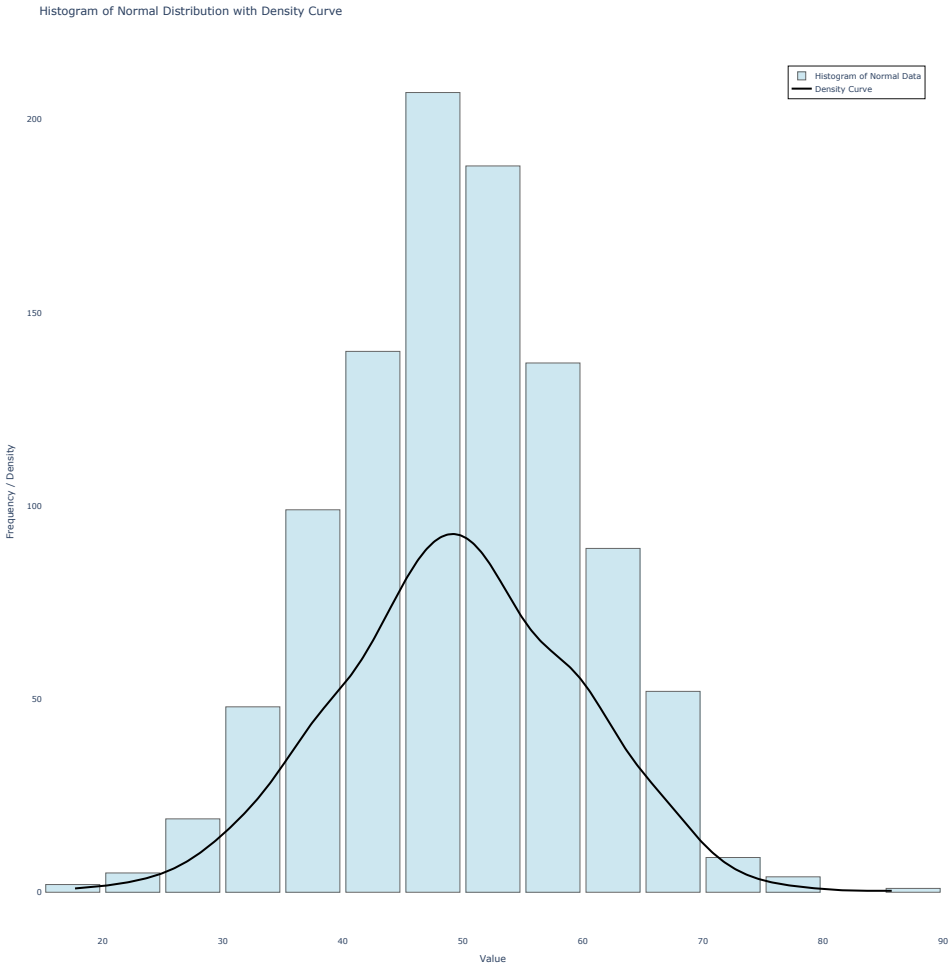
```
    bargap = 0.1,
    plot_bgcolor = 'white',
    paper_bgcolor = 'white',
    showlegend = TRUE,
    legend = list(
      orientation = 'v',    # vertical legend
      x = 0.98,             # almost at the right edge
      xanchor = 'right',
      y = 0.98,             # almost at the top
      yanchor = 'top',
      bgcolor = 'rgba(255,255,255,0.8)',  # semi-transparent background
      bordercolor = 'black',
      borderwidth = 0.3
    )
  )

# Show the plot
histogram_plot
```



8.8.2.2 Histogram in Python

```
import numpy as np
import plotly.graph_objs as go
from scipy.stats import gaussian_kde

# Generate normal distribution data (mean = 50, std = 10)
np.random.seed(123)
normal_data = np.random.normal(loc=50, scale=10, size=1000)

# Calculate density
density = gaussian_kde(normal_data)
x_density = np.linspace(min(normal_data), max(normal_data), 500)
y_density = density(x_density)

# Scale the density to match histogram frequency scale
```

```
hist_counts, hist_bins = np.histogram(normal_data, bins=30)
bin_width = hist_bins[1] - hist_bins[0]
scaling_factor = len(normal_data) * bin_width
y_density_scaled = y_density * scaling_factor

# Create histogram trace
histogram_trace = go.Histogram(
    x=normal_data,
    nbinsx=30,
    name='Histogram of Normal Data',
    marker=dict(color='lightblue', line=dict(color='black', width=1)),
    opacity=0.6
)

# Create density curve trace (placed *after* so it appears on top)
density_trace = go.Scatter(
    x=x_density,
    y=y_density_scaled,
    mode='lines',
    name='Density Curve',
    line=dict(color='black', width=3)
)

# Layout
layout = go.Layout(
    title='Histogram of Normal Distribution with Density Curve',
    xaxis=dict(title='Value', showgrid=False),
    yaxis=dict(title='Frequency / Density', showgrid=False),
    plot_bgcolor='white',
    paper_bgcolor='white',
    bargap=0.1,
    legend=dict(
        orientation='v',
        x=0.98,
        xanchor='right',
        y=0.98,
        yanchor='top',
        bgcolor='rgba(255,255,255,0.8)',
        bordercolor='black',
        borderwidth=0.3
    )
)

# Combine traces and show the plot
fig = go.Figure(data=[histogram_trace, density_trace], layout=layout)
fig.show()
```

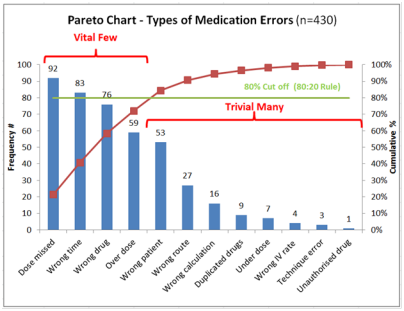


8.8.3 Pareto Chart

Pareto Chart is used to identify the main causes of a problem based on the 80/20 principle — that is, 80% of problems are caused by 20% of the causes. This chart combines a bar graph (showing frequency of problems) with a cumulative line graph (showing cumulative percentages), making it easier for users to focus on the most significant causes.

Aspect	Description
Brief Definition	A combination of bar and line charts that illustrates the 80/20 rule, where the majority of problems come from a few major causes.
Main Functions	- Prioritize the main causes- Focus on improvements with high impact- Develop mitigation strategies based on data

Aspect	Description
Example Case	Based on histogram data, 80% of delivery delays are caused by three factors: traffic jams, late drivers, and incomplete addresses.

