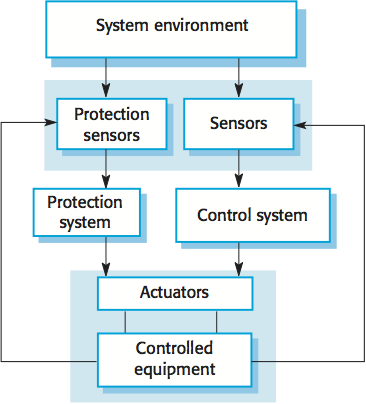
**Fault tolerance**

In critical situations, software systems must be fault tolerant. Fault tolerance is required where there are **high availability requirements or where system failure costs are very high**. Fault tolerance means that the system can continue in operation in spite of software failure. Even if the system has been proved to conform to its specification, it must also be fault tolerant as there may be specification errors or the validation may be incorrect.

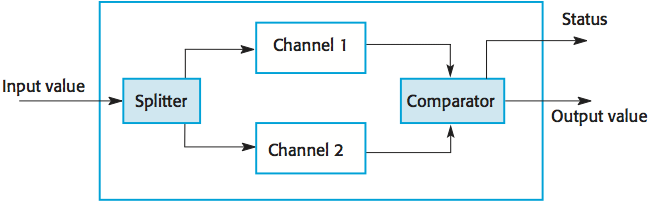
**Fault-tolerant systems architectures** are used in situations where fault tolerance is essential. These architectures are generally all based on redundancy and diversity. Examples of situations where dependable architectures are used:

* Flight control systems, where system failure could threaten the safety of passengers;
* Reactor systems where failure of a control system could lead to a chemical or nuclear emergency;
* Telecommunication systems, where there is a need for 24/7 availability.

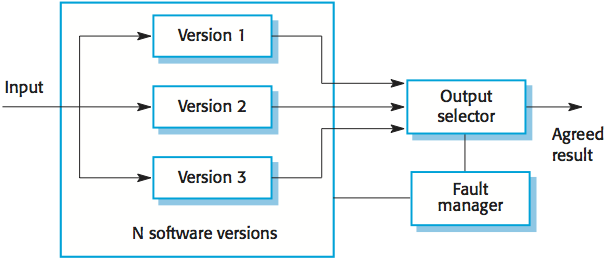
**Protection system** is a specialized system that is associated with some other control system, which can take emergency action if a failure occurs, e.g. a system to stop a train if it passes a red light, or a system to shut down a reactor if temperature/pressure are too high. Protection systems independently monitor the controlled system and the environment. If a problem is detected, it issues commands to take emergency action to shut down the system and avoid a catastrophe. Protection systems are redundant because they include monitoring and control capabilities that replicate those in the control software. Protection systems should be diverse and use different technology from the control software. They are simpler than the control system so more effort can be expended in validation and dependability assurance. Aim is to ensure that there is a low probability of failure on demand for the protection system.



**Self-monitoring architecture** is a multi-channel architectures where the system monitors its own operations and takes action if inconsistencies are detected. The same computation is carried out on each channel and the results are compared. If the results are identical and are produced at the same time, then it is assumed that the system is operating correctly. If the results are different, then a failure is assumed and a failure exception is raised. Hardware in each channel has to be diverse so that common mode hardware failure will not lead to each channel producing the same results. Software in each channel must also be diverse, otherwise the same software error would affect each channel. If high-availability is required, you may use several self-checking systems in parallel. This is the approach used in the Airbus family of aircraft for their flight control systems.



**N-version programming** involves multiple versions of a software system to carry out computations at the same time. There should be an odd number of computers involved, typically 3. The results are compared using a voting system and the majority result is taken to be the correct result. Approach derived from the notion of triple-modular redundancy, as used in hardware systems.



**Hardware fault tolerance** depends on triple-modular redundancy (TMR). There are three replicated identical components that receive the same input and whose outputs are compared. If one output is different, it is ignored and component failure is assumed. Based on most faults resulting from component failures rather than design faults and a low probability of simultaneous component failure.

**Differences between hardware and software reliability**

|  |  |
| --- | --- |
| Hardware | software |
| Failure rate has a bathtub curve | Without considering program evolution, failure rate is statistically non-increasing |
| Material deterioration can cause failures even though the system is not used | Failures never occur if the software is not used |
| Failure data are fitted to some distributions. The selection of the underlying distribution is based on the analysis of failure data and experiences. Emphasis is placed on analyzing failure data | Most models are analytically derived from assumptions. Emphasis is on developing the model, the interpretation of the model assumptions, and the physical meaning of the parameters |
| Failures are caused by material deterioration, random failures, design errors, misuse, and environment | Failures are caused by incorrect logic, incorrect statements, or incorrect input data. This is similar to design errors of a complex hardware system |
| Hardware reliability can be improved by better design, better material, applying redundancy and accelerated life testing | Software reliability can be improved by increasing the testing effort and by correcting detected faults |
| Hardware repairs restore the original condition | Software repairs establish a new piece of software |
| Software repairs establish a new piece of software | Software repairs establish a new piece of software |

**Programming for reliability**

Good programming practices can be adopted that help reduce the incidence of program faults. These programming practices support fault avoidance, detection, and tolerance.

**Limit the visibility of information in a program**

Program components should only be allowed access to data that they need for their implementation. This means that accidental corruption of parts of the program state by these components is impossible. You can control visibility by using abstract data types where the data representation is private and you only allow access to the data through predefined operations such as get () and put ().

**Check all inputs for validity**

All program take inputs from their environment and make assumptions about these inputs. However, program specifications rarely define what to do if an input is not consistent with these assumptions. Consequently, many programs behave unpredictably when presented with unusual inputs and, sometimes, these are threats to the security of the system. Consequently, you should always check inputs before processing against the assumptions made about these inputs.

**Provide a handler for all exceptions**

A program exception is an error or some unexpected event such as a power failure. Exception handling constructs allow for such events to be handled without the need for continual status checking to detect exceptions. Using normal control constructs to detect exceptions needs many additional statements to be added to the program. This adds a significant overhead and is potentially error-prone.

**Minimize the use of error-prone constructs**

Program faults are usually a consequence of human error because programmers lose track of the relationships between the different parts of the system This is exacerbated by error-prone constructs in programming languages that are inherently complex or that don't check for mistakes when they could do so. Therefore, when programming, you should try to avoid or at least minimize the use of these error-prone constructs.

Error-prone constructs:

* Unconditional branch (goto) statements
* Floating-point numbers (inherently imprecise, which may lead to invalid comparisons)
* Pointers
* Dynamic memory allocation
* Parallelism (can result in subtle timing errors because of unforeseen interaction between parallel processes)
* Recursion (can cause memory overflow as the program stack fills up)
* Interrupts (can cause a critical operation to be terminated and make a program difficult to understand)
* Inheritance (code is not localized, which may result in unexpected behavior when changes are made and problems of understanding the code)
* Aliasing (using more than 1 name to refer to the same state variable)
* Unbounded arrays (may result in buffer overflow)
* Default input processing (if the default action is to transfer control elsewhere in the program, incorrect or deliberately malicious input can then trigger a program failure)

**Provide restart capabilities**

For systems that involve long transactions or user interactions, you should always provide a restart capability that allows the system to restart after failure without users having to redo everything that they have done.

**Check array bounds**

In some programming languages, such as C, it is possible to address a memory location outside of the range allowed for in an array declaration. This leads to the well-known 'bounded buffer' vulnerability where attackers write executable code into memory by deliberately writing beyond the top element in an array. If your language does not include bound checking, you should therefore always check that an array access is within the bounds of the array.

**Include timeouts when calling external components**

In a distributed system, failure of a remote computer can be 'silent' so that programs expecting a service from that computer may never receive that service or any indication that there has been a failure. To avoid this, you should always include timeouts on all calls to external components. After a defined time period has elapsed without a response, your system should then assume failure and take whatever actions are required to recover from this.

