



# CORE MODULE 2

WEB DESIGN AND DEVELOPMENT

2022

---

## Table of Contents

Introduction to the internet, browsing, emailing.....	24
Overview of the Internet .....	24
Introduction to HTML .....	55
What is HTML?.....	55
Structure of an HTML document.....	56
HTML Tags and Attributes.....	57
HTML Colors.....	64
LINKING.....	68
Tables.....	70
Lists.....	76
Audio and Videos .....	81
HTML Forms and Input.....	83
HTML5 .....	89
Introduction to HTML5 .....	89
Page Layout Semantic Elements.....	90
HTML5 Web Forms .....	94
Different editors used for Web Page Developing .....	99
HTML Editors.....	99
Applications of HTML .....	121
How is its industry using HTML? .....	121
What is a Static Website? .....	121
TOP 10 USES OF HTML.....	130
Cascaded Style Sheet (CSS) .....	134

---

Introduction to CSS.....	134
Limitations of CSS.....	137
Advantages of CSS .....	137
CSS Syntax .....	138
Three ways to integrate CSS.....	139
Merits and demerits of - external Style Sheets, Embedded Style Sheets.....	141
CSS Units and Values.....	142
Absolute Lengths .....	143
Relative Lengths .....	143
CSS Styling Text.....	144
Styling Box .....	156
CSS Box Model .....	156
Website Layout .....	158
Introduction to JavaScript.....	177
What is JavaScript?.....	177
Why JavaScript? .....	178
What is JavaScript Used For? .....	178
Features of JavaScript .....	179
Application of JavaScript.....	179
JavaScript Example.....	179
JavaScript Syntax.....	182
JavaScript Variables.....	182
<b>What is Variable? .....</b>	182
<b>Declaring Multiple Variables at Once.....</b>	183

---

<b>Naming Conventions for JavaScript Variables.....</b>	183
JavaScript Datatype .....	184
<b>Primitive Data Types .....</b>	185
<b>Non-primitive data types .....</b>	186
JavaScript Operators .....	186
Javascript Condition.....	189
JavaScript If-else.....	189
JavaScript Switch.....	194
Loop Control statement.....	196
for loop.....	196
While loop.....	197
do while loop.....	198
for in loop.....	199
JavaScript Array.....	200
JavaScript Array Methods.....	202
JavaScript Object .....	204
JavaScript Function.....	209
User-defined functions.....	210
Declaring a Function.....	210
Calling a Function.....	210
Function Parameters.....	212
Function Return .....	213
Benefits of Using a Function .....	215
Function Expressions .....	215

---

Introduction to JavaScript Event.....	216
What is an Event? .....	216
What is Event handling? .....	216
Exceptional handling .....	220
How to handle exceptions in JavaScript .....	221
Throw statements - The throw statement is to raise your built-in exceptions. ....	223
Try catch statements - The try clause has the main code that may generate exceptions. If an exception is raised, the catch clause gets executed.....	223
Try catch finally statements - The finally statement is the last block to be executed. It executes after try and catch clauses.....	224
The Browser Object Model (BOM).....	225
JavaScript setTimeout() method .....	234
JavaScript setInterval() method .....	236
JavaScript Date .....	237
What is Ajax?.....	240
Understanding How Ajax Works.....	241
Performing an Ajax GET Request .....	242
Performing an Ajax POST Request .....	245
JavaScript JSON .....	249
Overview of Bootstrap .....	252
Advantages of Bootstrap.....	252
Saving time .....	253
Easy to use .....	253
Responsive Design.....	253
Cross Browser Compatible .....	253

---

Open Source.....	253
Customization .....	253
Bootstrap Container .....	254
Fixed Container.....	255
Fluid Container .....	255
How to use Bootstrap 5 on a webpage:.....	255
Bootstrap 5 CDN.....	255
MaxCDN:.....	256
Downloading Bootstrap 5 .....	256
Bootstrap Components.....	256
List of components:.....	257
Jumbotron .....	257
Alerts.....	257
Buttons .....	257
Button group .....	257
Badge .....	257
Progress Bar .....	257
Spinner .....	257
Scrollspy .....	257
List group.....	258
Card.....	258
Dropdown .....	258
Navs .....	258
Navbar.....	258

---

Forms .....	258
Input groups .....	258
Toast.....	258
Tooltip.....	258
Popovers.....	259
Collapse.....	259
Modal .....	259
Pagination .....	259
Media Object.....	259
Advance Components of Bootstrap .....	260
Accessibility.....	260
Breadcrumbs .....	260
Calendar .....	260
Carousel .....	261
Checkbox .....	261
Color Picker .....	262
Combobox.....	262
Contact Form .....	262
Datepicker .....	263
Dialog Boxes & Alerts.....	264
File Upload.....	266
Form Validation .....	266
HTML Tables.....	268
Image Gallery.....	269

---

In-Place Editing .....	269
Layout Grid.....	270
Magnify.....	270
Modal Windows.....	271
Navigation.....	272
Pagination .....	272
Progress Bars .....	272
Ratings .....	273
Social Buttons .....	273
Tabs.....	274
Bootstrap 5 Utilities .....	274
Background .....	274
Borders.....	275
Color .....	277
Display .....	278
Flex .....	280
Interactions.....	280
Overflow .....	281
Position .....	281
Box Shadow .....	282
Sizing .....	283
Spacing.....	284
Text .....	285
Introduction to jQuery.....	287

---

What is jQuery? .....	287
Basic syntax for any jQuery function is:.....	287
Why jQuery?.....	288
Advantages:.....	288
Disadvantages: .....	288
Adding jQuery to Your Web Pages .....	288
Downloading jQuery.....	289
jQuery CDN .....	289
jQuery Syntax .....	290
The Document Ready Event .....	290
jQuery Selectors.....	291
The element Selector.....	291
The #id Selector .....	293
The .class Selector .....	294
jQuery Attributes .....	296
Get Attribute Value.....	297
Set Attribute Value .....	298
Applying Styles.....	300
Attribute Methods .....	301
jQuery - DOM Traversing .....	302
Find Elements by Index .....	302
Filtering out Elements.....	304
Locating Descendant Elements.....	306
jQuery DOM Filter Methods .....	307

---

jQuery DOM Traversing Methods.....	308
jQuery - CSS Selectors Methods .....	310
Apply CSS Properties .....	310
Apply Multiple CSS Properties .....	310
Setting Element Width & Height.....	311
jQuery CSS Methods .....	312
jQuery Effects .....	314
jQuery hide() and show() .....	314
jQuery toggle().....	315
jQuery Fade.....	316
jQuery slide.....	317
jQuery Animations.....	318
Animation with callback .....	319
Purpose of database systems .....	323
What is Database? .....	323
What is DBMS? .....	324
What are the types of DBMS?.....	326
Centralized Database .....	326
Distributed Database.....	327
Personal Database .....	327
End User Database .....	327
Commercial Database .....	328
NoSQL Database .....	328
Operational Database .....	328

---

Relational Databases .....	328
Cloud Databases.....	329
Object-Oriented Databases .....	330
Graph Databases .....	331
DBMS vs. File System.....	332
Data Abstraction .....	333
Database Users.....	334
Naive Users .....	334
Application programmers.....	335
Sophisticated users.....	335
DBA [Database Administrators] .....	335
Data Independence (Logical & Physical) .....	336
Examples of changes under Physical Data Independence .....	338
Examples of changes under Logical Data Independence .....	339
Instance & Schemes.....	341
Differences Between Schema and Instance.....	342
Sub-Schema .....	342
Three Layered of Architecture in DBMS .....	343
Types of DBMS Architecture.....	343
1- Tier Architecture .....	344
2- Tier Architecture .....	344
3- Tier Architecture .....	345
Different Levels of Abstraction .....	346
The level of Abstraction is as follows.....	346

---

In the above diagram:.....	346
Physical level or Internal Level.....	347
Logical Level or Conceptual Level.....	347
View Level or External Level .....	347
Data Modelling .....	348
Relational Data Model .....	348
Entity-Relationship Data Model .....	349
Object-based Data Model.....	349
Semistructured Data Model.....	349
Hierarchical Model .....	349
Network Model .....	350
Flat Data Model.....	351
Record base Data Model .....	351
ER modeling .....	352
Component of ER Diagram .....	353
Logical Data Model .....	354
Entity Sets .....	356
Definition of Strong Entity.....	356
Definition of Weak Entity .....	358
Relationship Sets.....	360
One-to-One Relationship .....	360
One-to-many relationship.....	360
Many-to-one relationship .....	361
Many-to-many relationship.....	361

---

Concept of Attributes .....	362
Introduction.....	362
Types of Attributes .....	363
Simple Attributes .....	364
Single Valued Attributes .....	365
Key Attributes .....	365
Derived Attributes.....	365
Multivalued Attributes .....	365
Example 1 – Student Book Relation.....	366
Example 2 – Product Order System.....	367
Example 3 –Employee Management .....	368
Concept of Relationship.....	369
Introduction.....	369
Symbol .....	369
Types of Keys.....	370
Keys .....	370
Primary Key .....	370
Foreign Key .....	371
Candidate Key.....	372
Super Key.....	373
Compound Key .....	373
Alternate Key .....	373
Unique Key .....	374
Mapping Constraints .....	374

---

Introduction.....	374
Types of Mapping Constraints.....	375
One-to-One .....	375
One-to-many .....	376
Many-to One .....	376
Many-to-Many .....	376
Example of Employee Management System .....	377
Example of Movie Ticket Management.....	378
Entity Relationship Diagram.....	379
Introduction.....	379
Components of ER Diagram.....	379
Entity.....	380
a. Weak Entity .....	380
Examples of entities:.....	381
Attribute .....	382
a. Key Attribute .....	382
b. Composite Attribute .....	383
c. Multivalued Attribute .....	384
d. Derived Attribute.....	384
b. Weak Relationship .....	385
Example: Library Book Management.....	386
Example: Employee Management System .....	386
Example: Banking Organization.....	387
Example: Hospital Management.....	387

---

Relational Model.....	388
Introduction.....	388
What is Relational Model?.....	388
IMPORTANT TERMINOLOGIES .....	388
Constraints in Relational Model .....	389
Insertion Anomaly in Referencing Relation: .....	390
Deletion/ Updation Anomaly in Referenced Relation: .....	391
SUPER KEYS:.....	391
Properties of Relation .....	392
Network Model .....	394
Introduction.....	394
The benefits of the network model include: .....	394
The drawbacks of the network model include: .....	394
Hierarchical Model .....	396
Introduction.....	396
Advantages.....	396
Disadvantages .....	397
Relational Database Management System.....	398
Introduction.....	398
Characteristics of physical database design.....	399
Transform the logical data model for target DBMS .....	399
Database Engine.....	400
Database Schema .....	400
Advantages of relational database management system .....	401

---

Other advantages of the RDBMS include:.....	401
Disadvantages of RDBMS .....	401
Applications of RDBMS.....	402
Structure of DBMS .....	403
1-Tier Architecture.....	406
2-Tier Architecture.....	407
3-Tier Architecture.....	408
Three Schema Architecture.....	409
Internal Level.....	409
Conceptual Level .....	410
External Level.....	410
Relational Algebra .....	410
Select Operation.....	411
Project Operation .....	412
Union Operation.....	413
Set Intersection.....	414
Set Difference .....	415
Cartesian Product.....	415
Rename Operation .....	416
Join Operation.....	417
Inner Join .....	417
Theta Join.....	417
EQUI Join .....	417
Left Outer Join .....	417

---

Right Outer Join.....	418
Full Outer Join .....	418
Division Operator.....	418
Relational Calculus .....	419
Introduction.....	419
Types of Relational Calculus .....	419
Tuple Relational Calculus .....	419
Example 1 Find the loan number, branch, amount of loans of greater than or equal to10000 amount.....	421
Example 2 Find the loan number for each loan of an amount greater or equal to 10000...	421
Example 3 Find the names of all customers who have a loan and an account at the bank.	421
Domain Relational Calculus .....	421
Example 1 .....	422
Example 2 .....	422
Example 3 .....	422
Example 4 .....	422
Example 5 .....	422
RDBMS Technologies.....	423
Oracle Database Technology Features.....	423
MySQL Database Technology Features .....	423
MongoDB Database Technology Features .....	423
Microsoft SQL Server Database Technology Features .....	423
Relational Data Structure/Relational Model .....	424
What is a Relational Model? .....	424
Relational Model Concepts .....	424

---

Relational Integrity constraints .....	425
Types of integrity constraints.....	426
1.    Domain constraints .....	426
2.    Entity integrity constraints.....	427
3.    Referential Integrity Constraints .....	428
4.    Key constraints .....	428
Operations in the Relational Model .....	429
Insert Operation: .....	429
Update Operation .....	430
Delete Operation .....	430
Select Operation.....	431
Advantages of using the Relational model.....	431
Disadvantages of using Relational model .....	431
Key and Relational Data Manipulation.....	432
Keys .....	432
What are Keys? .....	432
Why do we need a Key? .....	432
Various Keys.....	433
Super key: .....	433
Primary Key .....	434
Rules for defining Primary key:.....	434
Alternate key .....	435
Candidate Key.....	436
Properties of Candidate key: .....	436

---

Foreign key .....	437
Compound key .....	439
Composite key .....	440
Surrogate Key .....	440
Let's see the difference between keys Primary Key and Foreign Key: .....	441
Primary and Candidate Key: .....	442
Super Key and Candidate Key .....	443
Primary key and Unique key.....	443
Data Manipulation Language (DML) .....	444
Introduction to DML .....	444
DDL commands are as follows:.....	444
1.    SELECT COMMAND .....	444
2.    INSERT COMMAND .....	445
3.    UPDATE COMMAND .....	446
4.    DELETE COMMAND.....	446
Relational Algebra .....	447
Relational Algebra .....	447
Relational Algebraic Operations .....	447
1.    Unary Relational Operations .....	447
2.    Relational Algebra Operations From Set Theory .....	447
3.    Binary Relational Operations .....	447
Unary Relational Operations.....	447
SELECT ( $\sigma$ ).....	447
Projection( $\pi$ ) .....	448

---

Rename( $\rho$ ).....	449
Relational Algebra Operations From Set Theory	
Union operation ( $\cup$ ).....	450
Intersection.....	451
Set Difference (-) .....	452
Cartesian product( $X$ ).....	453
Binary Relational Operations .....	454
JOIN.....	454
Types of JOINS.....	454
Inner Join: .....	455
OUTER JOIN.....	457
Left Outer Join ( $A \bowtie B$ ) .....	457
Right Outer Join: ( $A \ltimes B$ ).....	458
Full Outer Join: ( $A \times B$ ) .....	459
Set Operations .....	460
Types of Set Operation.....	460
1. Union .....	461
2. Union All .....	463
3. Intersect .....	464
4. Minus .....	465
Fundamental Operations .....	466
SELECT ( $\sigma$ ) .....	466
Example 1 .....	466
Example 2 .....	466
Example 3 .....	466

---

Projection( $\pi$ ) .....	466
Example of Projection: .....	467
Cartesian product(X) .....	467
Example – Cartesian product .....	468
Union operation (v) .....	468
Example .....	469
Set Difference (-) .....	470
Example .....	470
Relational Calculus .....	471
What is Relational Calculus? .....	471
1. Tuple Relational Calculus (TRC) .....	472
Example: .....	472
2. Domain Relational Calculus (DRC) .....	477
Predicate Calculus Formula: .....	477
Example: .....	477
Difference between Relational Algebra and Relational Calculus: .....	482
Data Definition language .....	483
Introduction to DDL .....	483
DDL commands are as follows: .....	483
1. CREATE COMMAND .....	483
Syntax: .....	483
2. DROP COMMAND .....	484
Syntax: .....	484
3. ALTER COMMAND .....	484

---

Syntax: .....	484
4. RENAME COMMAND .....	485
Syntax: .....	485
5. TRUNCATE COMMAND .....	486
Syntax: .....	486
Operators: Select, Project, Join, Rename etc .....	487
SELECT ( $\sigma$ ) .....	487
Example 1 .....	487
Example 2 .....	487
Example 3 .....	487
Projection( $\pi$ ) .....	487
Example of Projection: .....	488
JOIN .....	488
Types of JOIN .....	489
Inner Join: .....	489
Theta Join: .....	489
EQUI join: .....	490
NATURAL JOIN ( $\bowtie$ ) .....	491
Num     Square .....	491
Num     Cube .....	491
OUTER JOIN .....	491
Left Outer Join(A  B) .....	492
Right Outer Join: ( A  B ) .....	493
Full Outer Join: ( A  B ) .....	494

---

Rename( $\rho$ ).....	494
Learning Outcome .....	495
Software Development Life Cycle (SDLC).....	495
SDLC Overview.....	496
What is SDLC? .....	496
<b>Why SDLC?</b> .....	496
Phases of Software Development Life Cycle (SDLC) .....	496
Software Development Life Cycle Models.....	500
Software Test Levels.....	505
What is Software Testing? .....	505
Types of Software Testing .....	506
Levels of Testing.....	506

---

## Learning Outcome

In this section, we will read about:

- Introduction to the Internet, browsing, emailing
- Introduction to HTML
- Different editors used for Webpage Development
- Application of HTML

## Introduction to the internet, browsing, emailing

### Overview of the Internet

The Internet is defined as an electronic communications network that connects computer networks and organizational computer facilities around the world. - Merriam-webster

We can say that the internet is a global network of interconnected computer networks and other electronic devices worldwide, which uses standard Internet Protocol (TCP/IP). It is possible to access almost any information, communicate with anyone else in the world using the internet. Networking functionality of internet can be described as below:

- Every computer on the internet is identified by a unique IP address.
- IP Address is a unique set of numbers (such as 110.22.33.114) which identifies a computer location.
- A special computer DNS (Domain Name Server) is used to give a name to the IP Address so that users can locate a computer by name.