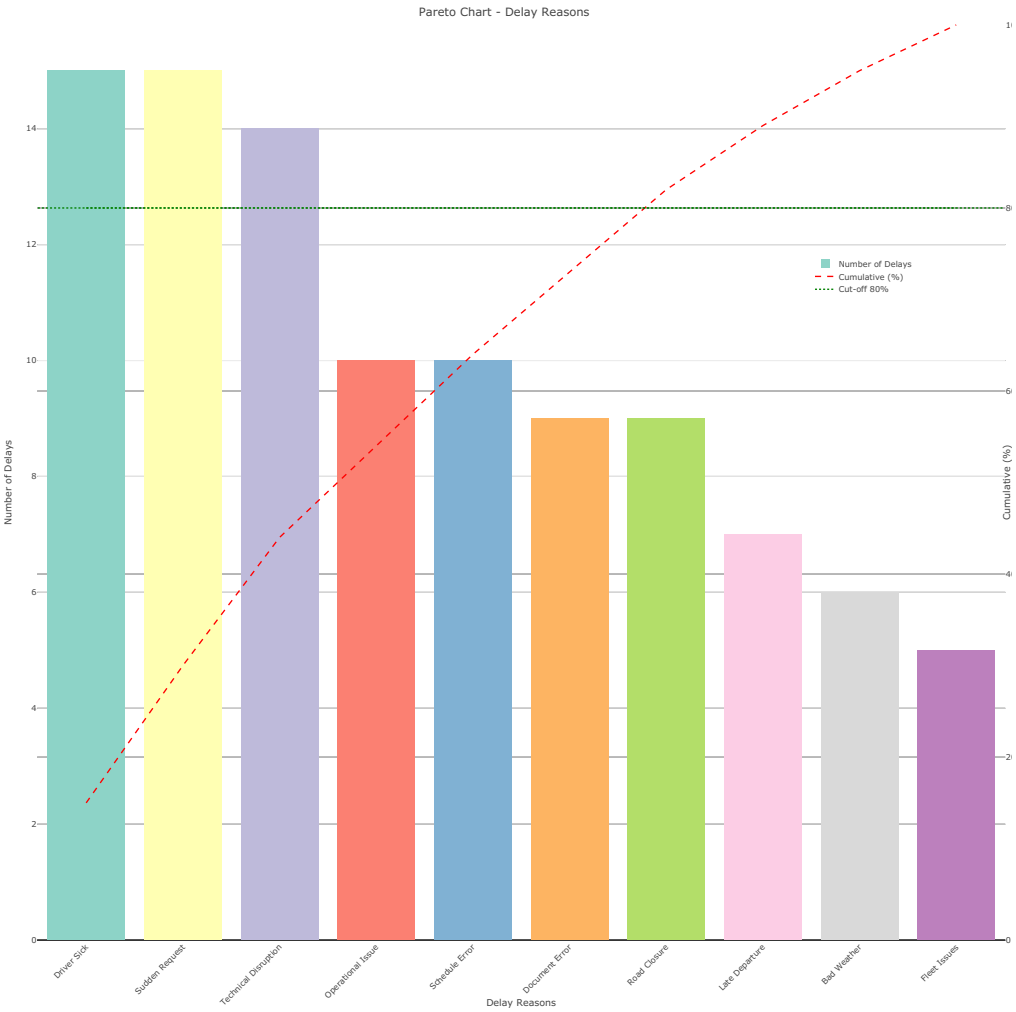


Case Example: A customer service manager records the main reasons for customer complaints over the course of one month.

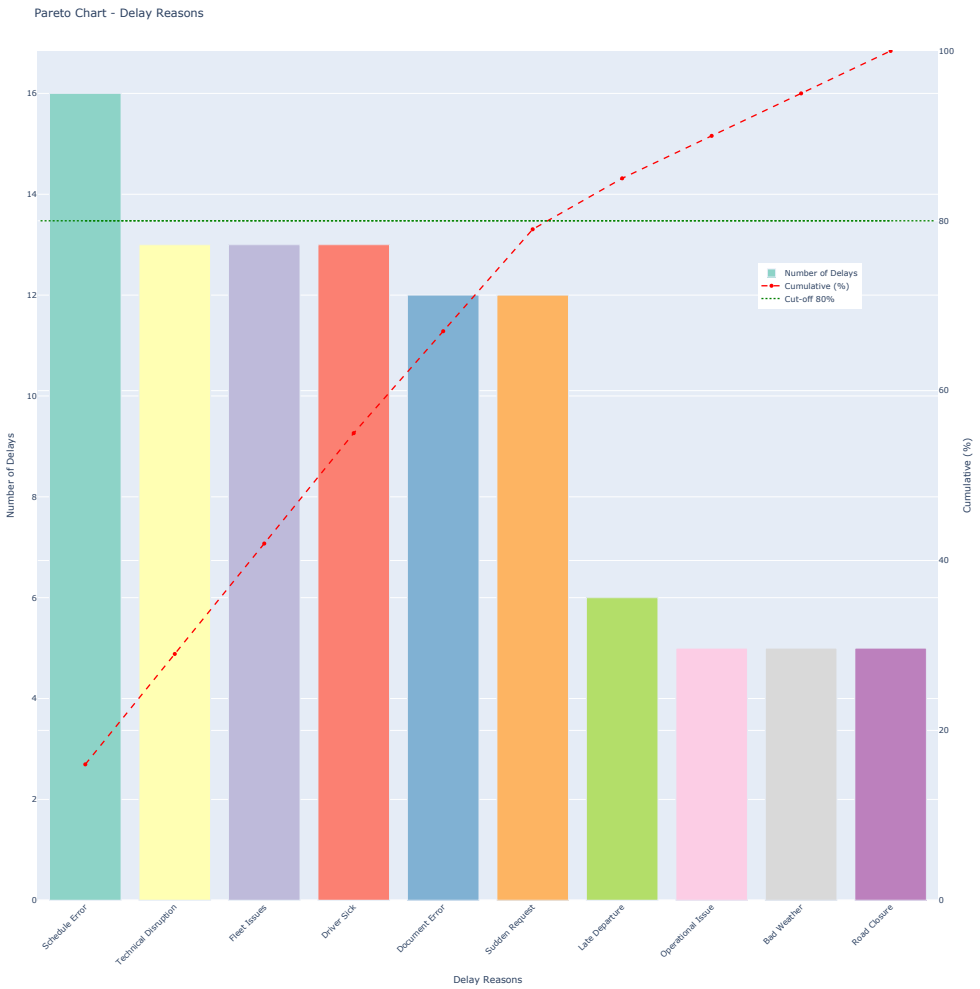
8.8.3.1 Pareto in R

```r
# Load libraries
library(dplyr)
library(plotly)

# Summarize the number of delays by reason
pareto_data <- check_sheet %>%
  count(Reason, sort = TRUE) %>%
  mutate(
    cum_freq = cumsum(n) / sum(n) * 100  # cumulative percentage
  )

# Create different colors for each Reason
colors <- RColorBrewer::brewer.pal(n = length(pareto_data$Reason), name = "Set3")

# Create Plotly Pareto Chart
fig <- plot_ly()

# Add Bar Chart (Count) - with different colors
fig <- fig %>% add_bars(
  x = ~reorder(pareto_data$Reason, -pareto_data$n),
  y = ~pareto_data$n,
  name = 'Number of Delays',
  marker = list(color = colors),
  yaxis = "y1"
)

# Add Cumulative Line
fig <- fig %>% add_lines(
  x = ~reorder(pareto_data$Reason, -pareto_data$n),
```

```
  y = ~pareto_data$cum_freq,
  name = 'Cumulative (%)',
  yaxis = "y2",
  line = list(color = 'red', dash = 'dash')
)

# Add Cut-off Line at 80%
fig <- fig %>% add_lines(
  x = ~reorder(pareto_data$Reason, -pareto_data$n),
  y = rep(80, length(pareto_data$Reason)),
  name = 'Cut-off 80%',
  yaxis = "y2",
  line = list(color = 'green', dash = 'dot')
)

# Adjust layout
fig <- fig %>% layout(
  title = "Pareto Chart - Delay Reasons",
  xaxis = list(
    title = "Delay Reasons",
    tickangle = -45   # tilt 45 degrees
  ),
  yaxis = list(title = "Number of Delays"),
  yaxis2 = list(
    title = "Cumulative (%)",
    overlaying = "y",
    side = "right",
    range = c(0, 100)
  ),
  legend = list(x = 0.8, y = 0.75),
  shapes = list(
    list(
      type = "line",
      x0 = -0.5,
      x1 = length(pareto_data$Reason) - 0.5,
      y0 = 80,
      y1 = 80,
      yref = "y2",
      line = list(color = "green", width = 2, dash = "dot")
    )
  )
)

# Show chart
fig
```