**Name all constants that represent real-world values**

Always give constants that reflect real-world values (such as tax rates) names rather than using their numeric values and always refer to them by name You are less likely to make mistakes and type the wrong value when you are using a name rather than a value. This means that when these 'constants' change (for sure, they are not really constant), then you only have to make the change in one place in your program.

## What is Quality?

Quality is extremely hard to define, and it is simply stated: "Fit for use or purpose." It is all about meeting the needs and expectations of customers with respect to functionality, design, reliability, durability, & price of the product.

## What is Assurance?

Assurance is nothing but a positive declaration on a product or service, which gives confidence. It is certainty of a product or a service, which it will work well. It provides a guarantee that the product will work without any problems as per the expectations or requirements.

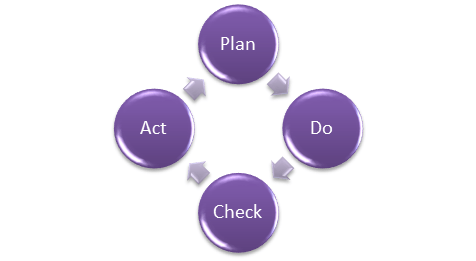
## Quality Assurance in Software Testing

**Quality Assurance in Software Testing** is defined as a procedure to ensure the quality of software products or services provided to the customers by an organization. Quality assurance focuses on improving the [software development process](https://www.guru99.com/software-development-life-cycle-tutorial.html) and making it efficient and effective as per the quality standards defined for software products. Quality Assurance is popularly known as QA Testing.

## How to do Quality Assurance: Complete Process

Quality Assurance methodology has a defined cycle called PDCA cycle or Deming cycle. The phases of this cycle are:

* Plan
* Do
* Check
* Act

[](https://www.guru99.com/images/Q2.png)Quality Assurance Process

These above steps are repeated to ensure that processes followed in the organization are evaluated and improved on a periodic basis. Let's look into the above QA Process steps in detail -

* Plan - Organization should plan and establish the process related objectives and determine the processes that are required to deliver a high-Quality end product.
* Do - Development and testing of Processes and also "do" changes in the processes
* Check - Monitoring of processes, modify the processes, and check whether it meets the predetermined objectives
* Act - A Quality Assurance tester should implement actions that are necessary to achieve improvements in the processes

An organization must use Quality Assurance to ensure that the product is designed and implemented with correct procedures. This helps reduce problems and errors, in the final product.

## What is Quality Control?

[](https://www.guru99.com/images/Q3.png)

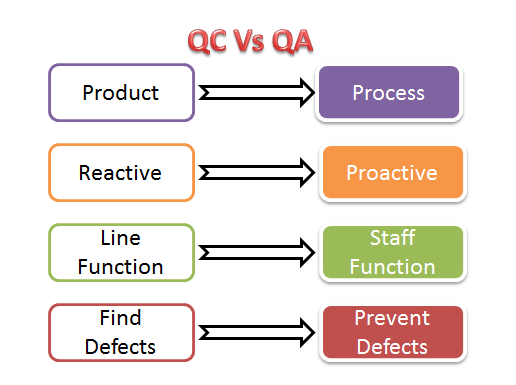
Quality control popularly abbreviated as QC. It is a Software Engineering process used to ensure quality in a product or a service. It does not deal with the processes used to create a product; rather it examines the quality of the "end products" and the final outcome.

The main aim of Quality control is to check whether the products meet the specifications and requirements of the customer. If an issue or problem is identified, it needs to be fixed before delivery to the customer.

QC also evaluates people on their quality level skill sets and imparts training and certifications. This evaluation is required for the service based organization and helps provide "perfect" service to the customers.

## Difference between Quality Control and Quality Assurance?

Sometimes, QC is confused with the QA. Quality control is to examine the product or service and check for the result. Quality Assurance in Software Engineering is to examine the processes and make changes to the processes which led to the end-product.

[](https://www.guru99.com/images/Q4.png)Quality Control Vs Quality Assurance

Examples of QC and QA activities are as follows:

|  |  |
| --- | --- |
| **Quality Control Activities** | **Quality Assurance Activities** |
| Walkthrough | Quality Audit |
| Testing | Defining Process |
| Inspection | Tool Identification and selection |
| Checkpoint review | Training of Quality Standards and Processes |

**The above activities are concerned with Quality Assurance and Control mechanisms for any product and not essentially software**. With respect to software

* QA becomes SQA ( Software Quality Assurance)
* QC becomes Software Testing**.**

## Differences between SQA and Software Testing

Following table explains on differences between SQA and Software Testing:

|  |  |
| --- | --- |
| **SQA** | **Software Testing** |
| Software Quality Assurance is about engineering process that ensures quality | Software Testing is to test a product for problems before the product goes live |
| Involves activities related to the implementation of processes, procedures, and standards. Example - Audits Training | Involves actives concerning verification of product Example - Review Testing |
| Process focused | Product focused |
| Preventive technique | Corrective technique |
| Proactive measure | Reactive measure |
| The scope of SQA applied to all products that will be created by the organization | The scope of Software Testing applies to a particular product being tested. |

## Best practices for Quality Assurance:

* Create a Robust Testing Environment
* Select release criteria carefully
* Apply [automated testing](https://www.guru99.com/automation-testing.html) to high-risk areas to save money. It helps to fasten the entire process.
* Allocate Time Appropriately for each process
* It is important to prioritize bugs fixes based on software usage
* Form dedicated security and performance testing team
* Simulate customer accounts similar to a production environment

## Quality Assurance Functions:

There are 5 primary Quality Assurance Functions:

1. **Technology transfer:** This function involves getting a product design document as well as trial and error data and its evaluation. The documents are distributed, checked and approved
2. **Validation:** Here validation master plan for the entire system is prepared. Approval of test criteria for validating product and process is set. Resource planning for execution of a validation plan is done.
3. **Documentation:** This function controls the distribution and archiving of documents. Any change in a document is made by adopting the proper change control procedure. Approval of all types of documents.
4. **Assuring Quality of products**
5. **Quality improvement plans**

## Quality Assurance Certifications:

There are several certifications available in the industry to ensure that Organizations follow Standards Quality Processes. Customers make this as qualifying criteria while selecting a software vendor.

### **ISO 9000**

This standard was first established in 1987, and it is related to Quality Management Systems. This helps the organization ensure quality to their customers and other stakeholders. An organization who wishes to be certified as ISO 9000 is audited based on their functions, products, services and their processes. The main objective is to review and verify whether the organization is following the process as expected and check whether existing processes need improvement.

This certification helps -

* Increase the profit of the organization
* Improves Domestic and International trade
* Reduces waste and increase the productivity of the employees
* Provide Excellent customer satisfaction

## CMMI level

The **Capability Maturity Model Integrated (CMMI)** is a process improvement approach developed specially for software process improvement. It is based on the process maturity framework and used as a general aid in business processes in the Software Industry. This model is highly regarded and widely used in Software Development Organizations.

CMMI has 5 levels. An organization is certified at CMMI level 1 to 5 based on the maturity of their Quality Assurance Mechanisms.

* Level 1 - **Initial:** In this stage the quality environment is unstable. Simply, no processes have been followed or documented
* Level 2 - **Repeatable:** Some processes are followed which are repeatable. This level ensures processes are followed at the project level.
* Level 3 - **Defined:**Set of processes are defined and documented at the organizational level. Those defined processes are subject to some degree of improvement.
* Level 4 - **Managed:** This level uses process metrics and effectively controls the processes that are followed.
* Level 5 - **Optimizing:** This level focuses on the continuous improvements of the processes through learning &  innovation.

## Test Maturity Model (TMM):

This model assesses the maturity of processes in a Testing Environment. Even this model has 5 levels, defined below-

* Level 1 - **Initial**: There is no quality standard followed for testing processes and only ad-hoc methods are used at this level
* Level 2 - **Definition:** Defined process. Preparation of test strategy, plans, test cases are done.
* Level 3 - **Integration:** Testing is carried out throughout the software development lifecycle (SDLC) - which is nothing but integration with the development activities, E.g., V- Model.
* Level 4 -**Management and Measurement:** Review of requirements and designs takes place at this level and criteria has been set for each level of testing
* Level 5 - **Optimization:** Many preventive techniques are used for testing processes, and tool support (Automation) is used to improve the testing standards and processes.