Image 1: Internet - users and services are pushed to the edge of the network.

Reference: <https://www.researchgate.net>

## Basics of Internet Architecture

The concept of the Internet was originated in 1969 and has undergone several technological & Infrastructural changes as discussed below:

- 1969: The origin of the Internet devised from the concept of the Advanced Research Project Agency Network (ARPANET).
- ARPANET was developed by the United States Department of Defense.
- The basic purpose of ARPANET was to provide communication among the various bodies of government.
- Initially, there were only four nodes, formally called Hosts.
- In 1972, the ARPANET spread over the globe with 23 nodes located in different countries and thus became known as the Internet.

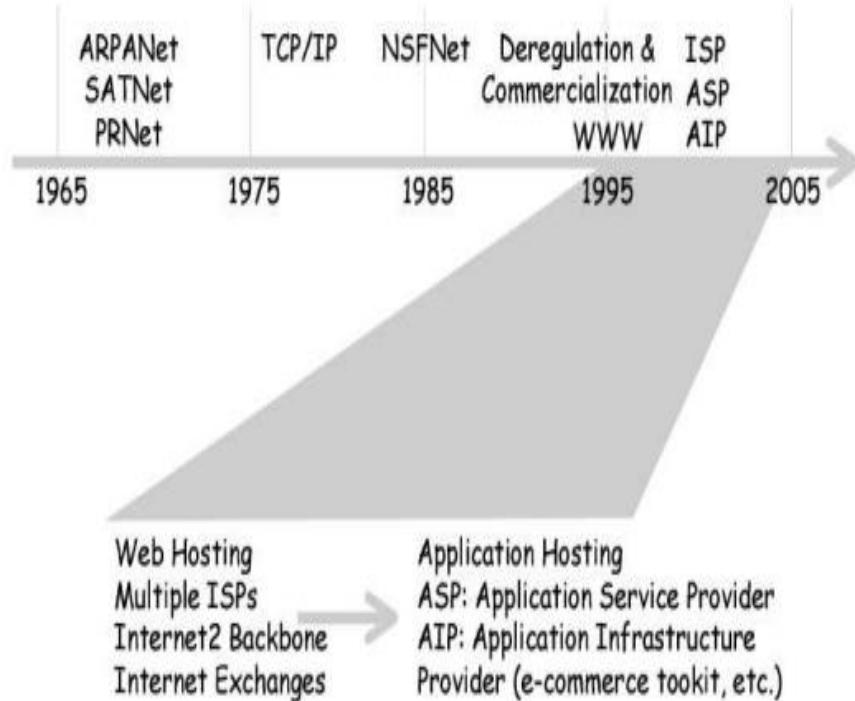


Image 2: Evolution of the Internet  
Reference: <https://www.researchgate.net>

**Internet architecture consists of three layers**

i. **IP**

In order to communicate, we need our data to be encapsulated as Internet Protocol (IP) packets. These IP packets travel across a number of hosts in a network through routing to

---

reach the destination. However, IP does not support error detection and error recovery and is incapable of detecting the loss of packets.

**ii. TCP**

TCP stands for "Transmission Control Protocol". It provides end to end transmission of data, i.e., from source to destination. It is a very complex protocol as it supports the recovery of lost packets.

**iii. Application Protocol**

The third layer in internet architecture is the application layer which has different protocols on which the internet services are built. Some of the examples of internet services include email (SMTP facilitates email feature), file transfer (FTP facilitates file transfer feature), etc.

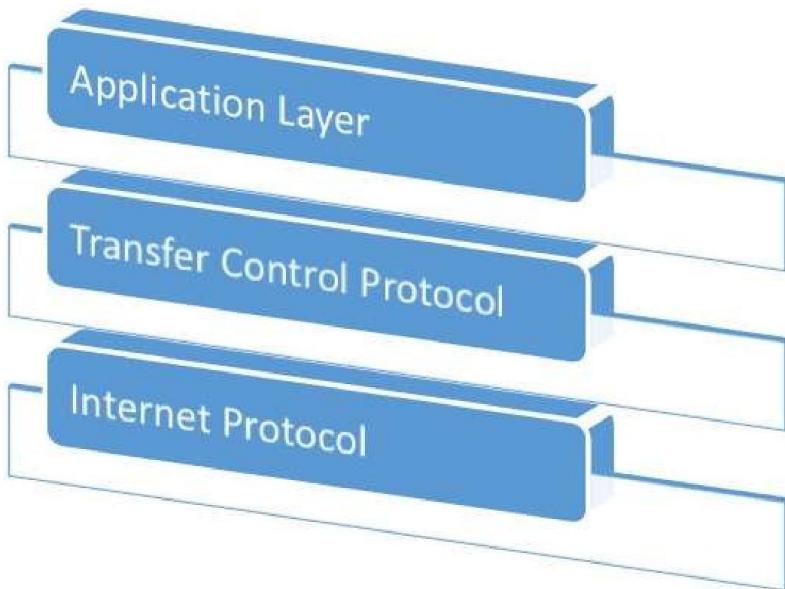


Image 3: Internet architecture  
Reference: [https://www.tutorialspoint.com/computer\\_concepts/computer\\_concepts\\_internet.htm](https://www.tutorialspoint.com/computer_concepts/computer_concepts_internet.htm)

## **Services on the Internet**

Internet Services allows us to access a huge amount of information such as text, graphics, sound and software over the internet.

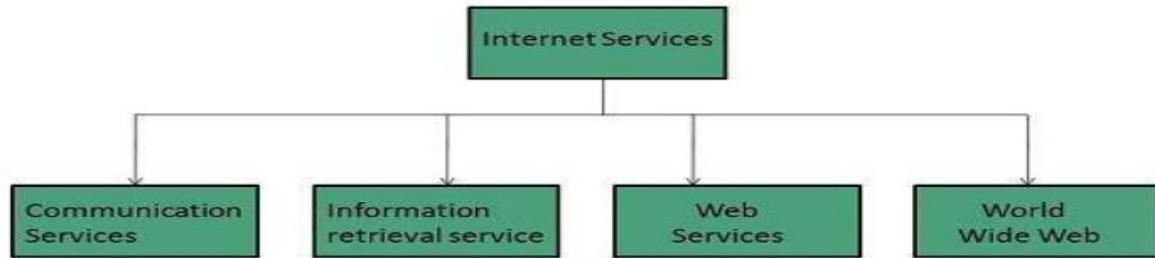


Image 4: categories of Internet Services

Reference: [https://www.tutorialspoint.com/internet\\_technologies/internet\\_services.htm](https://www.tutorialspoint.com/internet_technologies/internet_services.htm)

### i. Communication Services

There are various Communication Services available that offer exchange of information with individuals or groups

- Electronic Mail -Used to send electronic messages over the internet.
- Telnet-Used to log on to a remote computer that is attached to the internet
- Newsgroup-Offers a forum for people to discuss topics of common interests
- Internet Relay Chat (IRC)-Allows people from all over the world to communicate in real-time.
- Mailing Lists-Used to organize a group of internet users to share common information through e-mail.
- Internet Telephony (VoIP)-Allows internet users to talk across the internet to any PC equipped to receive the call.
- Instant Messaging-Offers real-time chat between individuals and groups of people. e.g. Yahoo messenger, MSN messenger.

### ii. Information Retrieval Services

There exist several Information retrieval services offering easy access to information present on the internet.

- File Transfer Protocol (FTP)-Enable the users to transfer files.
- Archie-It's updated database of public FTP sites and their content. It helps to search a file by its name.
- Gopher-Used to search, retrieve, and display documents on remote sites.

- 
- Very Easy Rodent Oriented Netwide Index to Computer Achieved (VERONICA)- VERONICA is a gopher-based resource. It allows access to the information resource stored on gopher's servers.

### **iii. Web Services**

Web services allow the exchange of information between applications on the web. Using web services, applications can easily interact with each other.

- The web services are offered using the concept of Utility Computing.

### **iv. World Wide Web (WWW)**

WWW is also known as W3. It offers a way to access documents spread over several servers over the internet. These documents may contain texts, graphics, audio, video, hyperlinks. The hyperlinks allow the users to navigate between the documents.

#### **Accessing Web Browser**

A web browser is a software application that enables a user to display and interact with text, images, videos, music, and other information that could be on a website.

There are several ways to access a web page like using URLs, hyperlinks, navigating tools, search engines, etc.

##### **a) Using URLs**

URL refers to "Uniform Resource Locator". Each and every website can be recognized using a unique address called "Uniform Resource Locator" or simply a URL. Once you provide the URL of a specific page in the address bar, the web browser will find the corresponding page and display the result to the user.

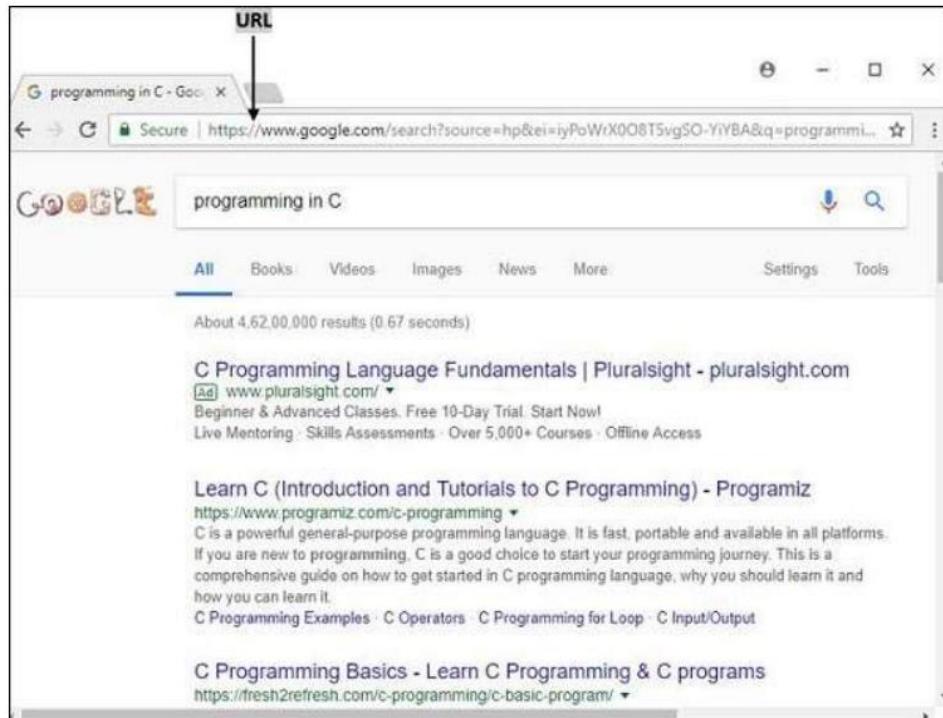


Image 5: URL

Reference: <https://www.google.com/search?q=programming+in+c&source>

## b) Using Hyperlinks

"Hyperlink" is a part of a web page that is linked to a URL. A hyperlink can be text, image, button, arrow, etc. By clicking on a hyperlink, you can move to a different URL specified in the link from the current URL. Hyperlinked text is an underlined blue color text which is represented using a hand symbol.

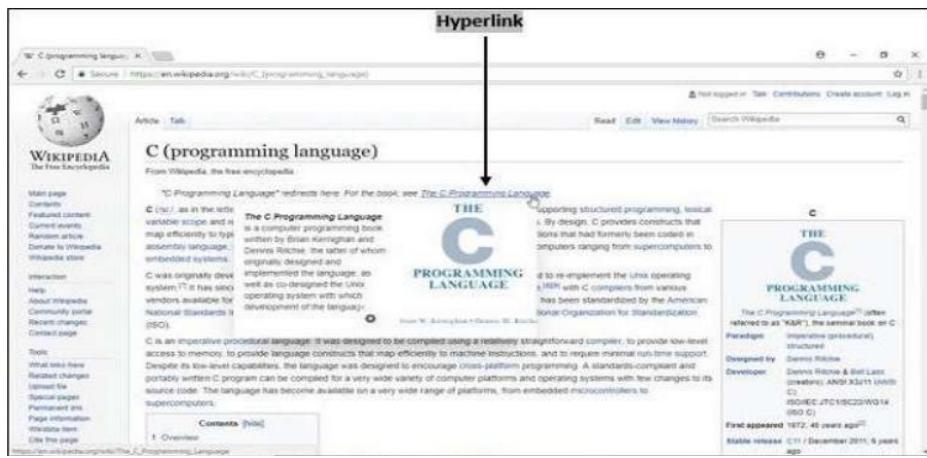


Image 6: Hyperlink

Reference: [https://en.wikipedia.org/wiki/C\\_\(programming\\_language\)](https://en.wikipedia.org/wiki/C_(programming_language))

## c) Using Browsers Navigation Tools

Web browsers offer a variety of tools to help you move around the web. These tools help you to quickly go back and forth through web pages.

Back Button — Helps to move back to the previous page from the current page.



Image 7: Back Button

Reference: [https://en.wikipedia.org/wiki/C\\_\(programming\\_language\)](https://en.wikipedia.org/wiki/C_(programming_language))

Forward Button - Helps to move to the next page from the current page.