Pareto Chart - Delay Reasons
Number of Delays
Cumulative (%)
Cut-off 80%
Number of Delays
Cumulative (%)
Delay Reasons
Driver Sick
Sudden Request
Technical Disruption
Operational Issue
Schedule Error
Document Error
Road Closure
Late Departure
Bad Weather
Fleet Issues

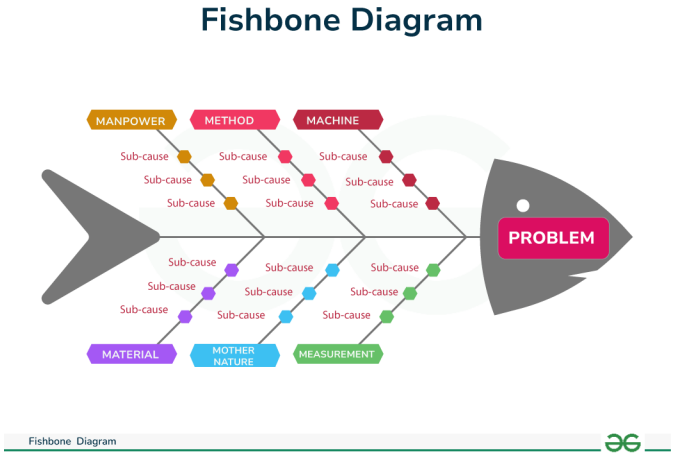
8.8.3.2 Pareto in Python



8.8.4 Fishbone

This tool is useful for **digging into root causes** thoroughly. The Fishbone Diagram divides causes into general categories, making it ideal for brainstorming sessions aimed at risk mitigation.

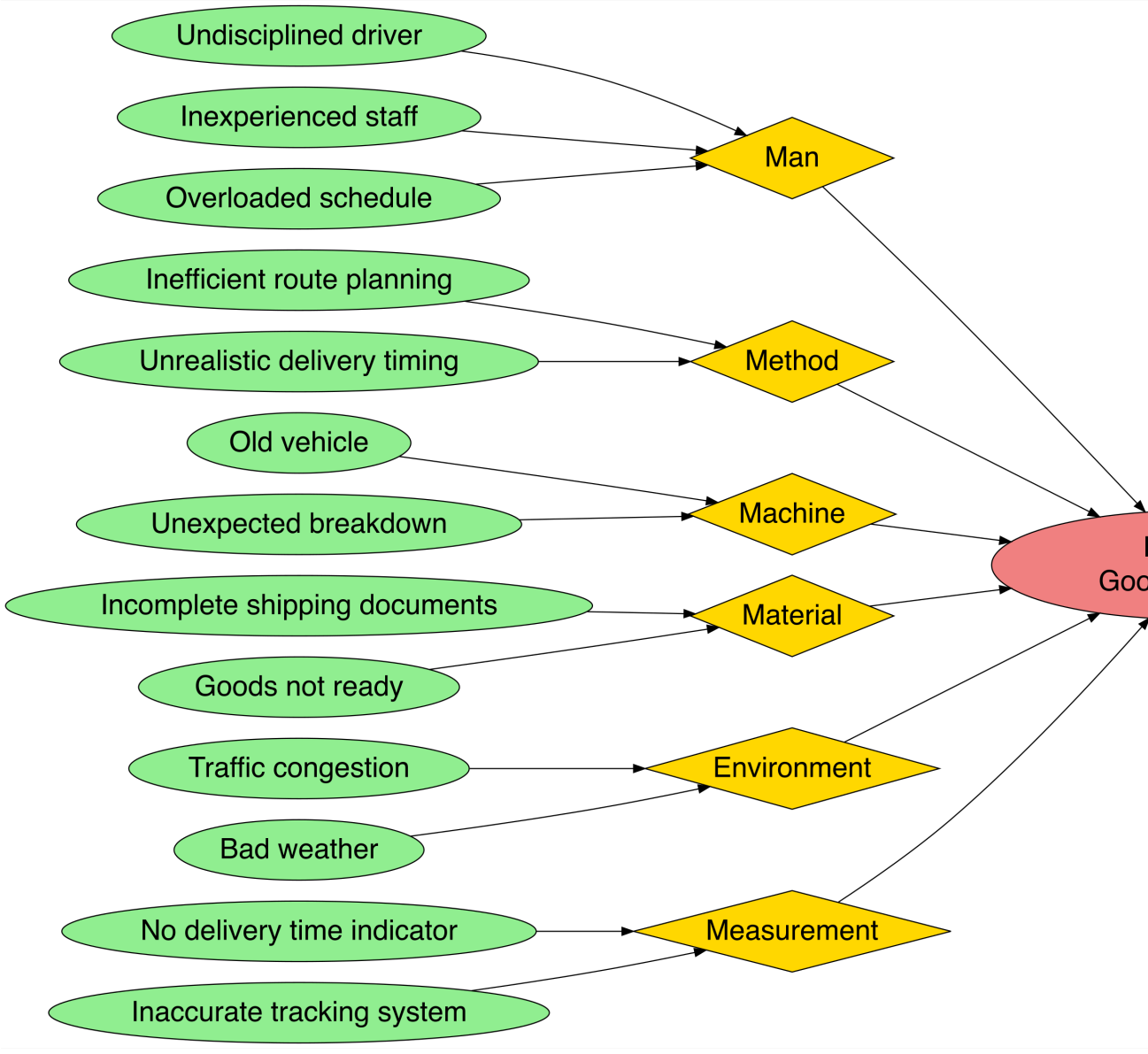
Aspect	Description
Brief Definition	A fishbone diagram used to identify and categorize potential causes of a problem.
Main Function	- Digging into root causes- Categorizing causes into groups (Man, Machine, Method, Material, etc.)
Case Example	Delays identified due to “Man” (driver lack of discipline), “Method” (inefficient route schedule), and “Machine” (old vehicle).