**Conclusion:**

Quality Assurance is to check whether the product developed is fit for use. For that, Organization should have processes and standards to be followed which need to be improved on a periodic basis. It concentrates mainly on the quality of product/service that we are providing to the customers during or after implementation of software.

FMEA(Failure Mode and Effects Analysis), software HAZOP(Hazard and Operability Analysis), and software

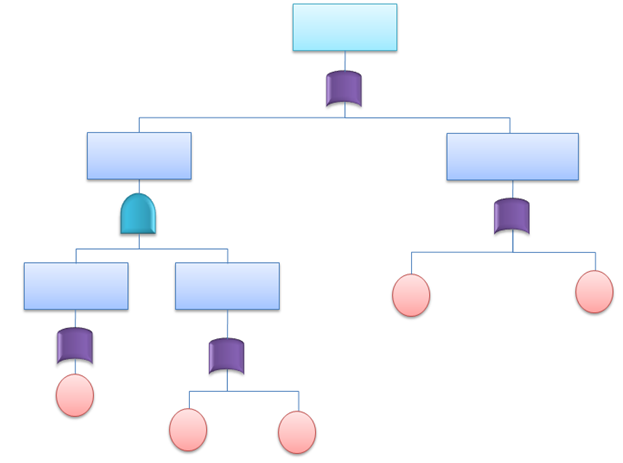
FTA(Fault Tree Analysis).

It was originally developed in the bell laboratories in 1962 by the H.A Watson. This is a top down approach which is a combination of events and logic gates to identify the top undesired event. This top and undesired event is the last event to occur. It does not make assumptions, it brings together every perspective of that can cause an event occurrence.

## What is Fault Tree Analysis

Fault tree analysis (FTA) is a graphical tool to explore the causes of system level failures. It uses boolean logic to combine a series of lower level events and it is basically a top-down approach to identify the component level failures (basic event) that cause the system level failure (top event) to occur. Fault tree analysis consists of two elements “events” and “logic gates” which connect the events to identify the cause of the top undesired event.

Fault tree analysis is an easier method than the [Failure Mode and Effects Analysis (FMEA)](https://sixsigmastudyguide.com/failure-mode-effects-analysis-fmea/)as it focuses on all possible system failures of an undesired top event. Whereas [FMEA](https://sixsigmastudyguide.com/failure-mode-effects-analysis-fmea/) conducts analysis to find all possible system failure modes irrespective of their severity.

FTA Diagram

## **History of Fault Tree Analysis**

Fault tree analysis is a top down approach that was originally developed in Bell laboratories by H Waston and A Mearns for the air force in the year 1962. This concept later adopted by Boeing and today it is widely used in aerospace, automobile, chemical, nuclear and software industries especially reliability and safety related events.

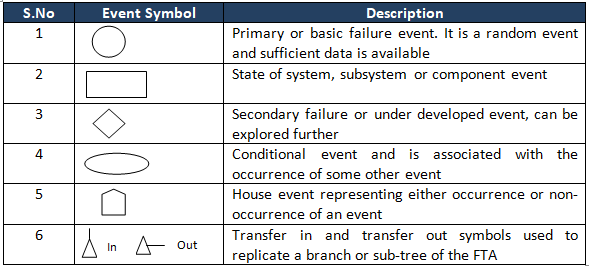
## When Would You Use FTA

Fault tree analysis can be used to perform for all types of system level risk assessment process. The purpose of FTA is to effectively identify cause(s) of system failure and mitigate the risks before it occurs. This is an invaluable tool for complex systems that visually displays the logical way of identifying the problem. Moreover system efficiency can be attained by this analysis. It can be implemented alone or complement to [Failure Mode and Effects Analysis (FMEA).](https://sixsigmastudyguide.com/failure-mode-effects-analysis-fmea/)

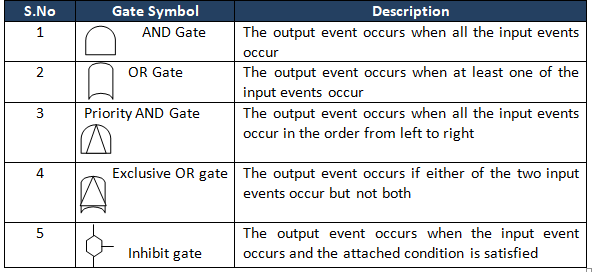
## FTA**Symbols**

Fault tree uses logical gates to perform the analysis. There are numerous FTA symbols exists, but these are broadly divided in to two categories, Event symbols and Gate symbols.

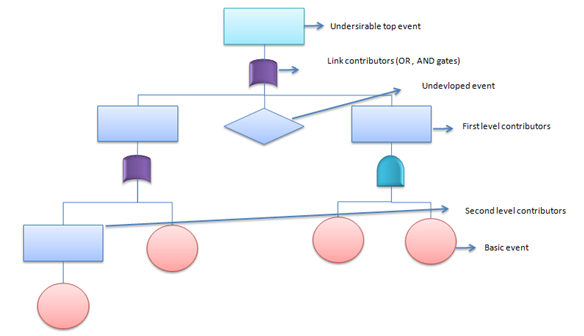
#### Event Symbols in FTA



#### Gate Symbols in FTA



## How do you do Fault Tree Analysis

* Define the primary failure to be analyzed in other words identify the undesirable top event
* Identify first level contributors which are just below the top level using the available technical information
* Link these contributors to top level event by using logical gates (AND, OR gates), and also see the relationship, so that it will help to identify the appropriate logical gate
* Identify the second level contributors and link to top by using logical gates.
* Identify minimal cut set
* Repeat the same steps till the basic causes
* Finally complete and evaluate the FTA
* Calculate probability of lowest level elements occurrence and also measure the probabilities from bottom up
* 

## **Minimal Cut Sets**

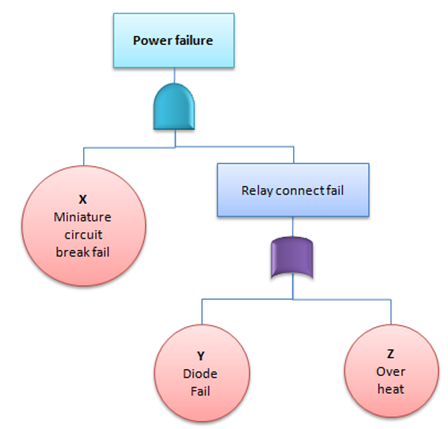
One of the important factors in qualitative analysis of fault tree is to identify minimal cut set. For instance complex and large fault tress have to use superior tools (algorithms for extraction) to get the minimal cut sets.

**Cut set:** A set of basic events that together cause the TOP undesirable event.

Ex: X, Y and Z (from the below picture)

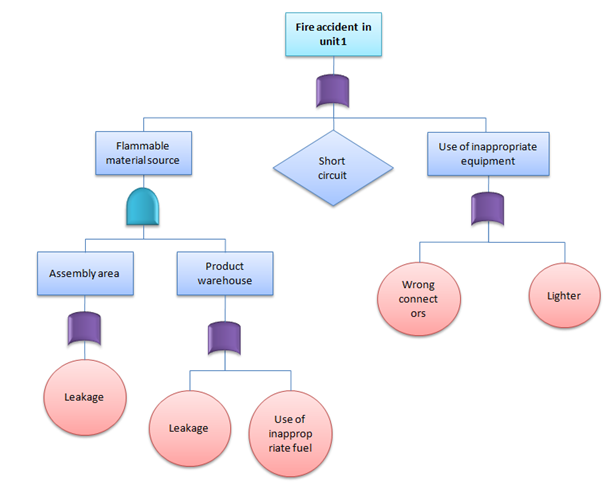
**Minimal cut set:** A cut set with minimal number of events that can still cause the TOP undesirable event. In other words the TOP undesirable event occurs if one or more minimal cut set occurs.