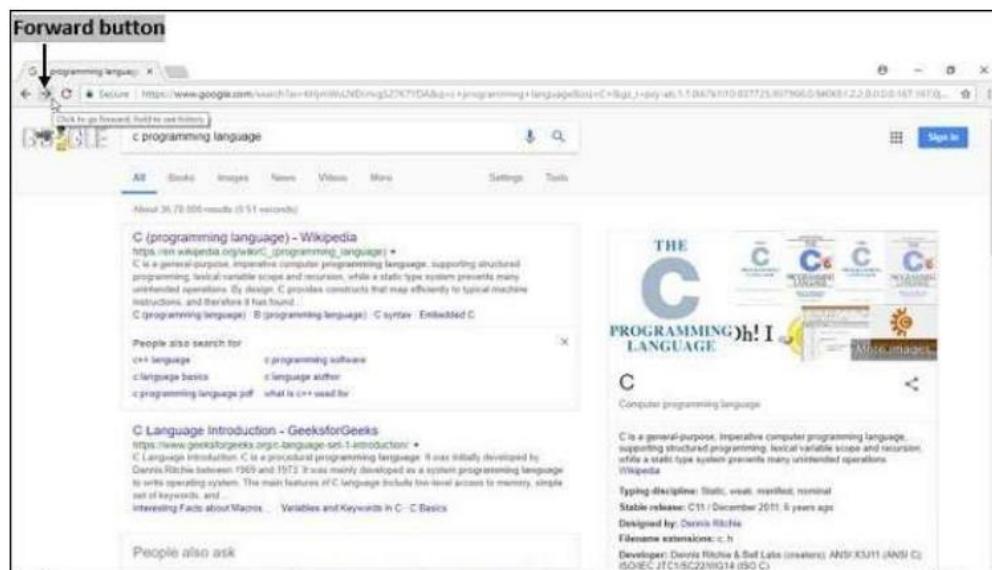


Image 8: Forward Button

Reference: [https://en.wikipedia.org/wiki/C\\_\(programming\\_language\)](https://en.wikipedia.org/wiki/C_(programming_language))

Refresh Button - Helps to refresh a current page.

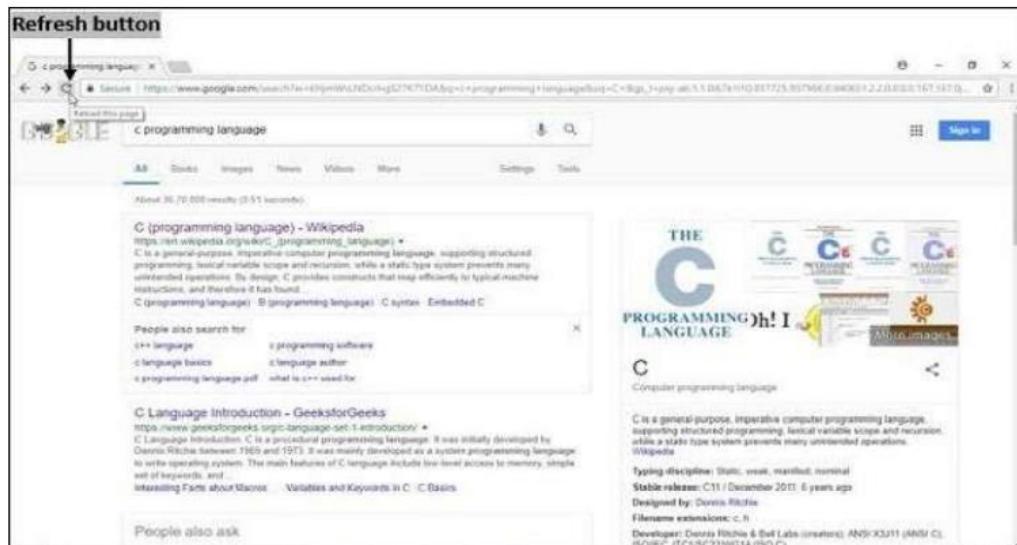


Image 9: Refresh Button

Reference: [https://en.wikipedia.org/wiki/C\\_\(programming\\_language\)](https://en.wikipedia.org/wiki/C_(programming_language))

Close Button - Helps to close a web page.

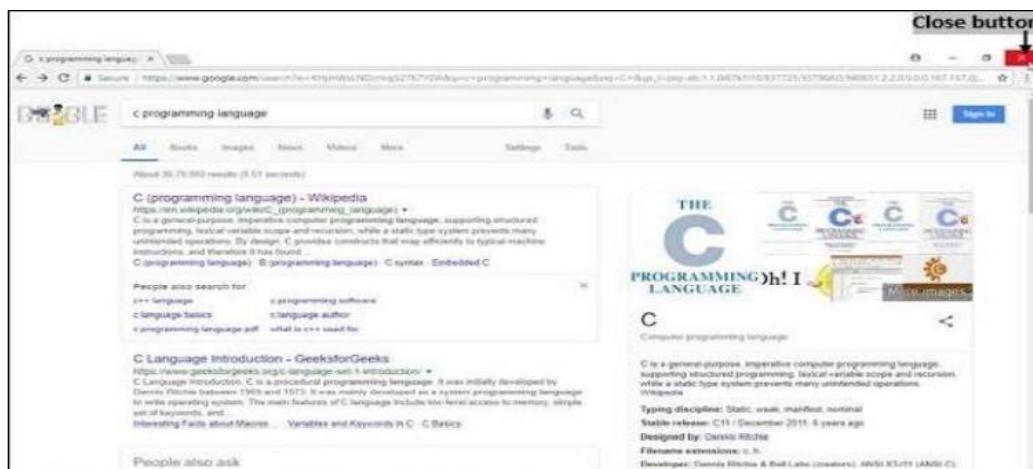


Image 10: Close Button

Reference: [https://en.wikipedia.org/wiki/C\\_\(programming\\_language\)](https://en.wikipedia.org/wiki/C_(programming_language))

#### d) Using Bookmark

Web browsers allow you to bookmark pages that you visit most frequently. This helps you to go to a web page directly by selecting from a list of bookmarks instead of typing the URL multiple times. This is displayed as an icon with a star symbol in the top right corner of the page.

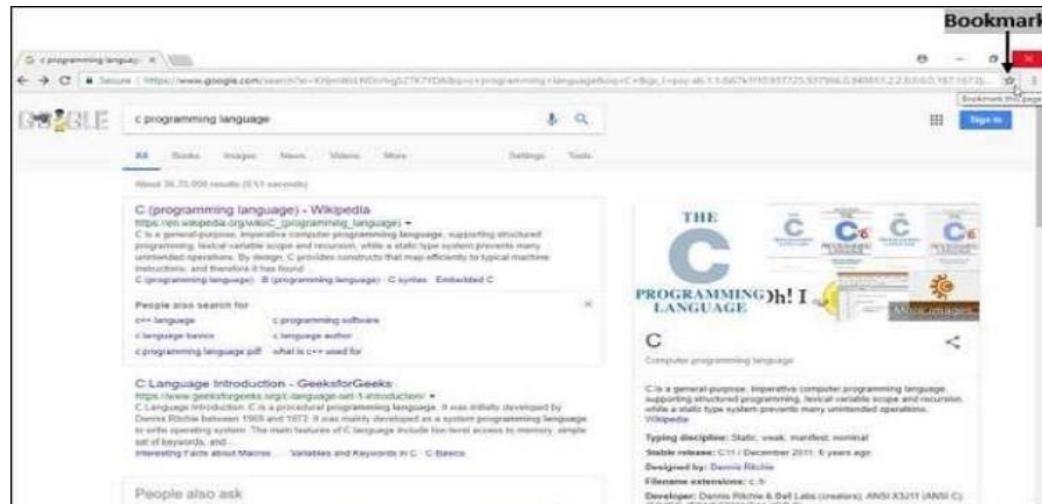


Image 11: Using Bookmark

Reference: [https://en.wikipedia.org/wiki/C\\_\(programming\\_language\)](https://en.wikipedia.org/wiki/C_(programming_language))

## e) Using History

When you type any URL in the address bar, the browser saves that URL automatically, thus creating a history list for the current session. You can choose the URL you want from the history list instead of typing it again.



Image 12: Using History

Reference: [https://en.wikipedia.org/wiki/C\\_\(programming\\_language\)](https://en.wikipedia.org/wiki/C_(programming_language))



Image 13: Select History

Reference: [https://en.wikipedia.org/wiki/C\\_\(programming\\_language\)](https://en.wikipedia.org/wiki/C_(programming_language))

## f) Using Search Engines

A search engine is an application that allows you to search for content on the web. It displays multiple web pages based on the content or a word you have typed. The most popular search engines include Google, Yahoo, Ask, etc. Below are the steps to use a search engine.

Step 1 - Launch your web browser.



Image 14: Launch Web browser

Reference: [https://www.tutorialspoint.com/computer\\_concepts/computer\\_concepts\\_web\\_browsing\\_software.htm](https://www.tutorialspoint.com/computer_concepts/computer_concepts_web_browsing_software.htm)

Step 2 - In "Address bar/Location", type the search engine you want to use and press enter.



Image 15: Enter the name of search engine

Reference: [https://www.tutorialspoint.com/computer\\_concepts/computer\\_concepts\\_web\\_browsing\\_software.htm](https://www.tutorialspoint.com/computer_concepts/computer_concepts_web_browsing_software.htm)

Step 3 - Type the content you want to search in the "search text box" and press enter.

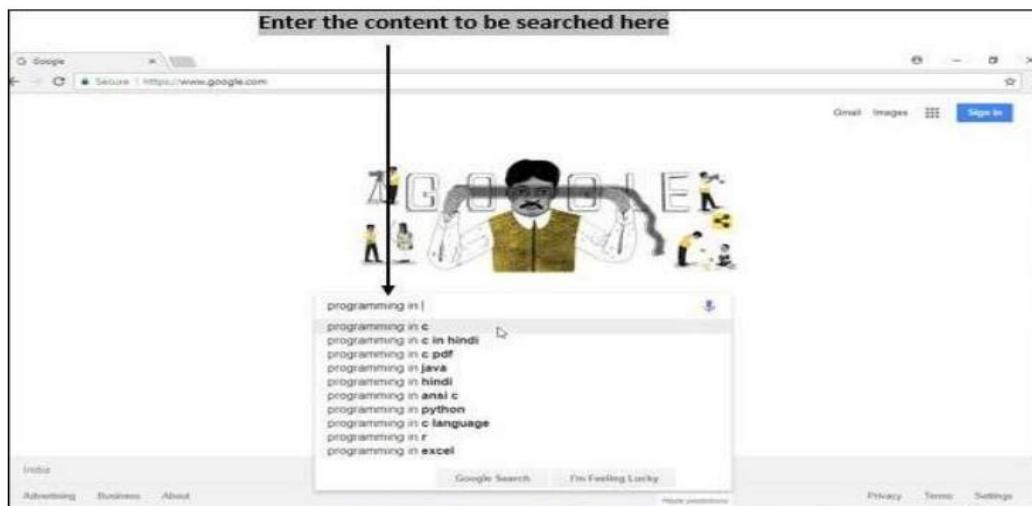


Image 16: Enter the content to be search

Reference: [https://www.tutorialspoint.com/computer\\_concepts/computer\\_concepts\\_web\\_browsing\\_software.htm](https://www.tutorialspoint.com/computer_concepts/computer_concepts_web_browsing_software.htm)

Step 4 - It displays a list of web pages from which you can select the content/web page you want.



Image 17: Webpages related to the search will be displayed

Reference: [https://www.tutorialspoint.com/computer\\_concepts/computer\\_concepts\\_web\\_browsing\\_software.htm](https://www.tutorialspoint.com/computer_concepts/computer_concepts_web_browsing_software.htm)

## g) Using Favorites Folder

Web browsers allow you to bookmark the pages that you visit most frequently. The Favorites folder is called Bookmarks. This helps you to go to the web page directly by selecting from the list of bookmarks instead of typing the URL again and again. Adding and removing the pages to the favorite folder include the following steps.

Step 1 - Click the star icon present at the top right corner of the page.

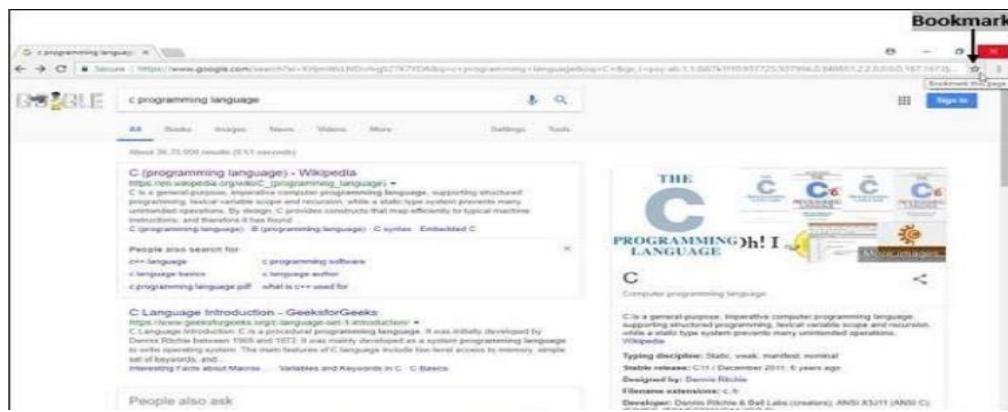


Image 18: Bookmark

Reference: [https://en.wikipedia.org/wiki/C\\_\(programming\\_language\)](https://en.wikipedia.org/wiki/C_(programming_language))

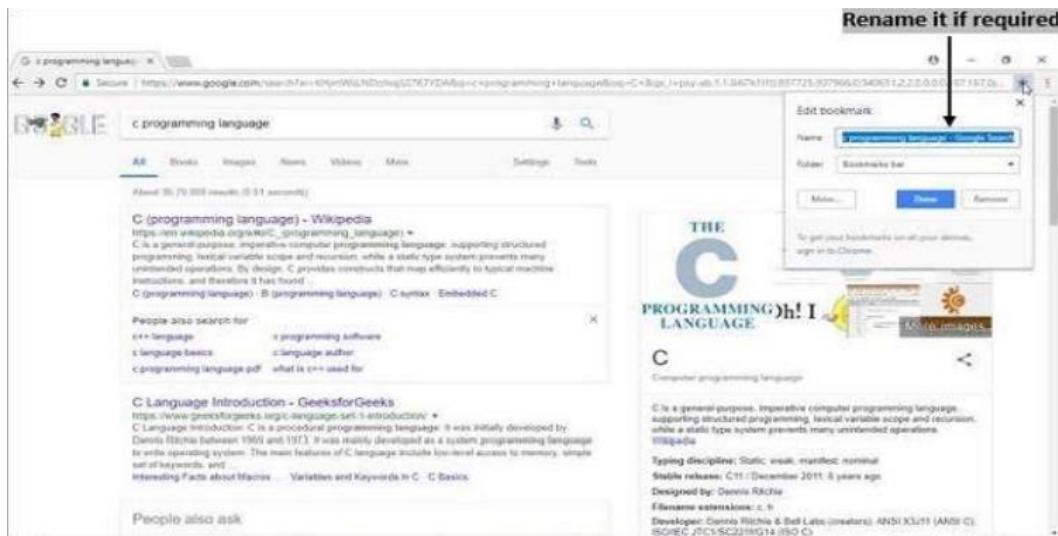


Image 19: Rename

Reference: [https://en.wikipedia.org/wiki/C\\_\(programming\\_language\)](https://en.wikipedia.org/wiki/C_(programming_language))

Step 2 - In order to add the web page, type the page you want to add as favorite and click "Done" button or you can click the three vertical dots (⋮) icon on the top right corner of the screen and select "Bookmark this page" option from the displayed menu to bookmark the current page.