8.8.4.1 Fishbone in R

```
library(DiagrammeR)
library(rsvg)

graph <- grViz("
digraph fishbone {
  graph [layout = dot, rankdir = LR]

  # Default node styles
  node [fontname=Helvetica, fontsize=25, style=filled]

  # Central problem
  Problem [label='Delayed \\n Goods Delivery', shape=ellipse, fillcolor=lightcoral, width=5.0

  # Category nodes (shared style)
  node [shape=diamond, width=2.5, height=1.0, fillcolor='#FFD700']
  A1 [label='Man']
  A2 [label='Method']
  A3 [label='Machine']
  A4 [label='Material']
  A5 [label='Environment']
  A6 [label='Measurement']

  # Reset node style for sub-categories
  node [shape=ellipse, width=2.5, height=0.6, fillcolor='#90EE90']
  A1a [label='Undisciplined driver']
  A1b [label='Inexperienced staff']
  A1c [label='Overloaded schedule']

  A2a [label='Inefficient route planning']
  A2b [label='Unrealistic delivery timing']
```

```
  A3a [label='Old vehicle']
  A3b [label='Unexpected breakdown']

  A4a [label='Incomplete shipping documents']
  A4b [label='Goods not ready']

  A5a [label='Traffic congestion']
  A5b [label='Bad weather']

  A6a [label='No delivery time indicator']
  A6b [label='Inaccurate tracking system']

  # Relationships
  A1 -> Problem
  A2 -> Problem
  A3 -> Problem
  A4 -> Problem
  A5 -> Problem
  A6 -> Problem

  A1a -> A1
  A1b -> A1
  A1c -> A1

  A2a -> A2
  A2b -> A2

  A3a -> A3
  A3b -> A3

  A4a -> A4
  A4b -> A4

  A5a -> A5
  A5b -> A5

  A6a -> A6
  A6b -> A6
}
")

# Output directory and saving
dir.create("images/bab8", recursive = TRUE, showWarnings = FALSE)
svg_code <- export_svg(graph)
rsvg_png(charToRaw(svg_code), file = "images/bab8/fishbone_delivery_en.png", width = 3000,
rsvg_pdf(charToRaw(svg_code), file = "images/bab8/fishbone_delivery_en.pdf")

knitr::include_graphics("images/bab8/fishbone_delivery_en.png")
```


Undisciplined driver
Inexperienced staff
Overloaded schedule
Man
Inefficient route planning
Unrealistic delivery timing
Method
Old vehicle
Unexpected breakdown
Machine
Incomplete shipping documents
Material
Goods not ready
Traffic congestion
Environment
Bad weather
No delivery time indicator
Measurement
Inaccurate tracking system
Goo

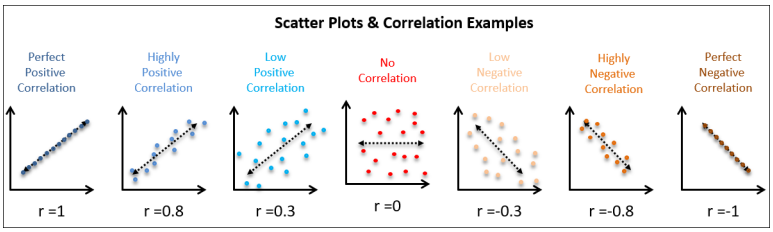
8.8.4.2 Fishbone in Python

```
Your job
```

8.8.5 Scatter Diagram

To examine the **relationship between two risk variables**, the Scatter Diagram is an ideal tool. For example, is there a relationship between the number of deliveries and the frequency of delays?