Ex: (X and Y); (X and Z) from the below picture



## Practical Example of FTA

A fire broke out at unit 1 of XYZ cable manufacturing company despite safety system in-place. General Manager was very concern about the accident and requests safety in-charge to evaluate the system. However as part of initial analysis of the existing system, safety team using FTA to identify the different causes for accident.



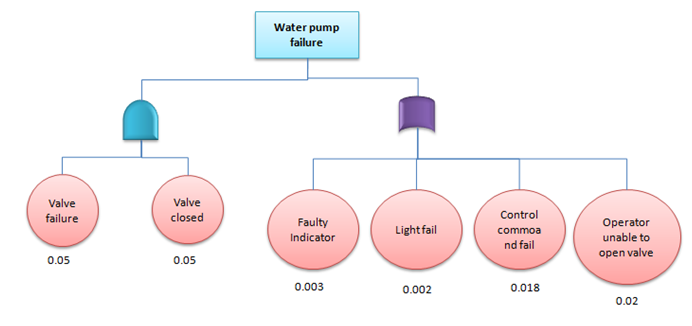
## **Quantitative Fault Tree Analysis**

Top undesirable event occurs if one or more of the minimal cut set occurs, hence the main target is to identify minimal cut sets. Moreover if all the minimal cuts are independent to each other, then we could compute the probability of top undesirable event by:

https://sixsigmastudyguide.com/wp-content/uploads/2020/02/f6.png

Where Pj is the failure probability of minimal cut set

**Example:**Find the probability of water pump failure from the below example



The water pump will fail because of value failure and value closed or fault indicator or light fail or control command fail or operator unable to open valve. Since OR gates add and AND gates multiply the probability of pump failure

μpumpfail=1-(0.05\*0.05)\*(1-0.003)\*(1-0.002)\*(1-0.018)\*(1-0.02) = 0.0448

Hence, the probability of water pump failure = 4.48%

## **Advantages of Fault tree** analysis

* Fault tree visually depict the analysis that will help team to work on cause of event in logical way that leads to failure
* Highlights the critical components related to system failure
* Provides an efficient method to analyze the system
* Unlike other analysis methods, human errors are also include in the analysis
* It helps to prioritize the action items to solve the problem
* Provides qualitative and quantitative analysis

## **Disadvantages of Fault tree** analysis

* Too many gates and events to be consider for large system analysis
* The basic disadvantage is that it examine only one top event
* Common cause failures are not always obvious
* Difficult to capture time related and other delay factors
* Needs experienced individuals to understand the logical gates

Difference between fault tree

Failure analysis mechanisms, including HAZOP, are required to relate a system representation to this underlying chain of causality. According to McDermid: “Failure analysis is primarily a synthetic exercise that determines ways in which the failure in the real world affects the behaviour of the system. In order to trace from faults to failures or *vice versa*, techniques such as Fault Tree Analysis (FTA) and Failure Modes and Effects Analysis (FMEA) require a representation of the dependencies between different components in the system so that it is possible to follow through the causal links between failure in a particular components to failure in the system as a whole.” [5] Later, Fenelon and McDermid suggest that “there is a more fundamental relationship: we believe that FTA and FMEA are both same underlying causal model of the propagation of failure (cause and effect) through a

System.

This simple model (figure 6) demonstrates the complementary approaches of FTA and FMEA. Specifically; FTA starts with a top event (system-level failure mode) and identifies a range of potential causes for that specific consequence. In contrast, FMEA is a bottom up analysis starting from a given component-level failure and working forwards to evaluate its effects.

Multiple

causes

Single

consequence

FTA

Multiple

causes

Single

consequence

FMEA

Multiple

causes

HAZOP

Multiple

consequences

HAZOP does not appear to fit into this simple model. The starting point for a HAZOP study is the deviation from the design intent. Once identified the HAZOP then aims to identify

potential causes and consequences of that deviation. Only when we extend our simple model

to include the immediate effect of an event are we then able to develop a clearer image of were the HAZOP study fits.

HAZOP stands for hazard and operability study. it is a process used to identify and access risk. It originated in the United Kingdom in 1960s and it has been used to examine traffic of failures. There are two aspects of HAZOP which are risk and consequence.

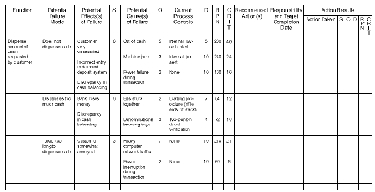
## WHEN TO USE FMEA

* When a process, product, or service is being designed or redesigned, after [quality function deployment (QFD)](https://asq.org/quality-resources/qfd-quality-function-deployment)
* When an existing process, product, or service is being applied in a new way
* Before developing control plans for a new or modified process
* When improvement goals are planned for an existing process, product, or service
* When analyzing failures of an existing process, product, or service
* Periodically throughout the life of the process, product, or service

## FMEA PROCEDURE

**Note:** This is a general procedure. Specific details may vary with standards of your organization or industry. Before undertaking an FMEA process, learn more about standards and specific methods in your organization and industry through other references and training.

1. Assemble a cross-functional [team](https://asq.org/quality-resources/teams) of people with diverse knowledge about the process, product or service, and customer needs. Functions often included are: design, manufacturing, quality, testing, reliability, maintenance, purchasing (and suppliers), sales, marketing (and customers), and customer service.
2. Identify the scope of the FMEA. Is it for concept, system, design, process, or service? What are the boundaries? How detailed should we be? Use [flowcharts](https://asq.org/quality-resources/flowchart) to identify the scope and to make sure every team member understands it in detail.
3. Fill in the identifying information at the top of your FMEA form. (Figure 1 shows a typical format.) The remaining steps ask for information that will go into the columns of the form.



**Figure 1: FMEA Example**

1. Identify the functions of your scope. Ask, "What is the purpose of this system, design, process, or service? What do our customers expect it to do?" Name it with a verb followed by a noun. Usually one will break the scope into separate subsystems, items, parts, assemblies, or process steps and identify the function of each.
2. For each function, identify all the ways failure could happen. These are potential failure modes. If necessary, go back and rewrite the function with more detail to be sure the failure modes show a loss of that function.
3. For each failure mode, identify all the consequences on the system, related systems, process, related processes, product, service, customer, or regulations. These are potential effects of failure. Ask, "What does the customer experience because of this failure? What happens when this failure occurs?"
4. Determine how serious each effect is. This is the severity rating, or S. Severity is usually rated on a scale from 1 to 10, where 1 is insignificant and 10 is catastrophic. If a failure mode has more than one effect, write on the FMEA table only the highest severity rating for that failure mode.
5. For each failure mode, determine all the potential root causes. Use tools classified as [cause analysis tools](https://asq.org/quality-resources/root-cause-analysis), as well as the best knowledge and experience of the team. List all possible causes for each failure mode on the FMEA form.
6. For each cause, determine the occurrence rating, or O. This rating estimates the probability of failure occurring for that reason during the lifetime of your scope. Occurrence is usually rated on a scale from 1 to 10, where 1 is extremely unlikely and 10 is inevitable. On the FMEA table, list the occurrence rating for each cause.
7. For each cause, identify current process controls. These are tests, procedures or mechanisms that you now have in place to keep failures from reaching the customer. These controls might prevent the cause from happening, reduce the likelihood that it will happen or detect failure after the cause has already happened but before the customer is affected.
8. For each control, determine the detection rating, or D. This rating estimates how well the controls can detect either the cause or its failure mode after they have happened but before the customer is affected. Detection is usually rated on a scale from 1 to 10, where 1 means the control is absolutely certain to detect the problem and 10 means the control is certain not to detect the problem (or no control exists). On the FMEA table, list the detection rating for each cause.
9. Optional for most industries: Ask, "Is this failure mode associated with a critical characteristic?" (Critical characteristics are measurements or indicators that reflect safety or compliance with government regulations and need special controls.) If so, a column labeled "Classification" receives a Y or N to show whether special controls are needed. Usually, critical characteristics have a severity of 9 or 10 and occurrence and detection ratings above 3.
10. Calculate the risk priority number, or RPN, which equals S × O × D. Also calculate Criticality by multiplying severity by occurrence, S × O. These numbers provide guidance for ranking potential failures in the order they should be addressed.
11. Identify recommended actions. These actions may be design or process changes to lower severity or occurrence. They may be additional controls to improve detection. Also note who is responsible for the actions and target completion dates.
12. As actions are completed, note results and the date on the FMEA form. Also, note new S, O, or D ratings and new RPNs.