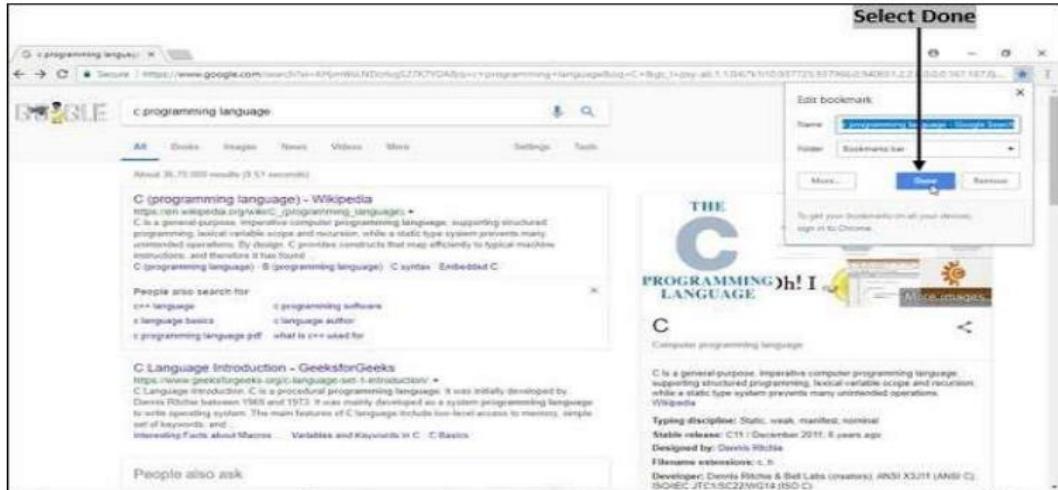


Image 20: Bookmark Done

Reference: [https://en.wikipedia.org/wiki/C\\_\(programming\\_language\)](https://en.wikipedia.org/wiki/C_(programming_language))

Step 3 - In order to remove the web page, select a page and press "Remove" button or you can click the three vertical dots icon(⋮) on the top right corner of the screen and select the web page you want to delete and "Right-dick" and click "Delete" option.



Image 21: Remove Bookmark

Reference: [https://www.tutorialspoint.com/computer\\_concepts/computer\\_concepts\\_web\\_browsing\\_software.htm](https://www.tutorialspoint.com/computer_concepts/computer_concepts_web_browsing_software.htm)

## Downloading Web Pages

Downloading is saving a file or document or web page on your hard disk. It consists of the following steps.

Step 1 - Open a web browser and navigate to the webpage which you want to download.



Image 22: Downloading web pages

Reference: [https://www.tutorialspoint.com/computer\\_concepts/computer\\_concepts\\_web\\_browsing\\_software.htm](https://www.tutorialspoint.com/computer_concepts/computer_concepts_web_browsing_software.htm)

Step 2 - Right-click on the file and choose Save as.

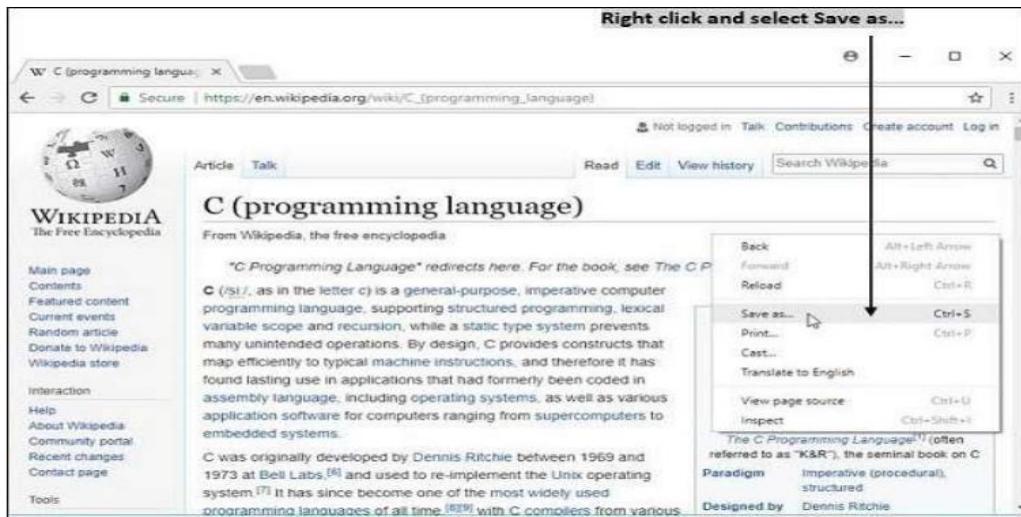


Image 23: Save as a webpage

Reference: [https://www.tutorialspoint.com/computer\\_concepts/computer\\_concepts\\_web\\_browsing\\_software.htm](https://www.tutorialspoint.com/computer_concepts/computer_concepts_web_browsing_software.htm)

Step 3 - Choose where you want to save the file, then click Save.

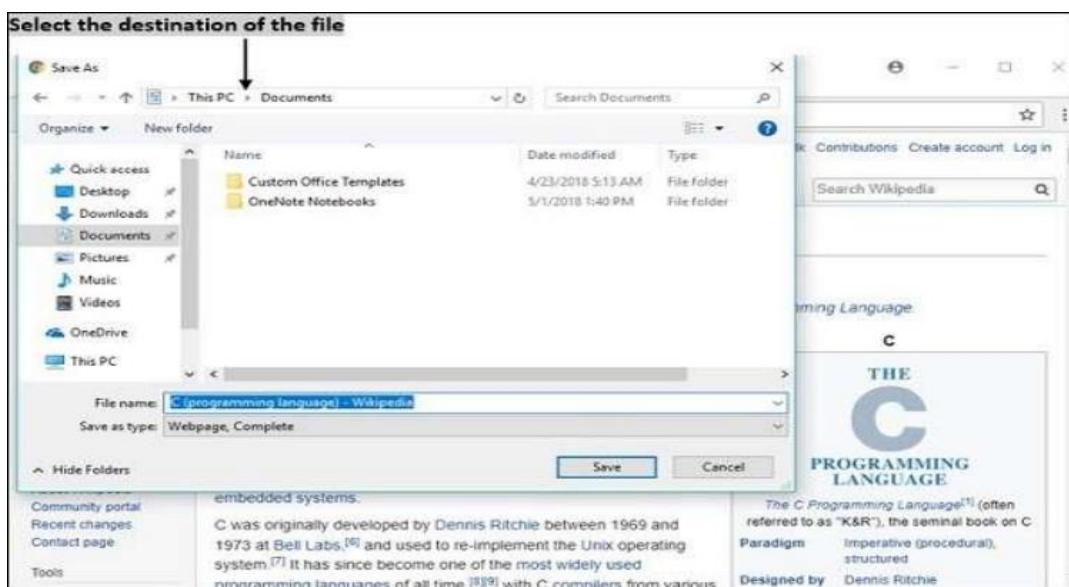


Image 24: Select the Destination File

Reference: [https://www.tutorialspoint.com/computer\\_concepts/computer\\_concepts\\_web\\_browsing\\_software.htm](https://www.tutorialspoint.com/computer_concepts/computer_concepts_web_browsing_software.htm)

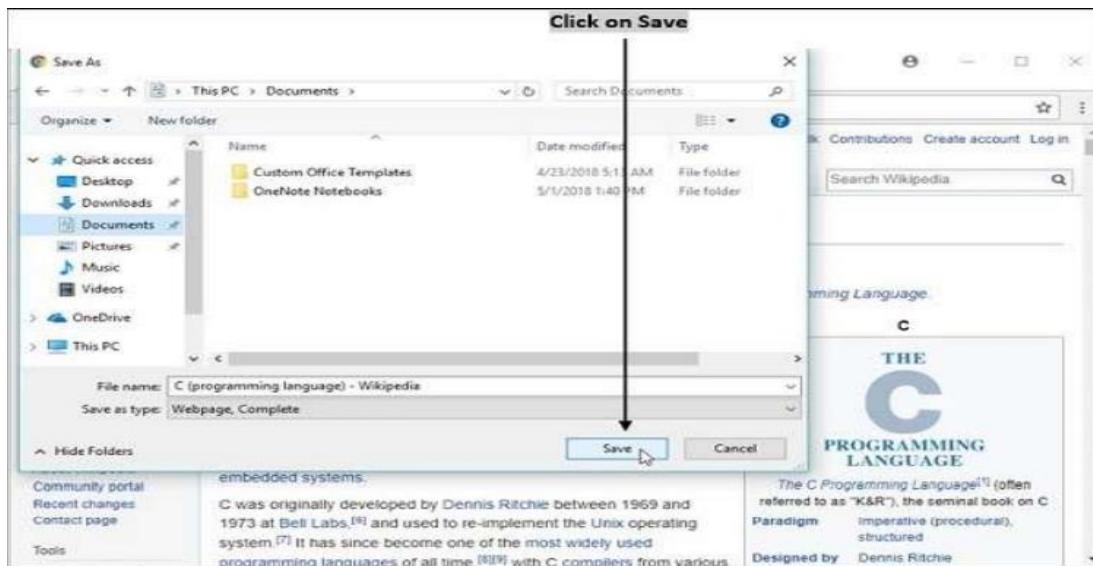


Image 25: Save web page

Reference: [https://www.tutorialspoint.com/computer\\_concepts/computer\\_concepts\\_web\\_browsing\\_software.htm](https://www.tutorialspoint.com/computer_concepts/computer_concepts_web_browsing_software.htm)

Step 4 - When the file is downloaded, you'll see it at the bottom of your Chrome window. Click the file name to open it.



Image 26: Downloaded File

Reference: [https://www.tutorialspoint.com/computer\\_concepts/computer\\_concepts\\_web\\_browsing\\_software.htm](https://www.tutorialspoint.com/computer_concepts/computer_concepts_web_browsing_software.htm)

## Pause, Resume or Cancel

At the bottom of the screen, you can see the downloading file. Click the arrow next to the file name at the bottom of your screen. Click Pause, Resume or Cancel, whatever action you want to perform.



Image 27: Pause Resume Cancel

Reference: [https://www.tutorialspoint.com/computer\\_concepts/computer\\_concepts\\_web\\_browsing\\_software.htm](https://www.tutorialspoint.com/computer_concepts/computer_concepts_web_browsing_software.htm)

## Printing Web Pages

Printing is creating a hard copy of a document which can be a web page or any other content. It includes the following steps -

Step 1 - After launching a web browser, open the page, image, or file you want to print.

Step 2 - Click on three vertical dots icon (⋮) on the top right corner of the screen or use a keyboard shortcut: Ctrl + P.



Image 28: Select Print

Reference: [https://www.tutorialspoint.com/computer\\_concepts/computer\\_concepts\\_web\\_browsing\\_software.htm](https://www.tutorialspoint.com/computer_concepts/computer_concepts_web_browsing_software.htm)

Step 3 - In the window that appears, select destination and change any print settings you want; when read y, click Print.

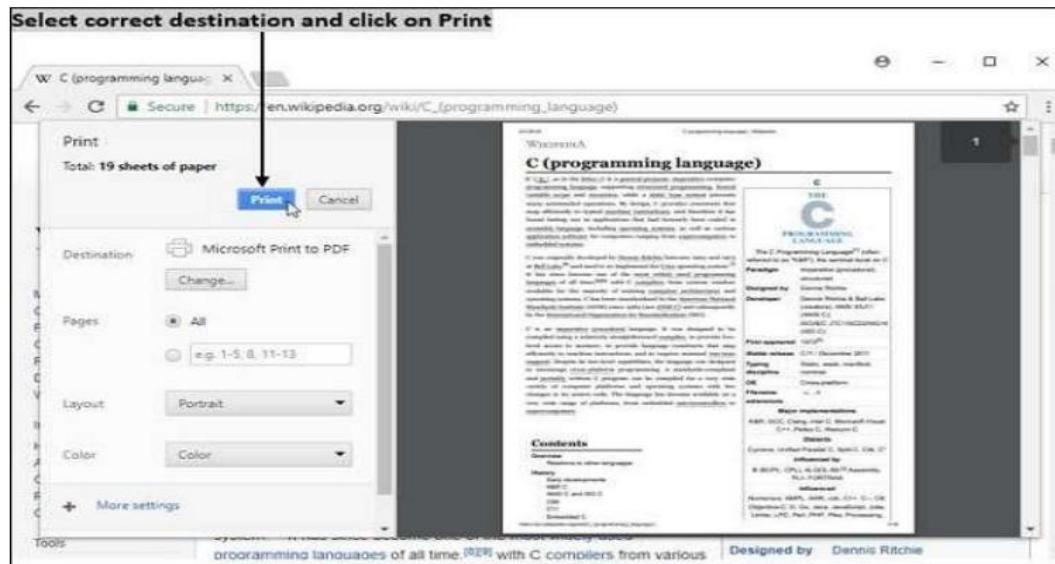


Image 29: Printing

Reference: [https://www.tutorialspoint.com/computer\\_concepts/computer\\_concepts\\_web\\_browsing\\_software.htm](https://www.tutorialspoint.com/computer_concepts/computer_concepts_web_browsing_software.htm)

## Search Engines

Search Engine refers to a huge database of internet resources such as web pages, newsgroups, programs, images, etc. It helps to locate information on the World Wide Web.

Users can search for any information by passing a query in the form of keywords or phrases. It then searches for relevant information in its database and returns it to the user.



Image 30: Search Engine

Reference: [https://www.tutorialspoint.com/internet\\_technologies/search\\_engines.htm](https://www.tutorialspoint.com/internet_technologies/search_engines.htm)

## Search Engine Components

Generally, there are three basic components of a search engine as listed below:

**a) Web crawler**

It is also known as a spider or bots. It is a software component that traverses the web together information.

**b) Database**

All the information on the web is stored in a database. It consists of huge web resources.

**c) Search Interfaces**

This component is an interface between the user and the database. It helps the user to search through the database.

- Search engines work by crawling hundreds of billions of pages using their own web crawlers. These web crawlers are commonly referred to as search engine bots or spiders. A search engine navigates the web by downloading web pages and following links on these pages to discover new pages that have been made available.

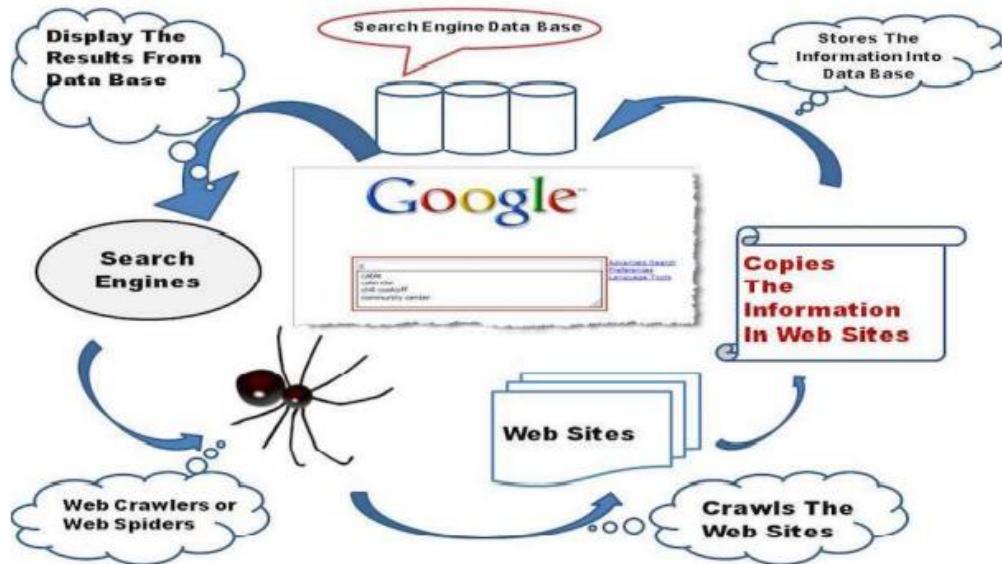


Image 31: How search engines Works?

Reference: <https://fossbytes.com/search-engine-works-makes-life -easier/>



Image 32: Popular Web Browsers  
Reference: <https://wildstonesolution.com/top-search-engines>

## Services Available on the Internet

### 1. Data Transfer

Data transfer is the process of using computing techniques and technologies to transmit or transfer electronic or analog data from one computer node to another. Data is transferred in the form of bits and bytes over a digital or analog medium, and the process enables digital or analog communications and its movement between devices. Data transfer is also known as data transmission.