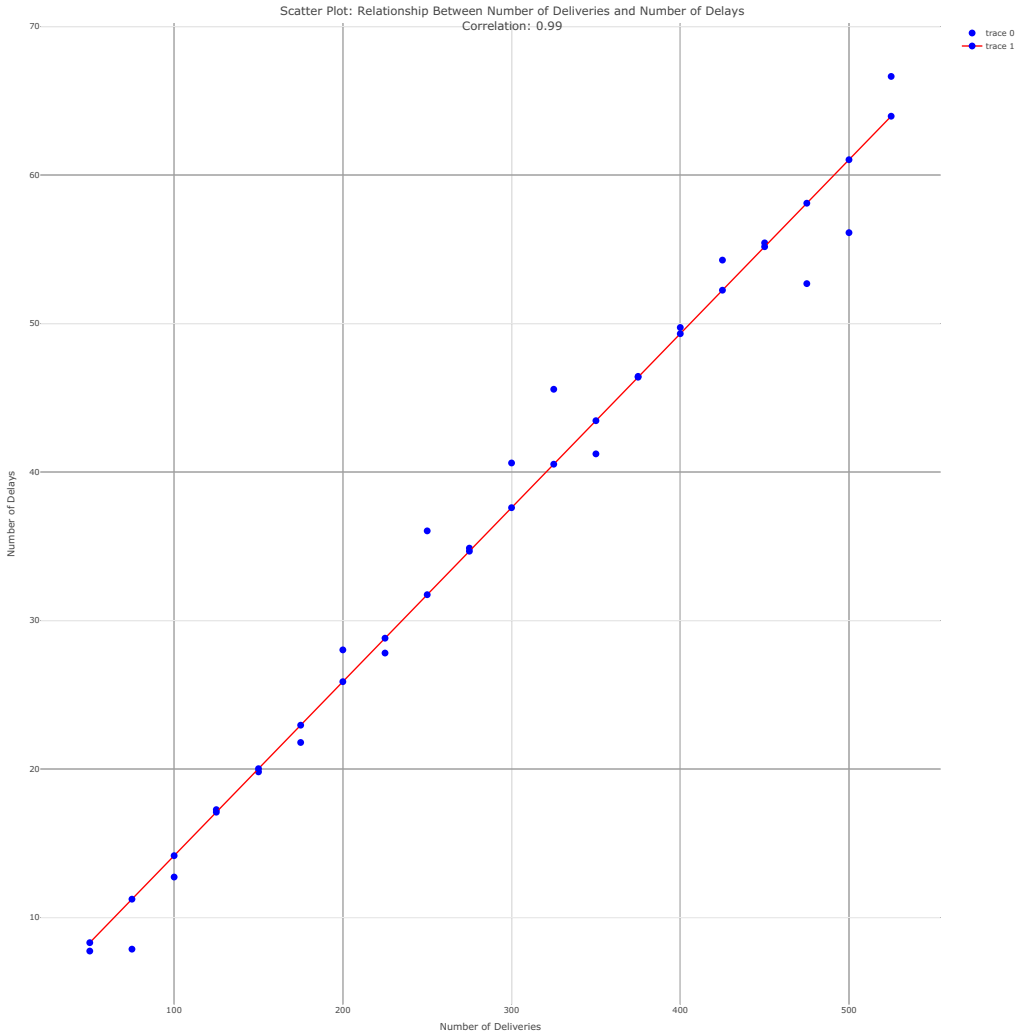
Aspect	Description
Brief Definition	A scatter diagram used to observe the relationship between two variables.
Main Function	- Determines if two variables are related- Supports correlation analysis
Case Example	The team examines the relationship between the number of daily deliveries and the number of delays. The result shows that as the number of deliveries increases, delays also increase.



8.8.5.1 Scatter Diagram in R

```r
# Install the required package if not already installed
# install.packages("plotly")

# Load the plotly package
library(plotly)

# Create a dataset with correlation around 0.90
set.seed(42)  # For reproducibility
data <- data.frame(
  Number_of_Deliveries = c(50, 75, 100, 125, 150, 175, 200, 225, 250, 275,
                           300, 325, 350, 375, 400, 425, 450, 475, 500, 525),
  Number_of_Delays = c(5, 9, 12, 16, 19, 22, 25, 28, 32, 35,
                       38, 41, 44, 47, 50, 53, 56, 58, 61, 64)
)

# Introduce some noise to get a correlation of around 0.9
data$Number_of_Delays <- data$Number_of_Delays + rnorm(n = 20, mean = 0, sd = 2)

# Calculate the correlation coefficient between the number of deliveries and the number of
```

```r
correlation_value <- cor(data$Number_of_Deliveries, data$Number_of_Delays)

# Create a scatter plot using Plotly with a linear regression line
fig <- plot_ly(data,
               x = ~Number_of_Deliveries,
               y = ~Number_of_Delays,
               type = 'scatter',
               mode = 'markers',
               marker = list(color = 'blue', size = 10)) %>%
  add_lines(x = data$Number_of_Deliveries,
            y = predict(lm(Number_of_Delays ~ Number_of_Deliveries, data = data)),
            line = list(color = 'red', dash = 'solid', width = 2)) %>%
  layout(title = paste("Scatter Plot: Relationship Between Number of Deliveries and Number of
         xaxis = list(title = "Number of Deliveries"),
         yaxis = list(title = "Number of Delays"))

# Show the plot
fig
```