Statistical quality control

Statistical quality control (SQC) is defined as the application of the 14 statistical and analytical tools (7-QC and 7-SUPP) to monitor process outputs (dependent variables). Statistical process control (SPC) is the application of the same 14 tools to control process inputs (independent variables). Although both terms are often used interchangeably, SQC includes acceptance sampling where SPC does not.

## THE 7 QUALITY CONTROL (7-QC) TOOLS

In 1974, [Dr. Kaoru Ishikawa](https://asq.org/about-asq/honorary-members/ishikawa) brought together a collection of process improvement tools in his text Guide to Quality Control. Known around the world as the [seven quality control (7-QC) tools](https://asq.org/quality-resources/seven-basic-quality-tools), they are:

1. [Cause-and-effect diagram](https://asq.org/quality-resources/fishbone) (also called Ishikawa diagram or fishbone diagram)
2. [Check sheet](https://asq.org/quality-resources/check-sheet)
3. [Control chart](https://asq.org/quality-resources/control-chart)
4. [Histogram](https://asq.org/quality-resources/histogram)
5. [Pareto chart](https://asq.org/quality-resources/pareto)
6. [Scatter diagram](https://asq.org/quality-resources/scatter-diagram)
7. [Stratification](https://asq.org/quality-resources/stratification)

## THE 7 SUPPLEMENTAL (7-SUPP) TOOLS

In addition to the basic 7-QC tools, there are also some additional statistical quality tools known as the seven supplemental (7-SUPP) tools:

1. [Data stratification](https://asq.org/quality-resources/stratification)
2. Defect maps
3. Events logs
4. [Process flowcharts](https://asq.org/quality-resources/flowchart)
5. Progress centers
6. Randomization
7. Sample size determination

FISHBONE DIAGRAM

[Quality Glossary Definition: Fishbone diagram](https://asq.org/quality-resources/quality-glossary/f)

Fishbone Diagram5

Also called: cause-and-effect diagram, Ishikawa diagram

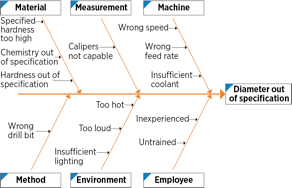
Variations: cause enumeration diagram, process fishbone, time-delay fishbone, CEDAC (cause-and-effect diagram with the addition of cards), desired-result fishbone, reverse fishbone diagram   
  
This [cause analysis tool](https://asq.org/quality-resources/root-cause-analysis/tools) is considered one of the [seven basic quality tools](https://asq.org/quality-resources/seven-basic-quality-tools). The fishbone diagram identifies many possible causes for an effect or problem. It can be used to structure a brainstorming session. It immediately sorts ideas into useful categories.

* [When to use a fishbone diagram](https://asq.org/quality-resources/fishbone#Use)
* [Fishbone diagram procedure](https://asq.org/quality-resources/fishbone#Procedure)
* [Fishbone diagram example](https://asq.org/quality-resources/fishbone#Example)
* [Create a fishbone diagram](https://asq.org/quality-resources/fishbone#Create)
* [Fishbone diagram resources](https://asq.org/quality-resources/fishbone#Resources)

WHEN TO USE A FISHBONE DIAGRAM

* When identifying possible causes for a problem
* When a team’s thinking tends to fall into ruts

FISHBONE DIAGRAM PROCEDURE



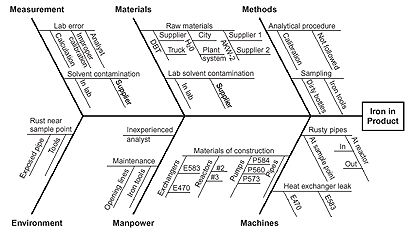
**Fishbone Diagram Example**

**Materials needed:** marking pens and flipchart or whiteboard.

1. Agree on a problem statement (effect). Write it at the center right of the flipchart or whiteboard. Draw a box around it and draw a horizontal arrow running to it.
2. Brainstorm the major categories of causes of the problem. If this is difficult use generic headings:
   * + Methods
     + Machines (equipment)
     + People (manpower)
     + Materials
     + Measurement
     + Environment
3. Write the categories of causes as branches from the main arrow.
4. Brainstorm all the possible causes of the problem. Ask "Why does this happen?" As each idea is given, the facilitator writes it as a branch from the appropriate category. Causes can be written in several places if they relate to several categories.
5. Again ask "Why does this happen?" about each cause. Write sub-causes branching off the causes. Continue to ask "Why?" and generate deeper levels of causes. Layers of branches indicate causal relationships.
6. When the group runs out of ideas, focus attention to places on the chart where ideas are few.

FISHBONE DIAGRAM EXAMPLE

This fishbone diagram was drawn by a manufacturing team to try to understand the source of periodic iron contamination. The team used the six generic headings to prompt ideas. Layers of branches show thorough thinking about the causes of the problem.



**Fishbone Diagram Example**

For example, under the heading "Machines," the idea "materials of construction" shows four kinds of equipment and then several specific machine numbers.

Note that some ideas appear in two different places. "Calibration" shows up under "Methods" as a factor in the analytical procedure, and also under "Measurement" as a cause of lab error. "Iron tools" can be considered a "Methods" problem when taking samples or a "Manpower" problem with maintenance personnel.

CHECK SHEET

[Quality Glossary Definition: Check sheet](https://asq.org/quality-resources/quality-glossary/c)

Also called: defect concentration diagram

A check sheet is a structured, prepared form for collecting and analyzing data. This is a generic [data collection and analysis tool](https://asq.org/quality-resources/data-collection-analysis-tools) that can be adapted for a wide variety of purposes and is considered one of the [seven basic quality tools](https://asq.org/quality-resources/seven-basic-quality-tools).

WHEN TO USE A CHECK SHEET

* When data can be observed and collected repeatedly by the same person or at the same location
* When collecting data on the frequency or patterns of events, problems, defects, defect location, defect causes, or similar issues
* When collecting data from a production process

**Looking for more quality tools?**

Try Plan-Do-Study-Act (PDSA) Plus QTools™ Training:

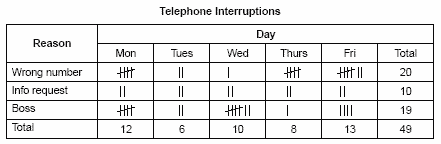
* [Check Sheet](https://asq.org/training/asq-quality-tools---check-sheet-csasq)
* [QToolsTM Bundle](https://asq.org/training/qtools-suite-qtbasq)
* [Plan-Do-Study-Act plus QToolsTM](https://asq.org/training/asq-quality-tools-plan-do-study-act-plus-qtools-pdsaqtasq)

CHECK SHEET PROCEDURE

1. Decide what event or problem will be observed. Develop operational definitions.
2. Decide when data will be collected and for how long.
3. Design the form. Set it up so that data can be recorded simply by making check marks or X's or similar symbols and so that data do not have to be recopied for analysis.
4. Label all spaces on the form.
5. Test the check sheet for a short trial period to be sure it collects the appropriate data and is easy to use.
6. Each time the targeted event or problem occurs, record data on the check sheet.

CHECK SHEET EXAMPLE

The figure below shows a check sheet used to collect data on telephone interruptions. The tick marks were added as data was collected over several weeks.