### 2. Internet Banking:

Traditionally, customers used to access banking services through Retail/ corporate branches. But in this digital era, Online Banking has taken a vital role. The online banking is also called as internet banking, virtual banking or e-banking. This is a value-added application to connect the core banking system and provide the self-service bank facilities for customers online.



Image 33: Internet banking  
Reference: [https://www.brainkart.com/article/Services-Available-on-the-Internet\\_36837/](https://www.brainkart.com/article/Services-Available-on-the-Internet_36837/)

---

Different features are: -

- A bank customer can perform transactional and non-transactional tasks through online banking
- Viewing account balances, transactions, statements of customer
- Viewing images of paid cheques request for cheque books
- Funds transfers between the customer's linked accounts
- Paying third parties, including bill payments and third-party fund transfers
- Register utility billers and make bill payments

Advantages: -

- Permanent online access for banking transactions.
- Access anywhere using the application via mobile or computer
- Less time consuming, easy to use and safe
- Customer can manage their funds instantly and accurately

### 3. E-commerce:

E-commerce application is a transaction of buying or selling goods and services online. Electronic commerce attraction technologies such as mobile commerce, electronic funds transfer, supply chain management, Internet marketing, online transaction processing, electronic data interchange (EDI), inventory management systems, and automated data collection systems.

E-commerce businesses may also employ some or all of the followings:

- Online shopping web sites for retail sales direct to consumers
- Providing or participating in online marketplaces, which process third-party business-to consumer or consumer-to-consumer sales
- Business-to-business buying and selling;
- Gathering and using demographic data through web contacts and social media
- Business-to-business (B2B) electronic data interchange

- Marketing to prospective and established customers by e-mail or fax (for example, with newsletters)
- Engaging in retail for launching new products and services
- Online financial exchanges for currency exchanges or trading purposes.

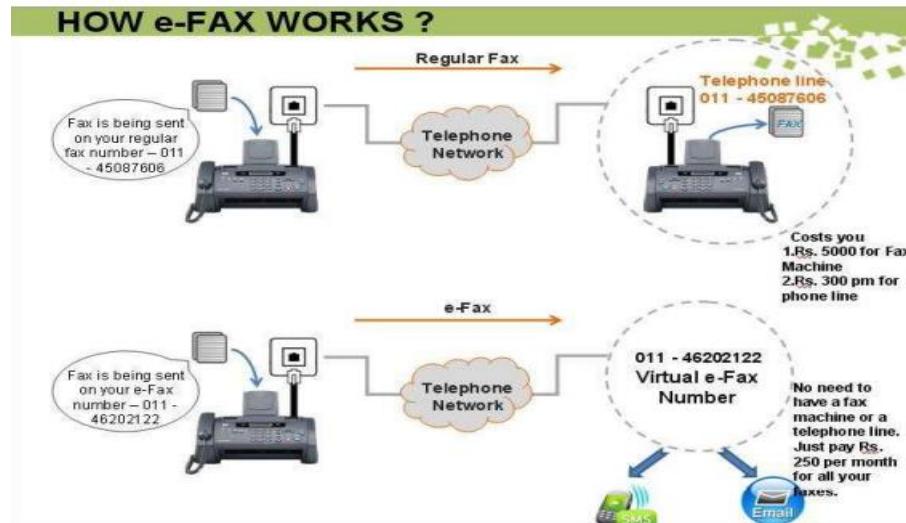


Image 34: eFax works

Reference: [https://www.brainkart.com/article/Services-Available-on-the-Internet\\_36837/](https://www.brainkart.com/article/Services-Available-on-the-Internet_36837/)

#### 4. e-Marketing: (Electronic Marketing)

E-Marketing is the process of marketing a product or service using the internet. It also includes marketing done via email and wireless media. It is also called Digital Marketing.



Image 35: e-marketing

Reference: [https://www.brainkart.com/article/Services-Available-on-the-Internet\\_36837/](https://www.brainkart.com/article/Services-Available-on-the-Internet_36837/)

Professionals working in e-marketing must design and implement Internet marketing plans. But they also must have a broad understanding of what makes these plans effective. Those working in e-marketing must be able to carry out many tasks:

- Following business market trends
- Consulting with companies about digital marketing needs

- Resolving issues businesses have in reaching customers-oriented problems
- Creating e-marketing objectives for business challenges
- Developing marketing strategies competitive business model.
- Choosing cost-effective marketing methods
- Launching digital marketing campaigns and monitoring results

#### 5. E-Learning:

A learning system based on electronic resources is known as E-learning. The use of computers and the Internet forms the major component of E-learning. E-learning can also be termed as a network-enabled transfer of skills and knowledge and the delivery of education is made to a large number of recipients at the same or different times.

#### 6. E-Governance:

E-governance is the application of Information and Communication Technology (ICT) for delivering government services, exchange of information, communication transactions, integration of various stand-alone systems and services between government-to-citizen (G2C), Government-to-business (G2B), Government-to-Government (G2G), Government-to-Employees (G2E) as well as back-office processes and interactions within the entire government framework. Through e-governance, government services will be made available to citizens in a convenient, efficient and transparent manner. The three main target groups that can be distinguished in governance concepts are government, citizens and businesses/ interest groups. In e-governance, there are no distinct boundaries.

It classifies four basic models they are:

- Government-to-Citizen (customer)
- Government-to-Employees
- Government-to-Government
- Government-to-Business

Examples of e-Governance:

- Aadhaar Card is a 12-digit unique identity number issued to all Indian residents based on their biometric and demographic data.

- Inspector General of Registration portal - Tamil Nadu - [www.tnreginet.gov.in](http://www.tnreginet.gov.in) used for Land and legal registrations.



Image 36: e-governance

Reference: [https://www.brainkart.com/article/Services-Available-on-the-Internet\\_36837/](https://www.brainkart.com/article/Services-Available-on-the-Internet_36837/)

## 7. Online Chatting

Online chat refers to any kind of communication via the Internet that offers a real-time transmission of text messages from sender to receiver. The chat messages are generally short in order to enable other participants to respond quickly. Online chat may address point-to-point communications as well as multicast communications from one sender to many receivers and voice and video chat and web conferencing service.

## Advantages of Internet

### 1. Communication

The main advantage of the internet is faster communication than any other device. It's an instant process. Communication in the form of video calls, emails, etc. is possible using the internet. Thus, there is no specific region that can be accessed. It is accessible all over the world. Hence, because of these global issues are reduced since video conferencing is possible where everyone across the world can be in a single place and can solve a problem.

---

## 2. Information

The internet is a source of knowledge. All kinds of information are present in it. It is easily accessed and can be searched more to get more additional knowledge. Information like educational related, government laws, market sales, stocks and shares, new creations, etc. are gathered from a single place.

## 3. Learning

The internet has now become a part of education. Education like homeschooling is easily carried out using the internet. Teachers can upload their teaching videos on the internet and are accessed by people across the world which is helpful for all students. The marks are also released on the internet since releasing marks for the whole institution in notice boards will create chaos.

## 4. Entertainment:

The internet is now the most popular form of entertainment. Movies, songs, videos, games, etc. are available on the internet for free. Social networking is also possible using the internet. Hence, there are tons of entertainment that are available online on the internet.

## 5. Social network:

Social networking is the sharing of information with people across the world. Apart from being an entertainment website, it has many uses. Any job vacancy, emergency news, ideas, etc. can be shared on the website and the information gets passed on quickly to a wide area. Also, social networking websites are used for easy communications. Example: Facebook and Twitter.

## 6. E-commerce:

All business deals can be carried on the internet like transactions of money etc. This is called E-commerce. Online reservations, online ticket booking for movies etc. can be done easily. It saves us lots of time. Online shopping is now the latest trend in the internet world where products from dresses to household furniture are available at doorstep.



Image 37: Service available in the internet  
Reference: <https://indiacelebratings.com/advantages-disadvantages-of-internet/>

## 7. E-Mail

Electronic Mail (email or e-mail) is a method of exchanging messages between people using electronic devices. Email first entered limited use in the 1960s and by the middle of 1970s had taken the form now recognized as email. Email operates across computer networks, which is primarily called the Internet.

Earlier email systems required the sender and the recipient to both be online at the same time, in common with instant messaging. Today's email systems are based on a store-and-forward model. Email servers accept, forward, deliver, and store messages.

The structure of the E-mail address is `username@domain name`

An example of E-mail address is raman@gmail.com

An E-mail address consists of two parts separated by @ symbol. The first part Raman is the user's name that identifies the address and the second part gmail.com is the domain name of the E-mail server.

Sample Email Application

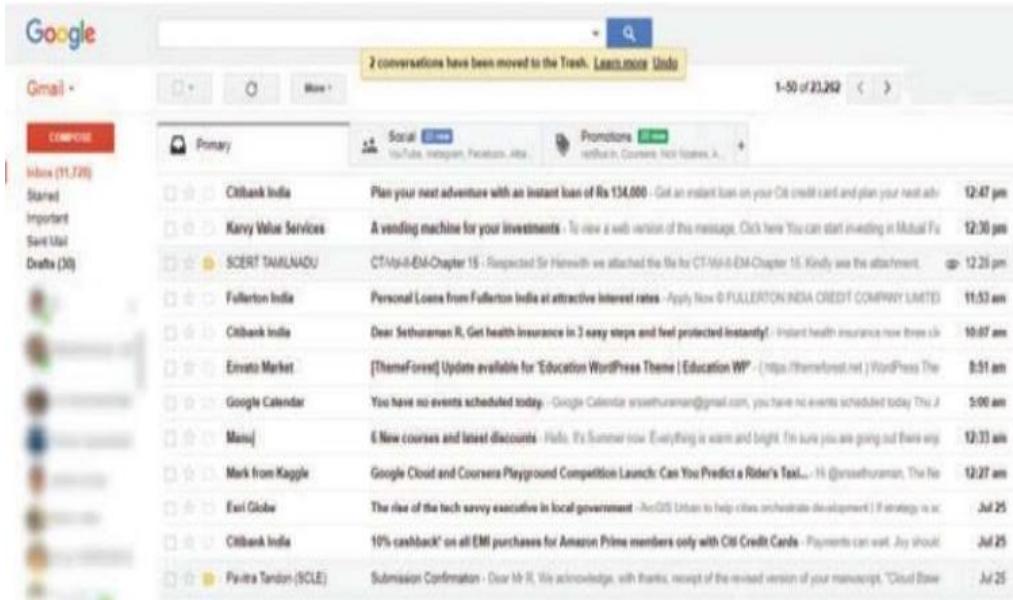


Image 38: Sample Email

Reference: [https://www.brainkart.com/article/Structure-and-Working-of-E-Mail\\_36840/](https://www.brainkart.com/article/Structure-and-Working-of-E-Mail_36840/)

### How Email works on the Internet:

- To send Internet e-mail, requires an Internet connection and access to a mail server. The standard protocol used for sending Internet e-mail is called SMTP (Simple Mail Transfer Protocol). The SMTP protocol is used to both send and receive email messages over the Internet.
- When a message is sent, the email client sends the message to the SMTP server. If the recipient of the email is local the message is kept on the server for accessing by the POP, IMAP or other mail services for later retrieval.
- If the recipient is remote (i.e., at another domain), the SMTP server communicates with a Domain Name Server (DNS) to find the corresponding IP address for the domain being sent to. Once the IP address has been resolved, the SMTP server connects with the remote SMTP server and the mail is delivered to this server for handling.
- If the SMTP server sending the mail is unable to connect with the remote SMTP server, then the message goes into a queue. Messages in this queue will be retried periodically. If the message is still undelivered after a certain amount of time (30 hours by default), the message will be returned to the sender as undelivered.

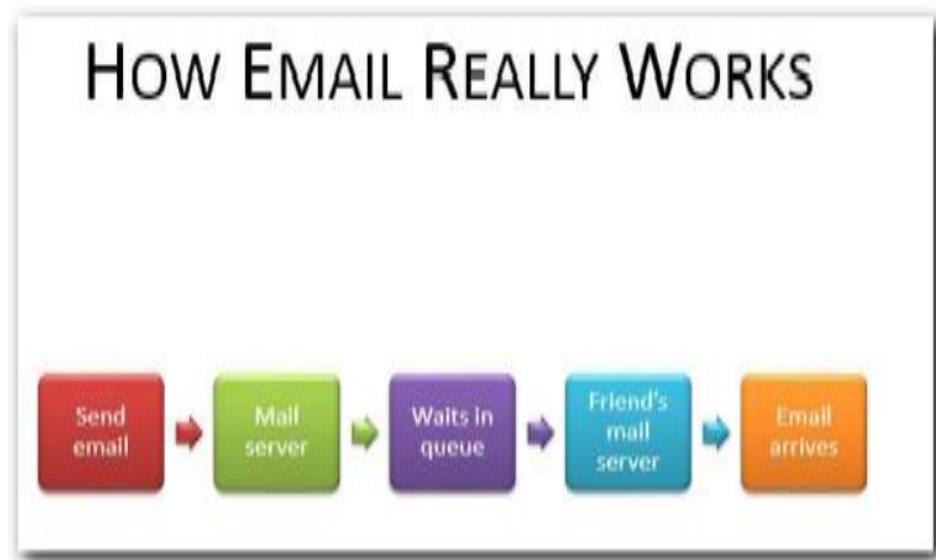


Image 39: How Email really works?

Reference: [https://www.brainkart.com/article/Structure-and-Working-of-E-Mail\\_36840/](https://www.brainkart.com/article/Structure-and-Working-of-E-Mail_36840/)

Structure of an Email message:

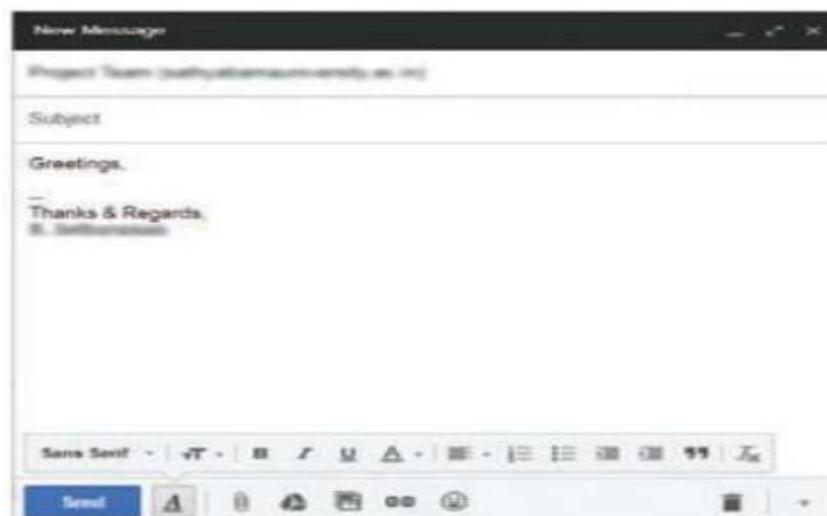


Image 40: Structure of an Email

Reference: [https://www.brainkart.com/article/Structure-and-Working-of-E-Mail\\_36840/](https://www.brainkart.com/article/Structure-and-Working-of-E-Mail_36840/)

**To:** This field consists of the address to whom the message has to be sent. This is mandatory.

**CC:** Short for carbon copy. This is optional. The people who were mailed copies of the message. The recipients of the message will know to whom all the copies have been sent.

---

BCC: It stands for Black Carbon Copy. It is used when we do not want one or more of the recipients to know that someone else was copied on the message. This is optional.

Subject: The Subject field indicates the purpose of the e-mail.