8.8.5.2 Scatter Diagram in Python

```
Your job
```

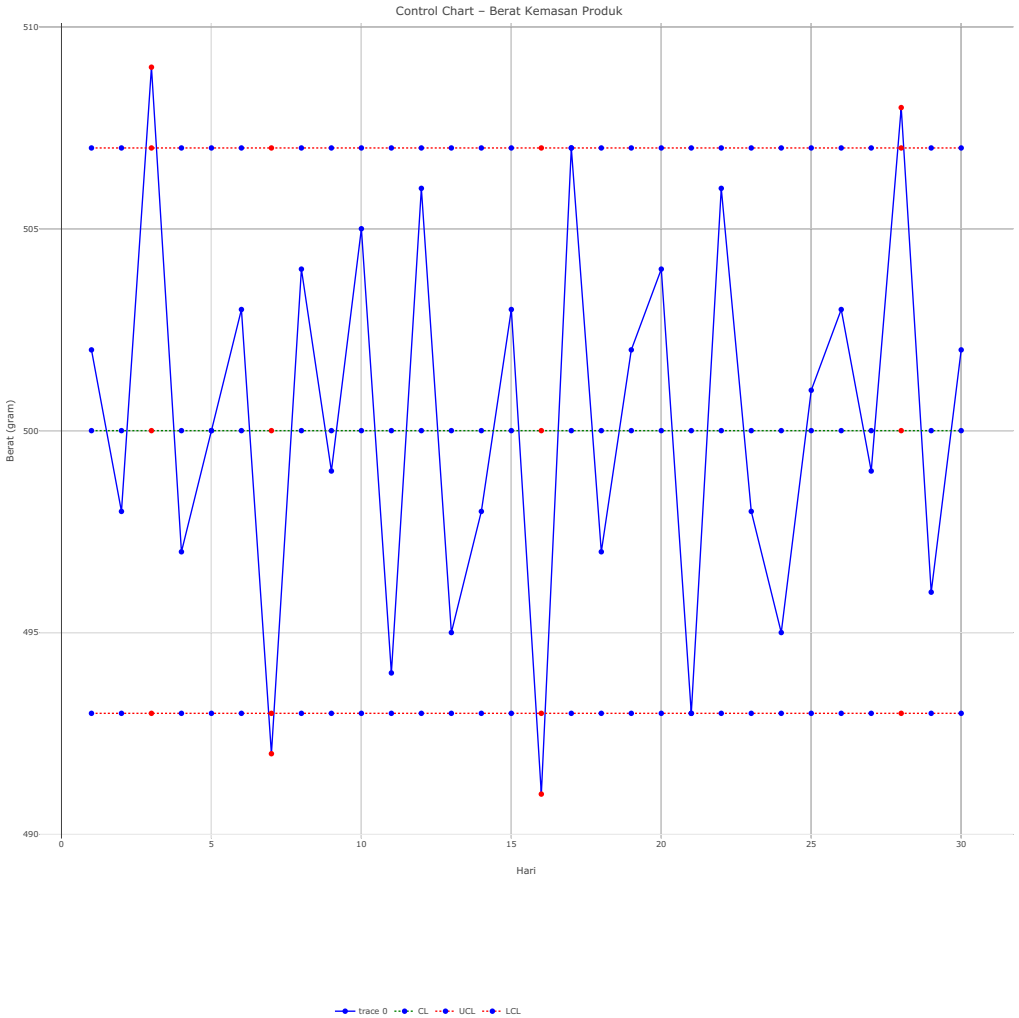
8.8.6 Control Chart

When a process is running, we need to **ensure its stability**. The Control Chart is used to monitor whether the variation in risk is still within acceptable limits or is heading toward dangerous deviations.

Aspect	Description
Brief Definition	A chart used to monitor process variation over time and identify whether the process is within statistical control.
Main Function	- Monitors process stability- Identifies deviations from control limits

Aspect	Description
Case Example	The Control Chart shows that in the second week, the variation in delays exceeded the upper control limit → indicating a specific issue occurred.

```r
# Load libraries
library(plotly)
library(dplyr)

# Buat data berat produk
berat_data <- data.frame(
  Hari = 1:30,
  Berat = c(
    502, 498, 509, 497, 500, 503, 492, 504, 499, 505,
    494, 506, 495, 498, 503, 491, 507, 497, 502, 504,
    493, 506, 498, 495, 501, 503, 499, 508, 496, 502
  )
)

# Tentukan batas kendali
CL <- 500
UCL <- 507
LCL <- 493

# Tandai outliers
berat_data <- berat_data %>%
  mutate(Outlier = ifelse(Berat > UCL | Berat < LCL, "Ya", "Tidak"))

# Buat plot
plot_ly(berat_data, x = ~Hari, y = ~Berat, type = 'scatter', mode = 'lines+markers',
        line = list(color = 'blue'),
        marker = list(size = 8, color = ifelse(berat_data$Outlier == "Ya", "red", "blue")),
        hoverinfo = 'text',
        text = ~paste("Hari:", Hari, "<br>Berat:", Berat, "gram")) %>%
  add_lines(y = rep(CL, 30), name = "CL", line = list(color = 'green', dash = 'dot')) %>%
  add_lines(y = rep(UCL, 30), name = "UCL", line = list(color = 'red', dash = 'dot')) %>%
  add_lines(y = rep(LCL, 30), name = "LCL", line = list(color = 'red', dash = 'dot')) %>%
  layout(title = "Control Chart - Berat Kemasan Produk",
         xaxis = list(title = "Hari"),
         yaxis = list(title = "Berat (gram)"),
         legend = list(orientation = 'h', x = 0.3, y = -0.2))
```