# CONTROL CHART

**Looking for more quality tools?**

Try Plan-Do-Study-Act (PDSA) Plus QTools™ Training:

* [Run Chart](https://asq.org/training/asq-quality-tools---run-chart-rcasq)
* [Flowchart](https://asq.org/training/asq-quality-tools---flowchart-fcasq)
* [QToolsTM Bundle](https://asq.org/training/qtools-suite-qtbasq)
* [Plan-Do-Study-Act plus QToolsTM](https://asq.org/training/asq-quality-tools-plan-do-study-act-plus-qtools-pdsaqtasq)

[Quality Glossary Definition: Control chart](https://asq.org/quality-resources/quality-glossary/c#cc)

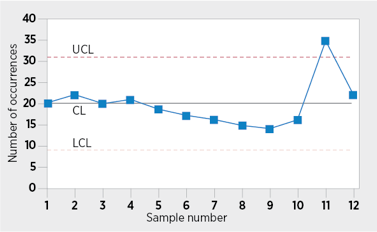
Also called: Shewhart chart, statistical process control chart

The control chart is a graph used to study how a process changes over time. Data are plotted in time order. A control chart always has a central line for the average, an upper line for the upper

control limit, and a lower line for the lower control limit. These lines are determined from historical data. By comparing current data to these lines, you can draw conclusions about whether the process variation is consistent (in control) or is unpredictable (out of control, affected by special causes of variation). This versatile [data collection and analysis tool](https://asq.org/quality-resources/data-collection-analysis-tools) can be used by a variety of industries and is considered one of the [seven basic quality tools](https://asq.org/quality-resources/seven-basic-quality-tools).

Control charts for variable data are used in pairs. The top chart monitors the average, or the centering of the distribution of data from the process. The bottom chart monitors the range, or the width of the distribution. If your data were shots in target practice, the average is where the shots are clustering, and the range is how tightly they are clustered. Control charts for attribute data are used singly.

* [When to use a control chart](https://asq.org/quality-resources/control-chart#Use)
* [Basic procedure](https://asq.org/quality-resources/control-chart#Procedure)
* [Create a control chart](https://asq.org/quality-resources/control-chart#Create)
* [Control chart resources](https://asq.org/quality-resources/control-chart#Resources)



**Control Chart Example**

## WHEN TO USE A CONTROL CHART

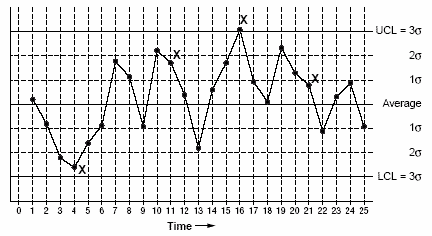
* When controlling ongoing processes by finding and correcting problems as they occur
* When predicting the expected range of outcomes from a process
* When determining whether a process is stable (in statistical control)
* When analyzing patterns of process variation from special causes (non-routine events) or common causes (built into the process)
* When determining whether your quality improvement project should aim to prevent specific problems or to make fundamental changes to the process

## BASIC PROCEDURE

1. Choose the appropriate control chart for your data.
2. Determine the appropriate time period for collecting and plotting data.
3. Collect data, construct your chart and analyze the data.
4. Look for "out-of-control signals" on the control chart. When one is identified, mark it on the chart and investigate the cause. Document how you investigated, what you learned, the cause and how it was corrected.

### Out-of-control signals

* + A single point outside the control limits. In Figure 1, point sixteen is above the UCL (upper control limit).
  + Two out of three successive points are on the same side of the centerline and farther than 2 σ from it. In Figure 1, point 4 sends that signal.
  + Four out of five successive points are on the same side of the centerline and farther than 1 σ from it. In Figure 1, point 11 sends that signal.
  + A run of eight in a row are on the same side of the centerline. Or 10 out of 11, 12 out of 14, or 16 out of 20. In Figure 1, point 21 is eighth in a row above the centerline.
  + Obvious consistent or persistent patterns that suggest something unusual about your data and your process.



**Figure 1 Control Chart: Out-of-Control Signals**

1. Continue to plot data as they are generated. As each new data point is plotted, check for new out-of-control signals.
2. When you start a new control chart, the process may be out of control. If so, the control limits calculated from the first 20 points are conditional limits. When you have at least 20 sequential points from a period when the process is operating in control, recalculate control limits.

# WHAT IS A HISTOGRAM?

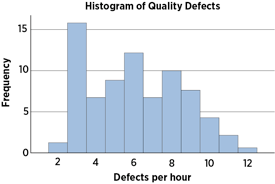
[Quality Glossary Definition: Histogram](https://asq.org/quality-resources/quality-glossary/h)

A frequency distribution shows how often each different value in a set of data occurs. A histogram is the most commonly used graph to show frequency distributions. It looks very much like a bar chart, but there are important differences between them. This helpful [data collection and analysis tool](https://asq.org/quality-resources/data-collection-analysis-tools) is considered one of the [seven basic quality tools](https://asq.org/quality-resources/seven-basic-quality-tools).

## WHEN TO USE A HISTOGRAM

Use a histogram when:

* The data are numerical
* You want to see the shape of the data’s distribution, especially when determining whether the output of a process is distributed approximately normally
* Analyzing whether a process can meet the customer’s requirements
* Analyzing what the output from a supplier’s process looks like
* Seeing whether a process change has occurred from one time period to another
* Determining whether the outputs of two or more processes are different
* You wish to communicate the distribution of data quickly and easily to others



**Histogram Example**

## HOW TO CREATE A HISTOGRAM

1. Collect at least 50 consecutive data points from a process.
2. Use a [histogram worksheet](https://asq.org/quality-resources/histogram#Worksheet) to set up the histogram. It will help you determine the number of bars, the range of numbers that go into each bar, and the labels for the bar edges. After calculating W in Step 2 of the worksheet, use your judgment to adjust it to a convenient number. For example, you might decide to round 0.9 to an even 1.0. The value for W must not have more decimal places than the numbers you will be graphing.
3. Draw x- and y-axes on graph paper. Mark and label the y-axis for counting data values. Mark and label the x-axis with the L values from the worksheet. The spaces between these numbers will be the bars of the histogram. Do not allow for spaces between bars.
4. For each data point, mark off one count above the appropriate bar with an X or by shading that portion of the bar.

## HISTOGRAM ANALYSIS

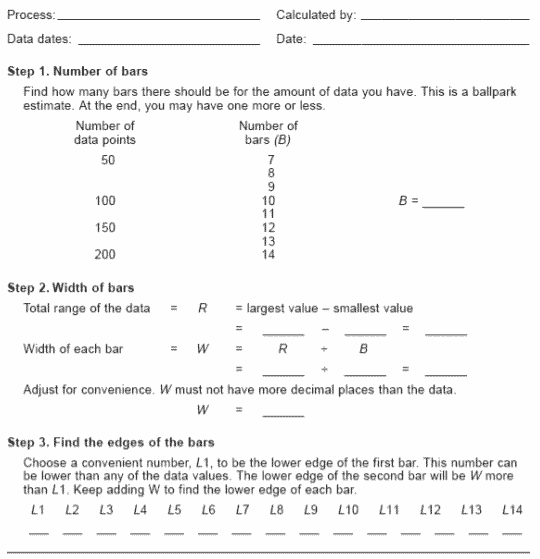
* Before drawing any conclusions from your histogram, be sure that the process was operating normally during the time period being studied. If any unusual events affected the process during the time period of the histogram, your analysis of the histogram shape likely cannot be generalized to all time periods.
* Analyze the meaning of your histogram's shape. Typical histogram [shapes and what they mean](https://asq.org/quality-resources/histogram#Shapes) are covered below.

## HISTOGRAM TOOLS & TEMPLATES

[Histogram template](https://asq.org/-/media/public/learn-about-quality/data-collection-analysis-tools/data-point-histogram.xls) (Excel) Analyze the frequency distribution of up to 200 data points using this simple, but powerful, histogram generating tool.

[Check sheet template](https://asq.org/-/media/public/learn-about-quality/data-collection-analysis-tools/check-sheet-histogram.xls) (Excel) Analyze the number of defects for each day of the week. Start by tracking the defects on the check sheet. The tool will create a histogram using the data you enter.