Attachment: Attachment contains files that you are sending, linked documents, pictures, etc. along with an e-mail.

Body: The email body is the main part of an email message. It contains the message's text, images and other data (such as attachments). The email's body is distinct from its header, which contains control information and data about the message (such as its sender, the recipient and the path an email took to reach its destination).

Signature: Name of the sender

Advantages of Email:

- Reliable: Because it notifies the sender if not delivered.
- Speed: E-mail is very fast delivered in a fraction of seconds.
- Inexpensive: It's very cheap.
- Waste Reduction: Helps in paperless communication thus eco-friendly.

Disadvantages of Email:

- Forgery: Anyone who hacks the password of the sender can send a message to anyone.
- Overload: Because it is cheap, loads and loads of messages keep coming.
- Junk: Junk emails are not intended mails and are inappropriate also. Junk emails are sometimes referred to as spam.

## Internet Applications

The Internet is a network of computers linking many different types of computers all over the world. It is a network of networks sharing a common mechanism for addressing(identifying) computers and a common set of communication protocols for communications between two computers on the network.

---

## Applications of Internet

### 1. Communication

Computer users around the world extensively use the email service on the internet to communicate with each other. Pictures, documents and other files are sent as email attachments. Emails can be cc-ed to multiple email addresses

Internet telephony is another common communications service made possible by the creation of the Internet. VoIP stands for Voice-over-Internet Protocol, referring to the protocol that underlies all Internet communication.

### 2. Job search

Nowadays, many people search for their jobs online as it is quicker and there is a larger variety of job vacancies present. People can publish resumes online for prospective jobs. Some of the web sites providing this service are naukri.com, monster.com, summerjob.com, recruitmentindia.com etc.

### 3. Online Shopping

The internet has also facilitated the introduction of a new market concept consisting of virtual shops. They provide information about products or services for sale through www servers. Using the internet services customers can submit specific product queries and request specific sales quotes. For example, amazon.com is a www-based bookshop on the internet where information on all types of international books can be found and books can be ordered online.

### 4. Stock market updates

You can sell or buy shares while sitting on the computer through the internet. Several websites like ndtvprofit.com, moneypore.com, provide information regarding investment

### 5. Travel

One can use the internet to gather information about various tourist places. It can be used for booking Holiday tours, hotels, train, bus, flights and cabs. Some of the web sites providing this service are goibibo.com, makemytrip.com, olacabs.com.

### 6. Research

Research papers are present online which helps in the researcher doing a literature review.

## 7. Video Conferencing:

It enables direct face-to-face communication across networks via web cameras, microphones, and other communication tools. Video conferencing can enable individuals in distant locations to participate in meetings on short notice, with time and money savings. The technology is also used for telecommuting, in which employees work from home. When video Conferencing is used in education, it is easier to have interactive communications between teacher to teacher, teacher to classroom, or classroom to classroom with students in different places.

## 8. E-Commerce

E-commerce (electronic commerce or EC) is the buying and selling of goods and services, or the transmitting of funds or data, over an electronic network, primarily the Internet. These business transactions occur either business-to-business, business-to-consumer, consumer-to-consumer or consumer-to-business. Largest e-commerce companies in India are Flipkart, Snapdeal, Amazon India, Paytm.

## 9. On-line payments

The rising boom of online payments in India has given way to many new entrants in the industry such as Paytm, MobiKwik, Oxigen etc who are majorly wallet driven payment companies. This growth has been driven by rapid adoption led by the increasing use of smartphones, tablets and speedy access to internet through broadband, 3G etc

## 10. Social networking

Social networking is the use of internet-based social media programs to make connections with friends, family, classmates, customers and clients. Social networking can be done for social purposes, business purposes or both. The programs show the associations between individuals and facilitate the acquisition of new contacts. Examples of social networking have included Facebook, LinkedIn, Classmates.com and Yelp.

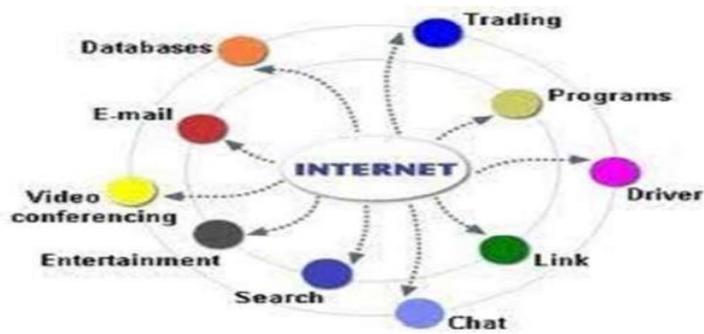


Image 41: Internet Applications  
Reference: <https://www.eucba.com/what-is-internet-application//>

## Introduction to HTML

### What is HTML?

HTML stands for Hyper Text Markup Language, is the standard markup language used to create web pages. Along with CSS, and JavaScript, HTML is a cornerstone technology used to create web pages, as well as to create user interfaces for mobile and web applications.

- HTML stands for Hyper Text Markup Language
- HTML elements are the building blocks of HTML pages
- HTML describes the structure of Web pages using HyperText markup language
- HTML elements are represented by tags
- When you test your page on browsers. Browsers do not display the HTML tags, but use them to render the content of the page.
- HTML page extension always will be .html (example.html)

"Hypertext" refers to the hyperlinks that an HTML page may contain. "Markup language" refers to the way tags are used to define the page layout and elements within the page.

## Structure of an HTML document

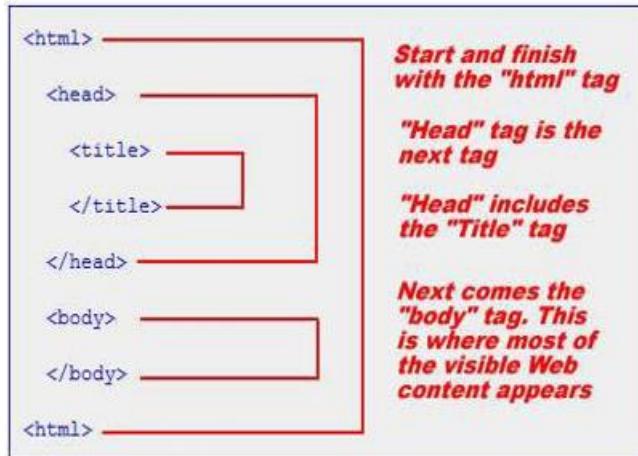


Image 42: Structure of an HTML  
Reference: <https://openlab.citytech.cuny.edu/clarkeadv2450/html/>

<DOCTYPE! html>:

This tag is used to tell the HTML version. This currently tells that the version is HTML 5.

<html>:

This is called the HTML root element and used to wrap all the code.

<head>:

Head tag contains metadata, title, page CSS etc. All the HTML elements that can be used inside the <head> elements are:

- <style >
- <title>
- <base>
- <noscript>
- <script >
- <meta>

<body>:

Body tag is used to enclose all the data which a web page has from texts to links. All of the content that you see rendered in the browser is contained within this element.

---

Example: HTML page can be created using any text editor (notepad). Then save that file using .htm or .html extension and open that file in the browser. It will get the HTML page response.

<meta>

The meta tag contains information for search engines, telling them what the page is about

<title>

The title tag is just that, the title that appears at the top of browser window and what first appears on search engine results

## **HTML Tags and Attributes**

### **HTML Elements**

Elements are the fundamentals of HyperText Markup Language (HTML). Each HTML document is made of elements that are specified using tags.

HTML elements and HTML tags are often confused. The tags are used to open and close the object, whereas the element includes both tags and its content. Let's consider an example with the <h1> tag: <h1> Title of the document </h1> - is an element, and <h1>, </h1> - are tags.

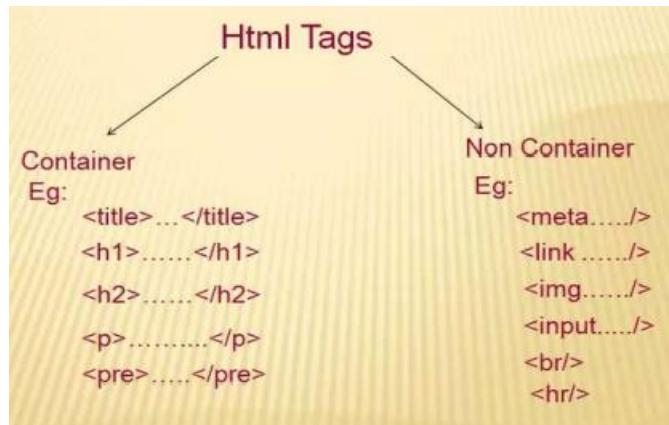


Image 43: HTML Elements  
Reference: <https://www.phptpoint.com/html-tags/>

### **Types of HTML Elements**

- empty elements

The empty elements have no closing tags. In XHTML the empty elements are "closed" in the opening tag like this: <br/> .

The empty elements in HTML are: <area>, <base>, <br>, <col>, <embed>, <hr>, <img>, <input>, <keygen>, <link>, <meta>, <param>, <source>, <track> and <wbr>.

<!DOCTYPE html>

Source:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Title of the document</title>
  </head>
  <body>
    <h1>Title of the document</h1>
    <p>The first paragraph</p>
    <p>The second paragraph, <br/> where line break is inserted </p>
  </body>
</html>
```

Output:

## Title of the document

The first paragraph

The second paragraph,  
where line break is inserted

### b) Block-level elements

For the purpose of styling, all HTML elements are divided into two categories: block-level elements and inline elements.

Block-level elements are those that structure the main part of your web page, by dividing your content into coherent blocks. They always start on a new line and take up the full width of a page, from left to right; they also can take up one line or multiple lines and have a line break before and after the element. Block-level elements can include both block-level & inline elements.

Block-level elements are <address>, <article>, <aside>, <blockquote>, <canvas>, <dd>, <div>,

<dl>, <dt>, <fieldset>, <figcaption>, <figure>, <footer>, <form>, <hl> - <h6>, <header>, <hr>, <li>, <main>, <nav>, <noscript>, <ol>, <output>, <p>, <pre>, <section>, <table>, <tfoot>, <ul> and <video>.

All block-level elements have opening and closing tags.

Source:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Title of the document</title>
  </head>
  <body>
    <footer>
      <p>Author: W3docs team</p>
      <p><a href="mailto:info@w3docs.com">Send message to the author</a>.</p>
    </footer>
  </body>
</html>
```

Output:



c) Inline elements

---

Inline elements are used to distinguish part of a text, to give it a particular function or meaning. Inline elements usually include a single or a few words. They do not cause a line break and do not take up the full width of a page, only the space bounded by its opening and closing tag. The inline elements are usually used within other HTML elements.

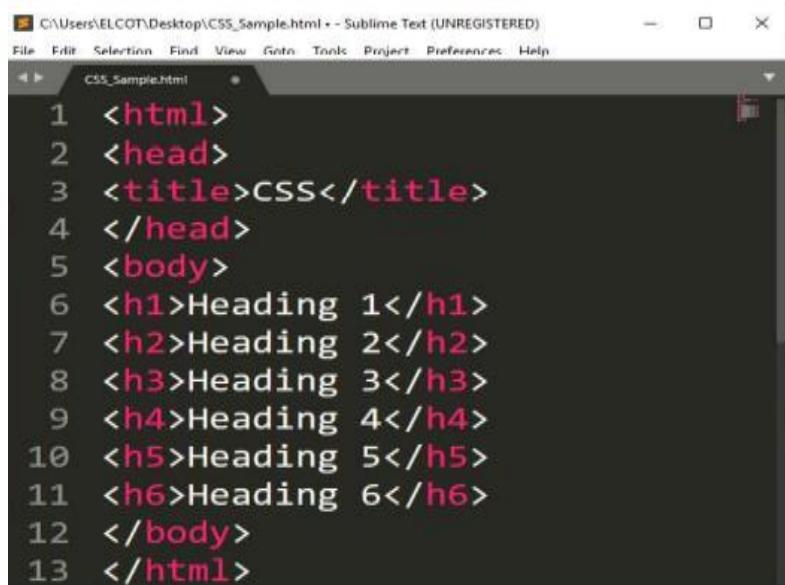
Inline elements are: `<a>`, `<abbr>`, `<acronym>`, `<b>`, `<bdo>`, `<big>`, `<br>`, `<button>`, `<cite>`, `<code>`, `<dfn>`, `<em>`, `<i>`, `<img>`, `<input>`, `<kbd>`, `<label>`, `<map>`, `<object>`, `<q>`, `<samp>`, `<script>`, `<select>`, `<small>`, `<span>`, `<strong>`, `<sub>`, `<sup>`, `<textarea>`, `<time>`, `<tt>` and `<var>`.

## Headings

Headings are defined with the `<h1>` to `<h6>` tags.

`<h1>` defines the most important heading. `<h6>` defines the least important heading.

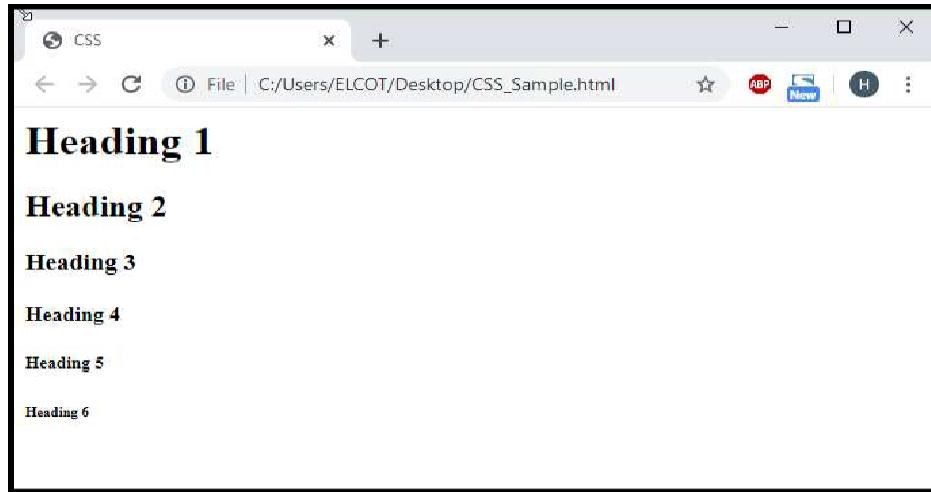
Source:



The screenshot shows a Sublime Text editor window with the title bar "C:\Users\ELCOTT\Desktop\CSS\_Sample.html - Sublime Text (UNREGISTERED)". The menu bar includes File, Edit, Selection, Find, View, Goto, Tools, Project, Preferences, and Help. The main editor area displays the following HTML code:

```
1 <html>
2 <head>
3 <title>CSS</title>
4 </head>
5 <body>
6 <h1>Heading 1</h1>
7 <h2>Heading 2</h2>
8 <h3>Heading 3</h3>
9 <h4>Heading 4</h4>
10 <h5>Heading 5</h5>
11 <h6>Heading 6</h6>
12 </body>
13 </html>
```

Output:



## Formatting

HTML also defines special elements for defining text with a special meaning. The formatting tags are divided into two categories:

1. Physical tags, used for text styling (visual appearance of the text)
2. Logical or semantic tags used to add semantic value to the parts of the text

Formatting includes,

- **<b>** - Bold text
- **<strong>** - Important text
- **<i>** - Italic text
- **<em>** - Emphasized text
- **<mark>** - Marked text
- **<small>** - Small text
- **<del>** - Deleted text
- **<ins>** - Inserted text
- **<sub>** - Subscript text
- **<sup>** - Superscript text

HTML uses elements like **<b>** and **<i>** for formatting output, like bold or italic text. Formatting elements were designed to display special types of text:

---

### <b> and <strong> Tags

The <b> tag is a physical tag that stands for bold text, whereas the <strong> tag being a logical tag is used to emphasize the importance of the text.