8.8.6.2 Control Chart in Python

```
Your job
```

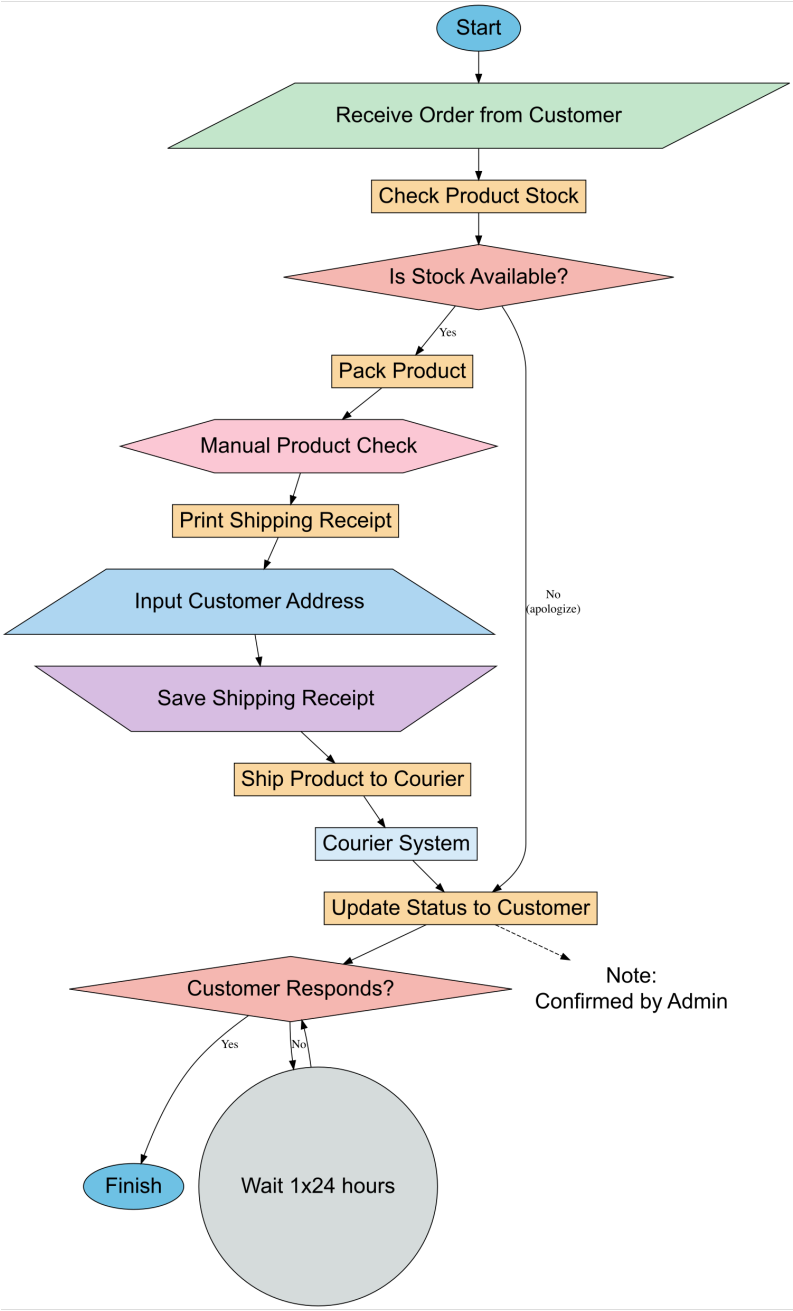
8.8.7 Flowchart

To understand the entire process flow and **identify potential obstacles or errors in the system**, a flowchart is very useful. It serves as a foundation for reviewing steps that need improvement.

Aspect	Description
Brief Definition	A process flow diagram that shows the stages or steps in a system or activity.
Main Function	- Visually maps the process flow- Identifies stages prone to errors or obstacles

Aspect	Description
Case Example	A flowchart is used to depict the process of distributing goods from the warehouse to the customer, highlighting that address verification is often overlooked.

8.8.7.1 Flowchart in R



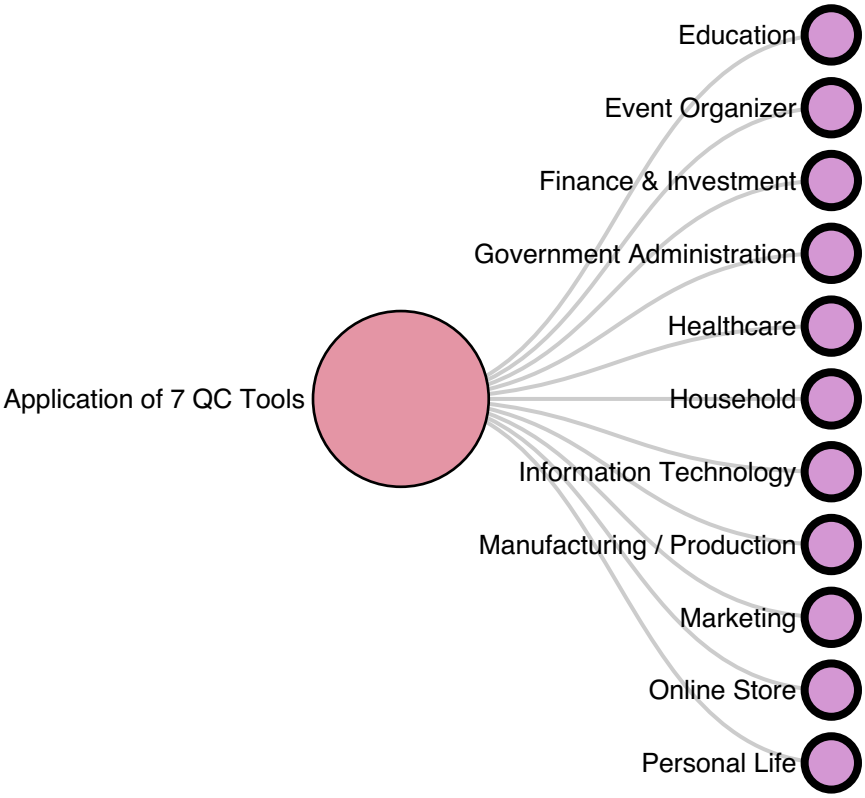
```
Your job
```

8.9 Discussion Materials

Topic	Trigger Questions	Notes / Personal Reflection
Relevance & Experience	• Have you encountered recurring problems in your work or life? • Which QC tool is most suitable to help understand or solve those problems?	
Readiness for Implementation	• Have you or your workplace already implemented a data-driven approach? • What are the main challenges in starting to use the 7 QC Tools?	
Impact & Change	• What is the difference in the impact of decisions based on assumptions vs. data? • Imagine a small change you could start tomorrow using one of the QC Tools.	
Team Collaboration	• How does team collaboration play a role in the use of QC tools? • How can you involve more team members in implementing QC Tools?	
Follow-up Actions	• Which QC tool would you like to learn more about? • Do you need training, mentoring, or real case studies to delve deeper into it?	