## HISTOGRAM WORKSHEET EXAMPLE

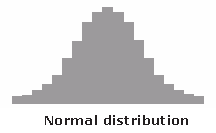


## TYPICAL HISTOGRAM SHAPES AND WHAT THEY MEAN

### Normal Distribution

A common pattern is the bell-shaped curve known as the "normal distribution." In a normal or "typical" distribution, points are as likely to occur on one side of the average as on the other. Note that other distributions look similar to the normal distribution. Statistical calculations must be used to prove a normal distribution.

It's important to note that "normal" refers to the typical distribution for a particular process. For example, many processes have a natural limit on one side and will produce skewed distributions. This is normal—meaning typical—for those processes, even if the distribution isn’t considered "normal."



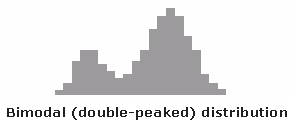
### Skewed Distribution

The skewed distribution is asymmetrical because a natural limit prevents outcomes on one side. The distribution’s peak is off center toward the limit and a tail stretches away from it. For example, a distribution of analyses of a very pure product would be skewed, because the product cannot be more than 100 percent pure. Other examples of natural limits are holes that cannot be smaller than the diameter of the drill bit or call-handling times that cannot be less than zero. These distributions are called right- or left-skewed according to the direction of the tail.



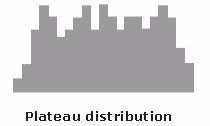
### Double-Peaked or Bimodal

The bimodal distribution looks like the back of a two-humped camel. The outcomes of two processes with different distributions are combined in one set of data. For example, a distribution of production data from a two-shift operation might be bimodal, if each shift produces a different distribution of results. Stratification often reveals this problem.



### Plateau or Multimodal Distribution

The plateau might be called a “multimodal distribution.” Several processes with normal distributions are combined. Because there are many peaks close together, the top of the distribution resembles a plateau.



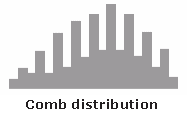
### Edge Peak Distribution

The edge peak distribution looks like the normal distribution except that it has a large peak at one tail. Usually this is caused by faulty construction of the histogram, with data lumped together into a group labeled “greater than.”



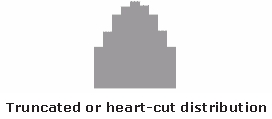
### Comb Distribution

In a comb distribution, the bars are alternately tall and short. This distribution often results from rounded-off data and/or an incorrectly constructed histogram. For example, temperature data rounded off to the nearest 0.2 degree would show a comb shape if the bar width for the histogram were 0.1 degree.



### Truncated or Heart-Cut Distribution

The truncated distribution looks like a normal distribution with the tails cut off. The supplier might be producing a normal distribution of material and then relying on inspection to separate what is within specification limits from what is out of spec. The resulting shipments to the customer from inside the specifications are the heart cut.



### Dog Food Distribution

The dog food distribution is missing something—results near the average. If a customer receives this kind of distribution, someone else is receiving a heart cut and the customer is left with the “dog food,” the odds and ends left over after the master’s meal. Even though what the customer receives is within specifications, the product falls into two clusters: one near the upper specification limit and one near the lower specification limit. This variation often causes problems in the customer’s process.



WHAT IS A PARETO CHART?

[Quality Glossary Definition: Pareto chart](https://asq.org/quality-resources/quality-glossary/p)

Also called: Pareto diagram, Pareto analysis

Variations: weighted Pareto chart, comparative Pareto charts

A Pareto chart is a bar graph. The lengths of the bars represent frequency or cost (time or money), and are arranged with longest bars on the left and the shortest to the right. In this way the chart visually depicts which situations are more significant. This [cause analysis tool](https://asq.org/quality-resources/root-cause-analysis/tools) is considered one of the [seven basic quality tools](https://asq.org/quality-resources/seven-basic-quality-tools).

* [When to use a Pareto chart](https://asq.org/quality-resources/pareto#Use)
* [Pareto chart procedure](https://asq.org/quality-resources/pareto#Procedure)
* [Pareto chart examples](https://asq.org/quality-resources/pareto#Examples)
* [Create a Pareto chart](https://asq.org/quality-resources/pareto#Create)
* [Pareto chart resources](https://asq.org/quality-resources/pareto#Resources)

WHEN TO USE A PARETO CHART

* When analyzing data about the frequency of problems or causes in a process
* When there are many problems or causes and you want to focus on the most significant
* When analyzing broad causes by looking at their specific components
* When communicating with others about your data

PARETO CHART PROCEDURE

1. Decide what categories you will use to group items.
2. Decide what measurement is appropriate. Common measurements are frequency, quantity, cost and time.
3. Decide what period of time the Pareto chart will cover: One work cycle? One full day? A week?
4. Collect the data, recording the category each time, or assemble data that already exist.
5. Subtotal the measurements for each category.
6. Determine the appropriate scale for the measurements you have collected. The maximum value will be the largest subtotal from step 5. (If you will do optional steps 8 and 9 below, the maximum value will be the sum of all subtotals from step 5.) Mark the scale on the left side of the chart.
7. Construct and label bars for each category. Place the tallest at the far left, then the next tallest to its right, and so on. If there are many categories with small measurements, they can be grouped as “other.”

**Note:** Steps 8 and 9 are optional but are useful for analysis and communication.

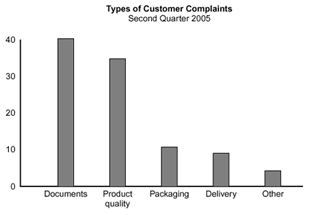
1. Calculate the percentage for each category: the subtotal for that category divided by the total for all categories. Draw a right vertical axis and label it with percentages. Be sure the two scales match. For example, the left measurement that corresponds to one-half should be exactly opposite 50% on the right scale.
2. Calculate and draw cumulative sums: add the subtotals for the first and second categories, and place a dot above the second bar indicating that sum. To that sum add the subtotal for the third category, and place a dot above the third bar for that new sum. Continue the process for all the bars. Connect the dots, starting at the top of the first bar. The last dot should reach 100% on the right scale.

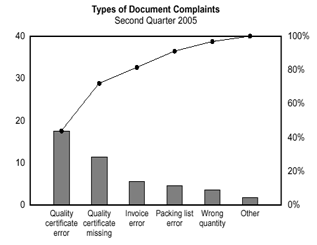
PARETO CHART EXAMPLES

Figure 1 shows how many customer complaints were received in each of five categories.

Figure 2 takes the largest category, "documents," from Figure 1, breaks it down into six categories of document-related complaints, and shows cumulative values.

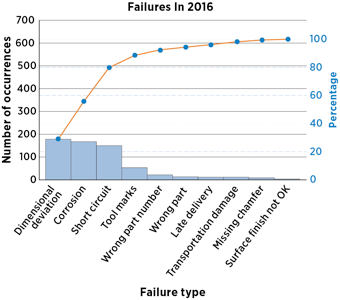
If all complaints cause equal distress to the customer, working on eliminating document-related complaints would have the most impact, and of those, working on quality certificates should be most fruitful.

  
**Figure 1: Pareto Chart, Customer Complaints**

  
**Figure 2: Pareto Chart, Document Complaints**

CREATE A PARETO CHART

Use the [Pareto chart template](https://asq.org/-/media/public/learn-about-quality/data-collection-analysis-tools/data-analysis.xls?la=en) (Excel) to create a Pareto chart and analyze the occurrences of up to 10 defects by entering the defects on the check sheet.



**Pareto Chart Template Example**

# WHAT IS A SCATTER DIAGRAM?

[Quality Glossary Definition: Scatter diagram](https://asq.org/quality-resources/quality-glossary/s)

Also called: scatter plot, X-Y graph

The scatter diagram graphs pairs of numerical data, with one variable on each axis, to look for a relationship between them. If the variables are correlated, the points will fall along a line or curve. The better the correlation, the tighter the points will hug the line. This [cause analysis tool](https://asq.org/quality-resources/root-cause-analysis/tools)is considered one of the [seven basic quality tools](https://asq.org/quality-resources/seven-basic-quality-tools).