<b>This text is bold</b>

<strong>This text is strong</strong>

### <i> and <em> Tags

The <i> and <em> tags define italic text. The <i> tag is only responsible for visual appearance of the enclosed text, without any extra importance. The <em> tag defines emphasized text, with added semantic importance.

<i>This text is italic</i>

<em>This text is emphasized</em>

### <pre> Tag

The <pre> tag is used to define preformatted text. The browsers render the enclosed text with white spaces and line breaks.

<pre>Spaces

and line breaks

within this element

are shown as typed.

</pre>

### <mark> Tag

The <mark> tag is used to present a part of text in one document as marked or highlighted for reference purposes.

<h2>HTML <mark> Marked</mark> Formatting</h2>

### <small> Tag

The <small> tag decreases the text font size by one size smaller than a document's base font size (from medium to small, or from x-large to large).

---

The `<p>` tag usually contains the items of secondary importance such as copyright notices, side comments, or legal notices.

`<p>The interest rate is only 10%*``</p>`

`<small>*` per day`</small>`

`<del>` and `<s>` Tags

The `<del>` tag specifies a part of the text that was deleted from the document. Browsers display this text as a strikethrough.

The `<s>` tag defines text that is no longer correct, or relevant. The content of both tags is displayed as strikethrough. However, despite the visual similarity, these tags cannot replace each other.

`<ins>` and `<u>` Tags

The `<ins>` tag defines the text that has been inserted (added) to the document. The content of the tag is displayed as underlined.

The `<u>` tag specifies text that is stylistically different from normal text, i.e., words or fragments of text that need to be presented differently. This could be misspelled words or proper nouns in Chinese.

`<p> She likes <del >violets </del> snowdrops. </p>`

`<p>She likes <del >violets </del> <ins >snowdrops</ins>. </p>`

`<sub >` and `<sup >` Tags

The `<sub>` defines subscript texts. Subscript text is under the baseline of other symbols of the line and has smaller font. The `<sup>` tag defines superscript, that is set slightly above the normal line of type and is relatively smaller than the rest of the text. The baseline passes through the upper or lower edge of the symbols.

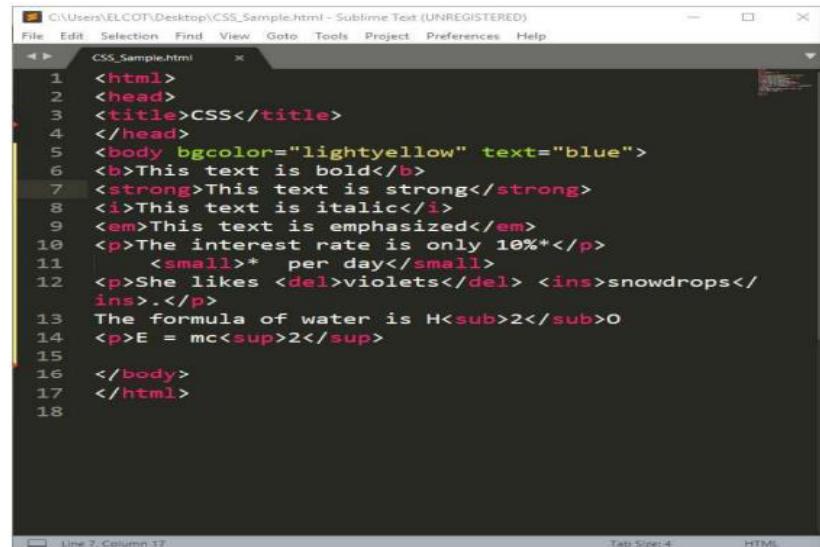
`<p>The formula of water is H<sub >2</sub>O, and the formula of alcohol is C<sub>2</sub>H<sub>5</sub>OH </p>`

`<p>E = mc<sup >2</sup>, where E-kinetic energy, m- mass, c- the speed of light. </p>`

`<p>, <br>` and `<hr>` Tags

The `<p>` tag defines the paragraph. Browsers automatically add 1 em margin before and after each paragraph.

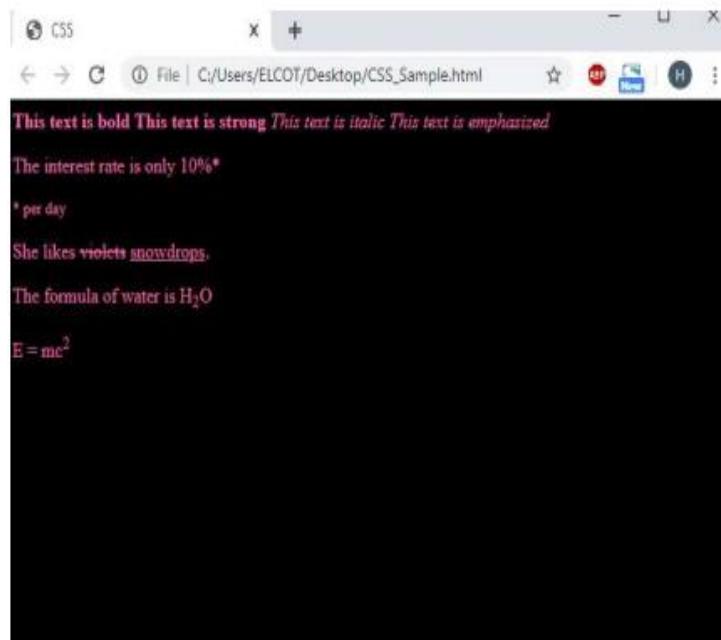
Source:



The screenshot shows a Sublime Text editor window with the file 'CSS\_Sample.html' open. The code contains various HTML tags and their attributes, demonstrating styling like bold, strong, italic, em, and small text. The code is as follows:

```
1 <html>
2 <head>
3 <title>CSS</title>
4 </head>
5 <body bgcolor="lightyellow" text="blue">
6 <b>This text is bold</b>
7 <strong>This text is strong</strong>
8 <i>This text is italic</i>
9 <em>This text is emphasized</em>
10 <p>The interest rate is only 10%*</p>
11 <small>* per day</small>
12 <p>She likes <del>violets</del> <ins>snowdrops</ins>.</p>
13 The formula of water is H<sub>2</sub>O
14 <math>E = mc^2</math>
15
16 </body>
17 </html>
18
```

## Output:



## HTML Colors

There are three ways of how you can change the color of the text in HTML:

1. Hex color codes
2. HTML color names

---

### 3. RGB values

#### **HTML Color Names**

To color the text element using an HTML color name, put the name of the color (blue, for ex.) instead of Hex code from the previous step.

```
<p style="color:red;"> This is a text in green</p>
```

#### **Hex Color Codes**

A hex color code is a hex triplet, which represents three separate values defining the levels of the component colors. It is specified with a hexadecimal (hex) notation for a mixture of red, green, and blue color values. The lowest value that can be given to one of the light sources is 0 (hex 00). The highest value is 255 (hex FF).

Hex values are written as six-digit numbers, starting with a# sign. Letters used in a hexadecimal digit may be uppercase or lowercase. For example, to specify white color you can write #FFFFFF or #ffffff.

```
<p style="color:#8ebf42;"> This is a text in green</p>
```

#### **RGB Color Values**

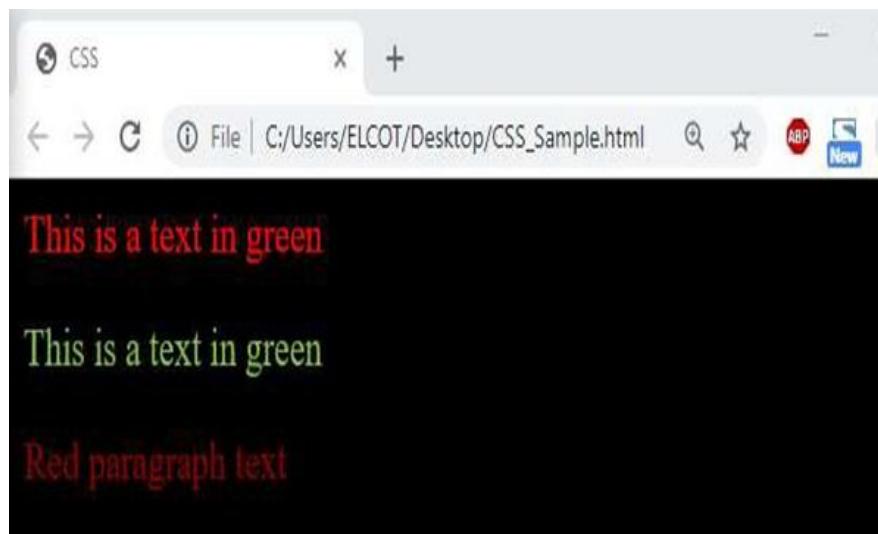
To add a color to the text element, use the style attribute (where the color property is your RGB value)

```
<p style="color:rgba(255,0,0,0.5);>Red paragraph text</p>
```

Source:

```
File Edit Selection Find View Goto Tools Project Preferences Help
CSS_Sample.html x
1 <html>
2 <head>
3 <title>CSS</title>
4 </head>
5 <body bgcolor="black" text="#FF1493">
6 <p style="color:red;"> This is a text in green</p>
7
8 <p style="color:#8ebf42;"> This is a text in green</p>
9
10 <p style="color:rgba(255,0,0,0.5);">Red paragraph text</p>
11
12
13 </body>
14 </html>
```

Output:



## HTML Image

- The `<img>` tag is empty, it contains attributes only, and does not have a closing tag.
- The `src` attribute specifies the URL (web address) of the image: ``

## Attributes of `<img>` tag

Attributes	Description
Vspace	Specifies the amount of space to the top and bottom of the image.
Hspace	Specifies the amount of space to the left and right of the image.
Alt	Specifies alternate text for an image when image is not found.

## Images in Another Folder

If not specified, the browser expects to find the image in the same folder as the web page.

However, it is common to store images in a sub-folder. You must then include the folder name in the src attribute: ``

## Images on Another Server

Some web sites store their images on image servers.

Actually, you can access images from any web address in the world: ``

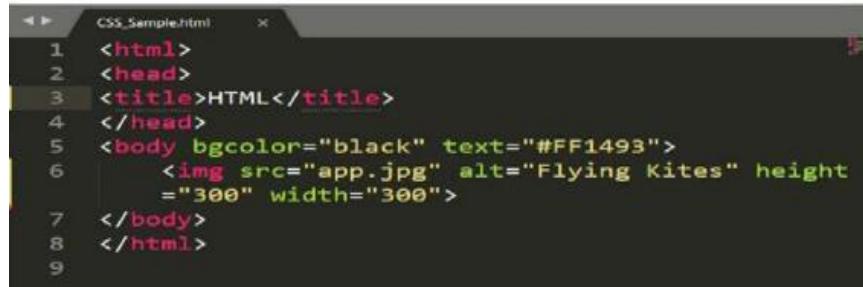
## Animated Images

HTML allows animated GIFs: ``

External Links (aka backlinks or inbound links)	Internal Links
Difficult to Control	Easy, fast and free to create
Pass SEO authority from other sites to your site, increasing your "domain authority"	Pass SEO authority between pages on your site, increasing the "page authority" of specific pages
Appear within the body text, in content	Appear in website navigation, as well as in the content

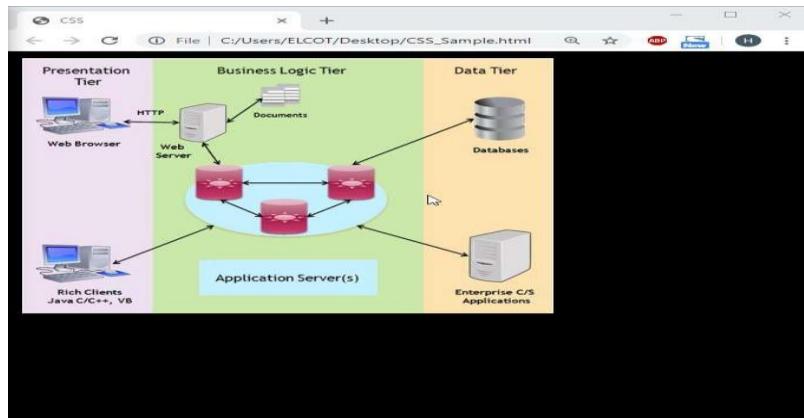
Image 44: External and internal links  
Reference: <https://www.orbitmedia.com/blog/internal-linking/>

Source:



```
1 <html>
2 <head>
3 <title>HTML</title>
4 </head>
5 <body bgcolor="black" text="#FF1493">
6   
7 </body>
8 </html>
9
```

Output:



## LINKING

**HTML links are hyperlinks.**

You can click on a link and jump to another document.

---

When you move the mouse over a link, the mouse arrow will turn into a little hand. The href attribute specifies the destination address of the link.

<a href=" [https://www.w3schools.com/html/html\\_basics.asp](https://www.w3schools.com/html/html_basics.asp)">HTML tutorial</a>

ALINK: It indicates the colour of the active hyperlink. An active link is the one on which the mouse button is pressed. e. VLINK: It indicates the colour of the hyperlinks after the mouse is clicked on it.

### **HTML Links - The target Attribute**

The target attribute specifies where to open the linked document. The target attribute can have one of the following values:

- \_blank - Opens the linked document in a new window or tab
- \_self - Opens the linked document in the same window/tab as it was clicked (default)
- \_parent - Opens the linked document in the parent frame
- \_top - Opens the linked document in the full body of the window

<a href=" <https://www.w3schools.com/>" target="\_blank">Visit W3Schools! </a>

### **Local Links**

The example above used an absolute URL (a).

A local link (link to the same web site) is specified with a relative URL (without https://www).

#### **Absolute URL -full web address**

<a href=" [https://www.w3schools.com/html/html\\_basics.asp](https://www.w3schools.com/html/html_basics.asp)">HTML tutorial</a>

**Relative URL (without https://www )**.

<a href=" html\_images.asp">HTML Images </a>

### **HTML Links - Image as a Link**

<a href=" default.asp">



<1 a>

Source:

```
File Edit Selection Find View Goto Tools Project Preferences Help
CSS_Sample.html ×
1 <html>
2 <head>
3 <title>HTML</title>
4 </head>
5 <body bgcolor="black" text="#FF1493">
6
7 <h1>Link</h1>
8 <a href="http://www.google.com">Google</a>
9 <br/>
10 <h1>Image as a Link</h1>
11 <a href="default.asp">
12 
13 </a>
14 </body>
15 </html>
16
```

Output:



## Tables

- An HTML table is defined with the `<table>` tag. Each table row is defined with the `<tr>` tag.
- A table header is defined with the `<th>` tag.
- By default, table headings are bold and centered. A table data/cell is defined with the `<td>` tag.