8.10 Several Applications

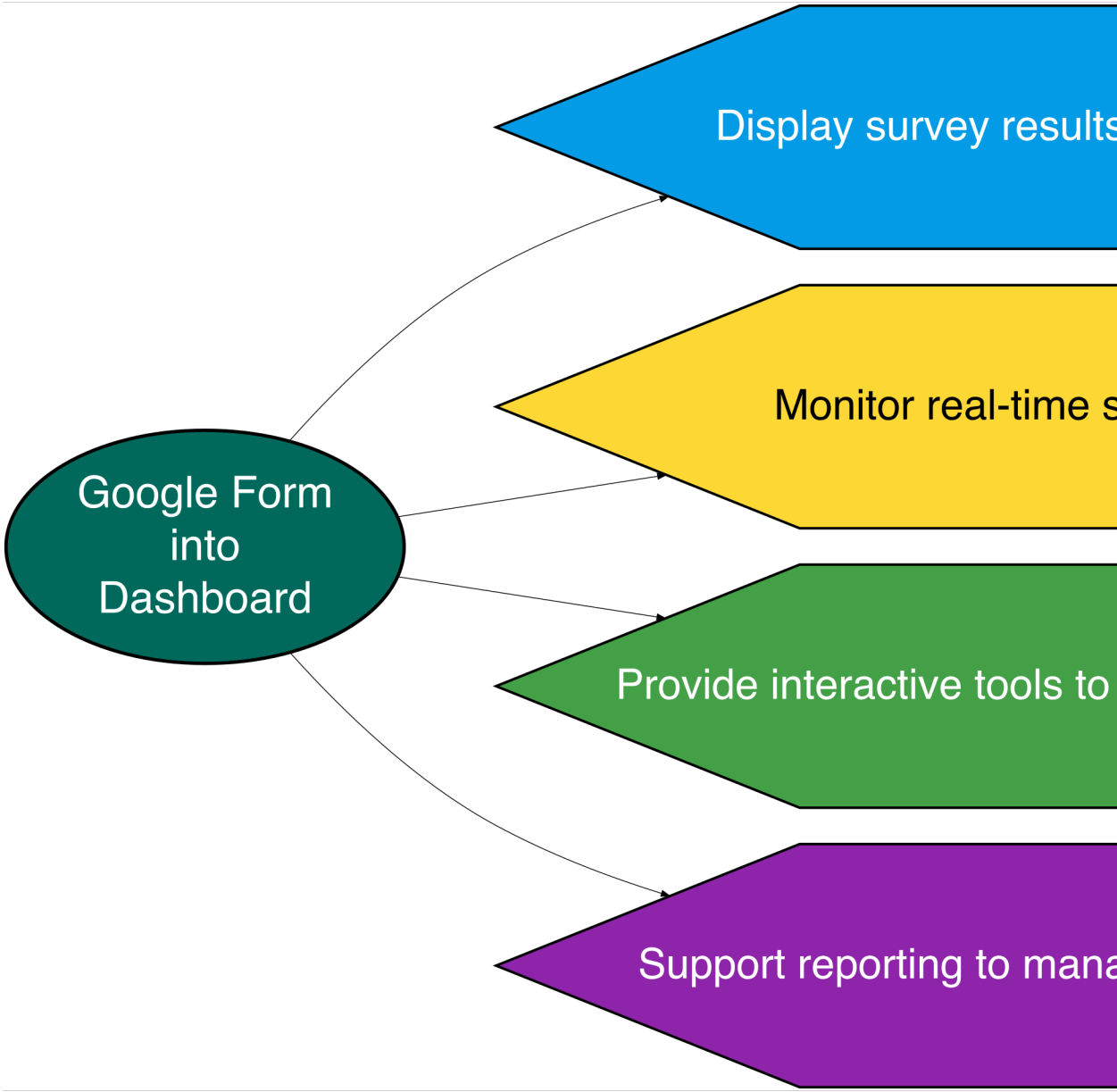
The Application of 7 QC Tools in Business & Everyday Life


Application of 7 QC Tools
Education
Event Organizer
Finance & Investment
Government Administration
Healthcare
Household
Information Technology
Manufacturing / Production
Marketing
Online Store
Personal Life

Chapter 9

Form Survey into Dashboard

Integrating survey results into dashboards offers key advantages in **analysis speed**, **dynamic visualization**, and **data-driven decision making**.



9.1 Key Dashboard Elements

An effective survey dashboard is made up of several interactive and visual components that help users explore and interpret data easily. Below is a list of common elements found in most survey dashboards, along with their core functions.

Component	Function
KPI/Value Boxes	Show key metrics (e.g., total responses, average satisfaction)
Interactive Charts	Visualize distribution and trends using bar, pie, or line charts
Filters/Dropdowns	Allow data slicing by time, region, or user segment
Table/Data Grid	Display survey responses in detail, with export/download option
Word Cloud	Visualize keywords from open-ended responses
Map (Optional)	Show geographical distribution of respondents

9.2 Dashboard Frameworks

Choosing the right tool depends on your team's technical skills, data sources, and the complexity of your dashboard needs. Below is a comparison of popular platforms used to build survey dashboards, highlighting their key features and strengths.

Tool/Platform	Key Features
R (shiny / flexdashboard)	- Direct connection to Google Sheets or local databases - Open-source and highly customizable - Supports packages like `shinydashboard`, `plotly`, `DT`, `highcharter`
Python (dash / streamlit)	- Flexible for data analysis and machine learning - Interactive and responsive dashboards
Excel + PivotCharts + Slicers	- Ideal for quick, offline, or small-scale dashboards
Power BI / Tableau	- Professional-grade visuals with drag-and-drop UI - Easy integration with Google Sheets, cloud services, and databases

9.3 Conneting Google Form

9.3.1 Using R

1. **Create Google Form** → Automatically saves responses to Google Sheets.
2. **Use `googlesheets4` R package** to fetch survey data.
3. **Clean and analyze the data**.
4. **Build interactive dashboards** using `flexdashboard` or `shiny`.

```
# Load the googlesheets4 package to interact with Google Sheets
library(googlesheets4)

# Disable authentication - this allows reading only public Google Sheets
gs4_deauth()

# Define the URL of the Google Sheet you want to read
```

```r
sheet_url <- "https://docs.google.com/spreadsheets/d/1hNDRkTtvi7nO_SUW3JJtA4DOR7_wBOL_WG7f

# Read the data from the Google Sheet into R
survey_data <- read_sheet(sheet_url)

# Display the data that was read from the Google Sheet
survey_data
```

```
# A tibble: 1 x 8
  Timestamp           Staff di program studi membantu m~1 Dosen memperhatikan ~2
  <dttm>              <chr>                               <chr>
1 2025-05-19 11:44:58 Sangat memuaskan                    Sangat memuaskan
# i abbreviated names:
#   1: `Staff di program studi membantu mahasiswa dalam layanan akademik`,
#   2: `Dosen memperhatikan kehadiran mahasiswa di dalam perkuliahan`
# i 5 more variables:
#   `Proses perkuliahan mengutamakan pendekatan ilmiah seperti (PBL, Studi kasus, proyek c
#   `Proses pembelajaran relevan dengan karakteristik keilmuan program studi dan memberi k
#   `Fasilitas, sarana dan prasarana (gedung-gedung, ruangan kelas, laboratorium, taman, c
```

9.3.2 Using Python

1. Create Your Google Form

- Go to Google Forms.
- Create a new form and add questions.
- Click on "Responses" tab → click the Google Sheets icon to link responses to a spreadsheet.

2. Get the Public CSV Link from Google Sheets

- Open the linked Google Sheet.
- Click Share → set the sharing to:
- Anyone with the link can view
- Copy the sheet URL.

3. Read the Data in Python using pandas

For example:

```python
import pandas as pd

# Use Google Sheet CSV export link
sheet_url = "https://docs.google.com/spreadsheets/d/1hNDRkTtvi7nO_SUW3JJtA4DOR7_wBOL_WG7f-
survey_data = pd.read_csv(sheet_url)

# Show the data
survey_data
```

```
             Timestamp  ...           Email address
0  19/05/2025 11:44:58  ...  siregarbakti@gmail.com

[1 rows x 8 columns]
```

9.4 Shiny App

Chapter 10

Case Study in Surveys