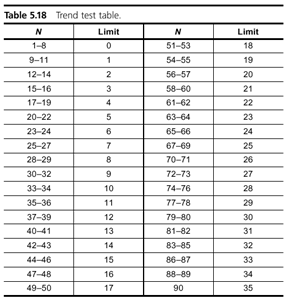
* [When to use a scatter diagram](https://asq.org/quality-resources/scatter-diagram#Use)
* [Scatter diagram procedure](https://asq.org/quality-resources/scatter-diagram#Procedure)
* [Scatter diagram example](https://asq.org/quality-resources/scatter-diagram#Example)
* [Scatter diagram considerations](https://asq.org/quality-resources/scatter-diagram#Considerations)
* [Scatter diagram resources](https://asq.org/quality-resources/scatter-diagram#Resources)

## WHEN TO USE A SCATTER DIAGRAM

* When you have paired numerical data
* When your dependent variable may have multiple values for each value of your independent variable
* When trying to determine whether the two variables are related, such as:
  + When trying to identify potential [root causes](https://asq.org/quality-resources/root-cause-analysis) of problems
  + After [brainstorming](https://asq.org/quality-resources/brainstorming) causes and effects using a [fishbone diagram](https://asq.org/quality-resources/fishbone) to determine objectively whether a particular cause and effect are related
  + When determining whether two effects that appear to be related both occur with the same cause
  + When testing for autocorrelation before constructing a [control chart](https://asq.org/quality-resources/control-chart)

## SCATTER DIAGRAM PROCEDURE

1. Collect pairs of data where a relationship is suspected.
2. Draw a graph with the independent variable on the horizontal axis and the dependent variable on the vertical axis. For each pair of data, put a dot or a symbol where the x-axis value intersects the y-axis value. (If two dots fall together, put them side by side, touching, so that you can see both.)
3. Look at the pattern of points to see if a relationship is obvious. If the data clearly form a line or a curve, you may stop because variables are correlated. You may wish to use regression or correlation analysis now. Otherwise, complete steps 4 through 7.
4. Divide points on the graph into four quadrants. If there are X points on the graph:  
   * Count X/2 points from top to bottom and draw a horizontal line.
   * Count X/2 points from left to right and draw a vertical line.
   * If number of points is odd, draw the line through the middle point.
5. Count the points in each quadrant. Do not count points on a line.
6. Add the diagonally opposite quadrants. Find the smaller sum and the total of points in all quadrants.  
   A = points in upper left + points in lower right  
   B = points in upper right + points in lower left  
   Q = the smaller of A and B  
   N = A + B
7. Look up the limit for N on the trend test table.  
   * If Q is less than the limit, the two variables are related.
   * If Q is greater than or equal to the limit, the pattern could have occurred from random chance.



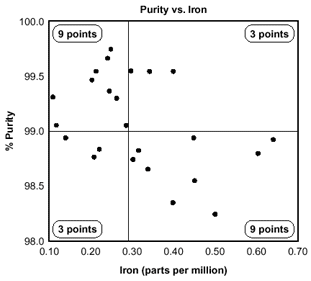
## SCATTER DIAGRAM EXAMPLE

The ZZ-400 manufacturing team suspects a relationship between product purity (percent purity) and the amount of iron (measured in parts per million or ppm). Purity and iron are plotted against each other as a scatter diagram, as shown in the figure below.

There are 24 data points. Median lines are drawn so that 12 points fall on each side for both percent purity and ppm iron.

To test for a relationship, they calculate:  
A = points in upper left + points in lower right = 9 + 9 = 18  
B = points in upper right + points in lower left = 3 + 3 = 6  
Q = the smaller of A and B = the smaller of 18 and 6 = 6  
N = A + B = 18 + 6 = 24

Then they look up the limit for N on the trend test table. For N = 24, the limit is 6.  
Q is equal to the limit. Therefore, the pattern could have occurred from random chance, and no relationship is demonstrated.

  
**Scatter Diagram Example**

### Additional Scatter Diagram Examples

Below are some examples of situations in which might you use a scatter diagram:

* Variable A is the temperature of a reaction after 15 minutes. Variable B measures the color of the product. You suspect higher temperature makes the product darker. Plot temperature and color on a scatter diagram.
* Variable A is the number of employees trained on new software, and variable B is the number of calls to the computer help line. You suspect that more training reduces the number of calls. Plot number of people trained versus number of calls.
* To test for autocorrelation of a measurement being monitored on a control chart, plot this pair of variables: Variable A is the measurement at a given time. Variable B is the same measurement, but at the previous time. If the scatter diagram shows correlation, do another diagram where variable B is the measurement two times previously. Keep increasing the separation between the two times until the scatter diagram shows no correlation.

## SCATTER DIAGRAM CONSIDERATIONS

* Even if the scatter diagram shows a relationship, do not assume that one variable caused the other. Both may be influenced by a third variable.
* When the data are plotted, the more the diagram resembles a straight line, the stronger the relationship.
* If a line is not clear, statistics (N and Q) determine whether there is reasonable certainty that a relationship exists. If the statistics say that no relationship exists, the pattern could have occurred by random chance.
* If the scatter diagram shows no relationship between the variables, consider whether the data might be stratified.
* If the diagram shows no relationship, consider whether the independent (x-axis) variable has been varied widely. Sometimes a relationship is not apparent because the data do not cover a wide enough range.

WHAT IS STRATIFICATION?

[Quality Glossary Definition: Stratification](https://asq.org/quality-resources/quality-glossary/s)

Stratification is defined as the act of sorting data, people, and objects into distinct groups or layers. It is a technique used in combination with other data analysis tools. When data from a variety of sources or categories have been lumped together, the meaning of the data can be difficult to see. This [data collection and analysis technique](https://asq.org/quality-resources/data-collection-analysis-tools) separates the data so that patterns can be seen and is considered one of the [seven basic quality tools](https://asq.org/quality-resources/seven-basic-quality-tools).

WHEN TO USE STRATIFICATION

* Before collecting data
* When data come from several sources or conditions, such as shifts, days of the week, suppliers, or population groups
* When data analysis may require separating different sources or conditions

Here are examples of different sources that might require data to be stratified:

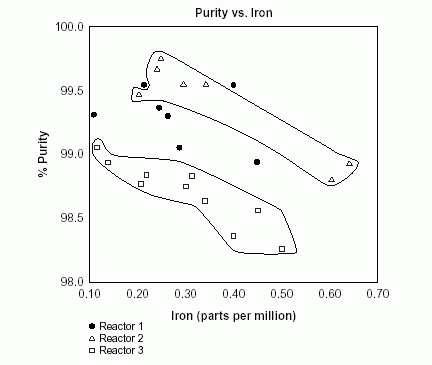
* Equipment
* Shifts
* Departments
* Materials
* Suppliers
* Day of the week
* Time of day
* Products

STRATIFICATION PROCEDURE

1. Before collecting data, consider which information about the sources of the data might have an effect on the results. Set up the data collection so that you collect that information as well.
2. When plotting or graphing the collected data on a [scatter diagram](https://asq.org/quality-resources/scatter-diagram), [control chart](https://asq.org/quality-resources/control-chart), [histogram](https://asq.org/quality-resources/histogram), or other analysis tool, use different marks or colors to distinguish data from various sources. Data that are distinguished in this way are said to be "stratified."
3. Analyze the subsets of stratified data separately. For example, on a scatter diagram where data are stratified into data from source 1 and data from source 2, draw quadrants, count points, and determine the critical value only for the data from source 1, then only for the data from source 2.

STRATIFICATION EXAMPLE

The ZZ-400 manufacturing team drew a scatter diagram to test whether product purity and iron contamination were related, but the plot did not demonstrate a relationship. Then a team member realized that the data came from three different reactors. The team member redrew the diagram, using a different symbol for each reactor’s data (Figure 1).

  
**Figure 1: Stratification Diagram**

Now patterns can be seen. The data from reactor 2 and reactor 3 are circled. Even without doing any calculations, it is clear that for those two reactors, purity decreases as iron increases. However, the data from reactor 1, the solid dots that are not circled, do not show that relationship. Something is different about reactor 1.