## Adding Captions to Tables

You can use the <caption> element to specify a caption for tables. It should be placed immediately after the opening <table>

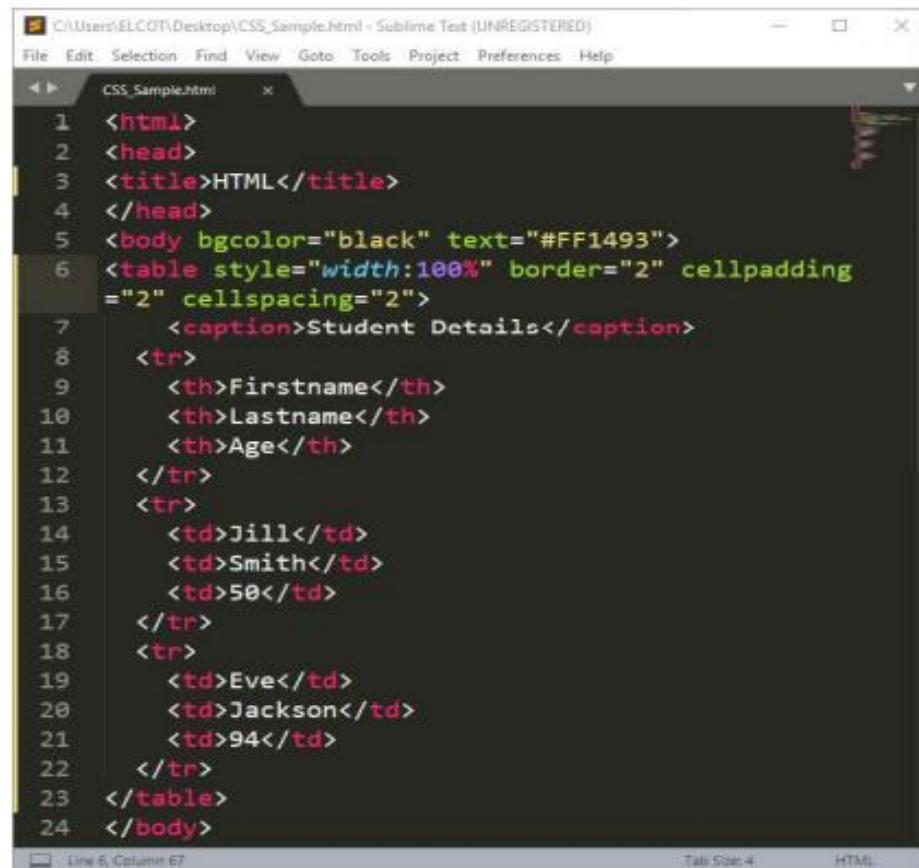
```
<caption>Student Details</caption>
```



Cell Spacing is used to set space between different table cells.

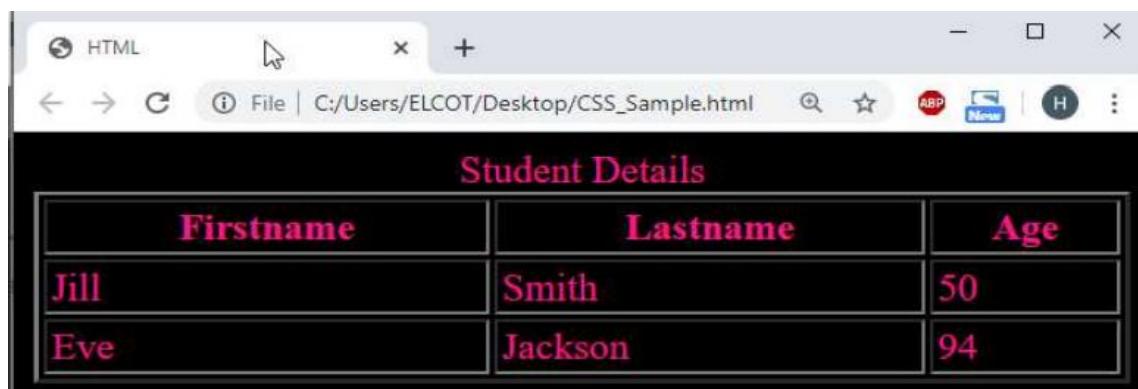
Cell Padding is used for the space between the edges of the cell and the content of the cell.

Source:



```
<html>
<head>
<title>HTML</title>
</head>
<body bgcolor="black" text="#FF1493">
<table style="width:100%" border="2" cellpadding="2" cellspacing="2">
    <caption>Student Details</caption>
    <tr>
        <th>Firstname</th>
        <th>Lastname</th>
        <th>Age</th>
    </tr>
    <tr>
        <td>Jill</td>
        <td>Smith</td>
        <td>50</td>
    </tr>
    <tr>
        <td>Eve</td>
        <td>Jackson</td>
        <td>94</td>
    </tr>
</table>
</body>
```

Output:

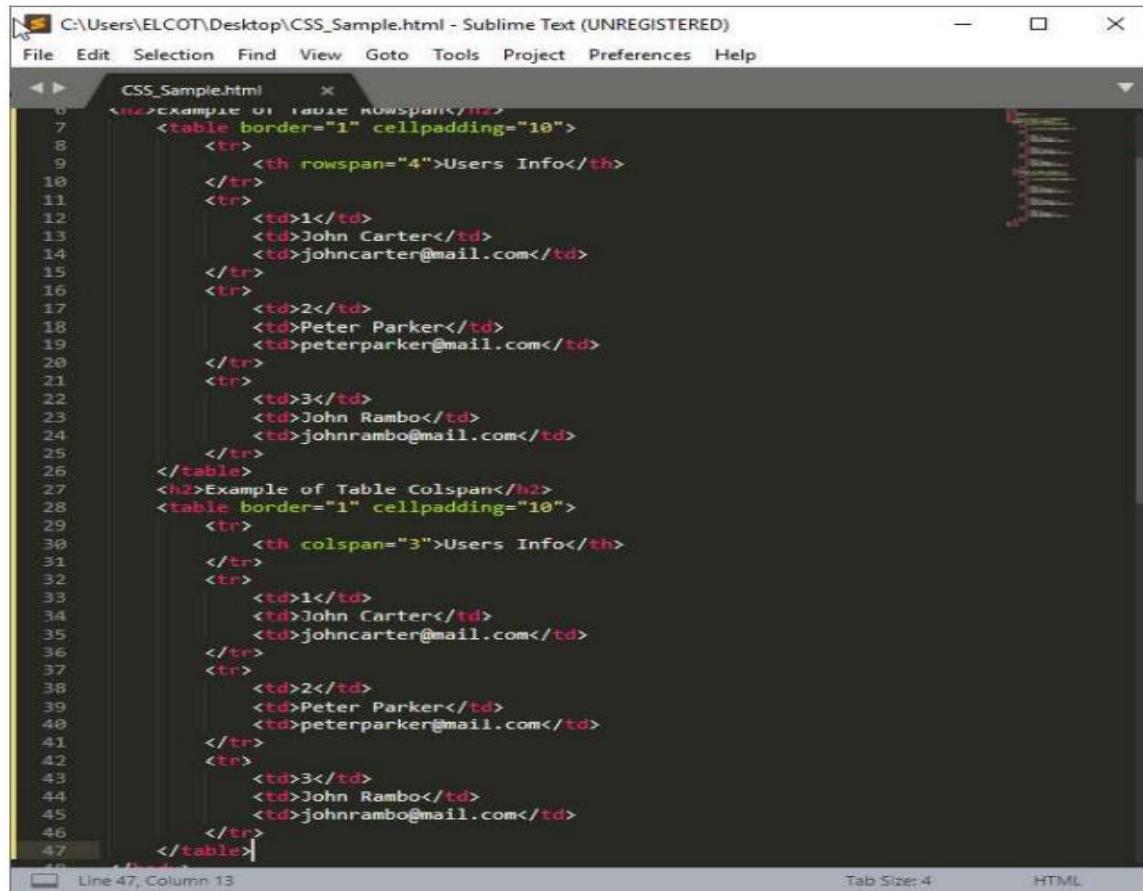


Firstname	Lastname	Age
Jill	Smith	50
Eve	Jackson	94

## Rowspan and Colspan

The rowspan and colspan are `<td>` tag attributes. These are used to specify the number of rows or columns a cell should span. The rowspan attribute is for rows as well as the colspan attribute is for columns.

Source:



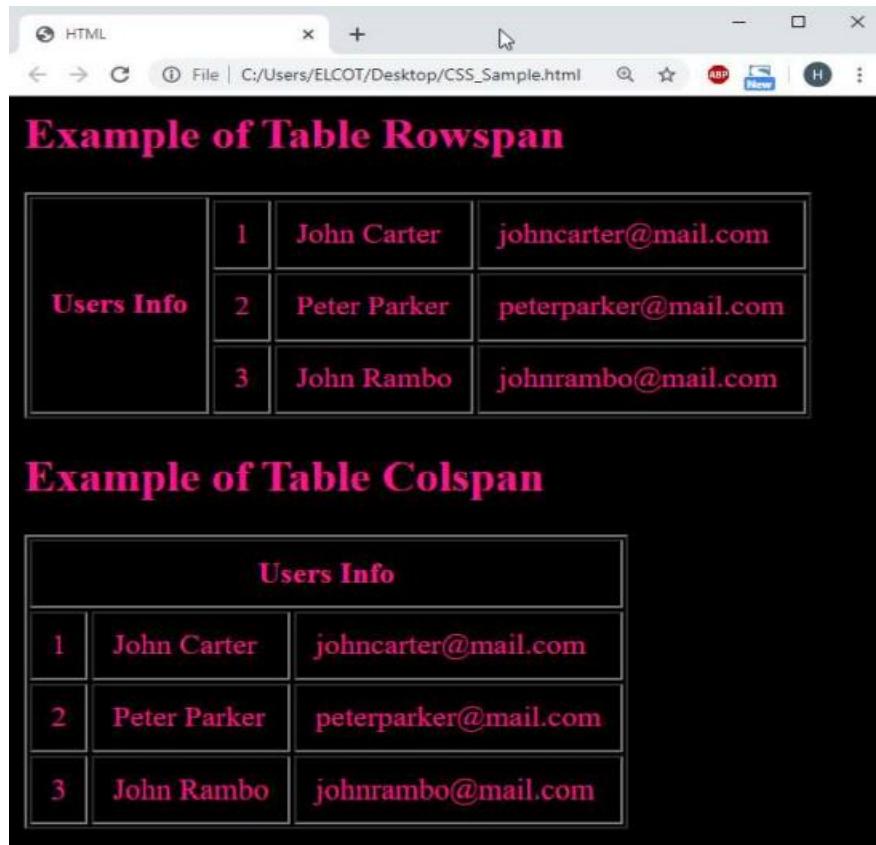
The screenshot shows a Sublime Text editor window with the following code:

```
C:\Users\ELCOT\Desktop\CSS_Sample.html - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
CSS_Sample.html
<h2>Example of Table Rowspan</h2>
<table border="1" cellpadding="10">
  <tr>
    <th rowspan="4">Users Info</th>
  </tr>
  <tr>
    <td>1</td>
    <td>John Carter</td>
    <td>johncarter@mail.com</td>
  </tr>
  <tr>
    <td>2</td>
    <td>Peter Parker</td>
    <td>peterparker@mail.com</td>
  </tr>
  <tr>
    <td>3</td>
    <td>John Rambo</td>
    <td>johnrambo@mail.com</td>
  </tr>
</table>
<h2>Example of Table Colspan</h2>
<table border="1" cellpadding="10">
  <tr>
    <th colspan="3">Users Info</th>
  </tr>
  <tr>
    <td>1</td>
    <td>John Carter</td>
    <td>johncarter@mail.com</td>
  </tr>
  <tr>
    <td>2</td>
    <td>Peter Parker</td>
    <td>peterparker@mail.com</td>
  </tr>
  <tr>
    <td>3</td>
    <td>John Rambo</td>
    <td>johnrambo@mail.com</td>
  </tr>
</table>

```

The code defines two tables. The first table has a single header cell with `rowspan="4"` that spans four rows. The second table has a single header cell with `colspan="3"` that spans three columns.

Output:



The screenshot shows a web browser window with the title 'HTML'. The address bar indicates the file is located at 'C:/Users/ELCOT/Desktop/CSS\_Sample.html'. The content of the page is as follows:

### Example of Table Rowspan

Users Info	1	John Carter	johncarter@mail.com
	2	Peter Parker	peterparker@mail.com
	3	John Rambo	johnrambo@mail.com

### Example of Table Colspan

Users Info		
1	John Carter	johncarter@mail.com
2	Peter Parker	peterparker@mail.com
3	John Rambo	johnrambo@mail.com

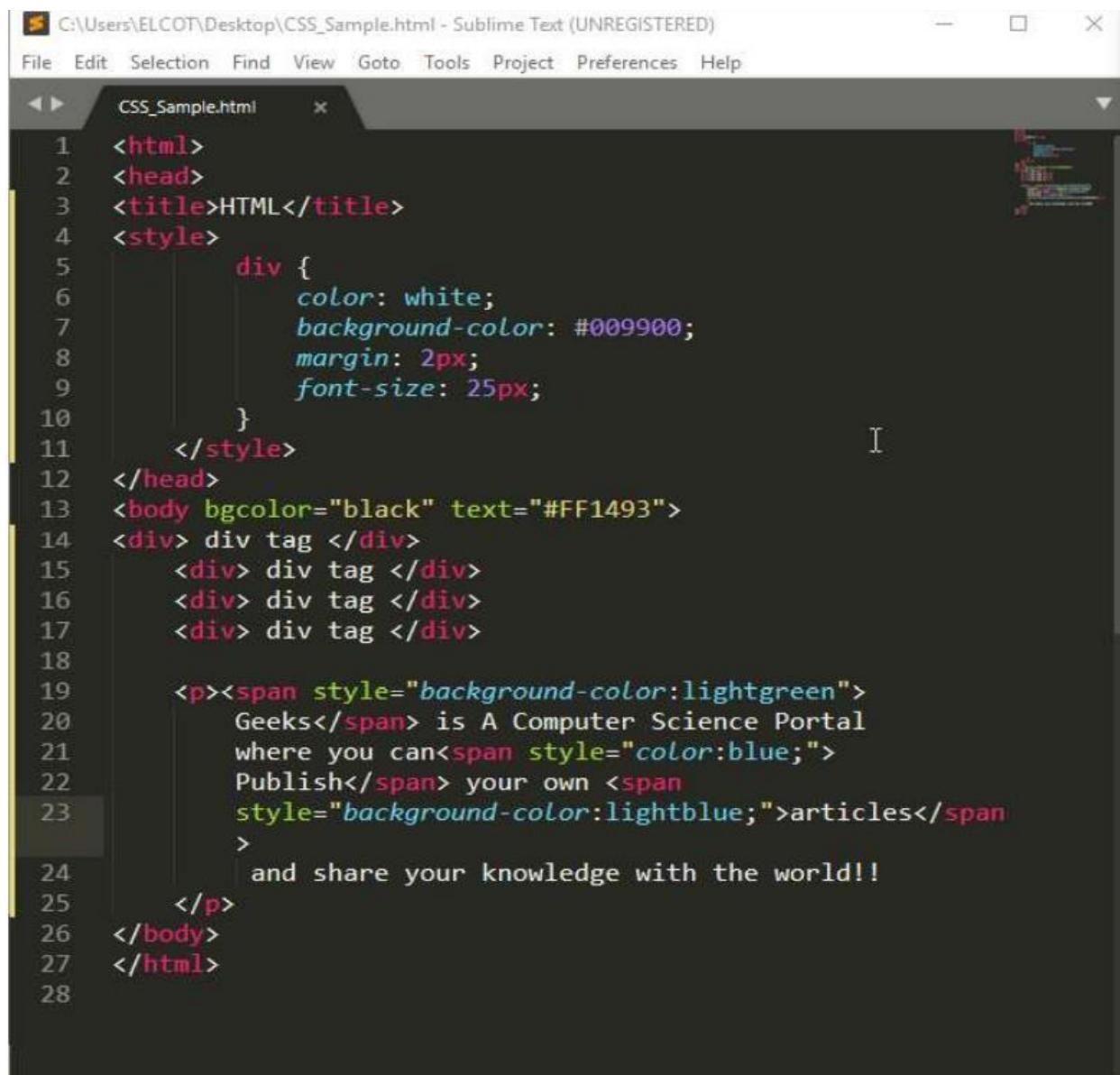
## Div and Span

<div> tag is used as a block part of the webpage

<span> tag is used as an inline part of the webpage

The div should be used to wrap sections of a document, while use spans to wrap small portions of text, images, etc. The <div> element is used while creating CSS based layouts in html, whereas <span> element is used to stylize texts.

Source:

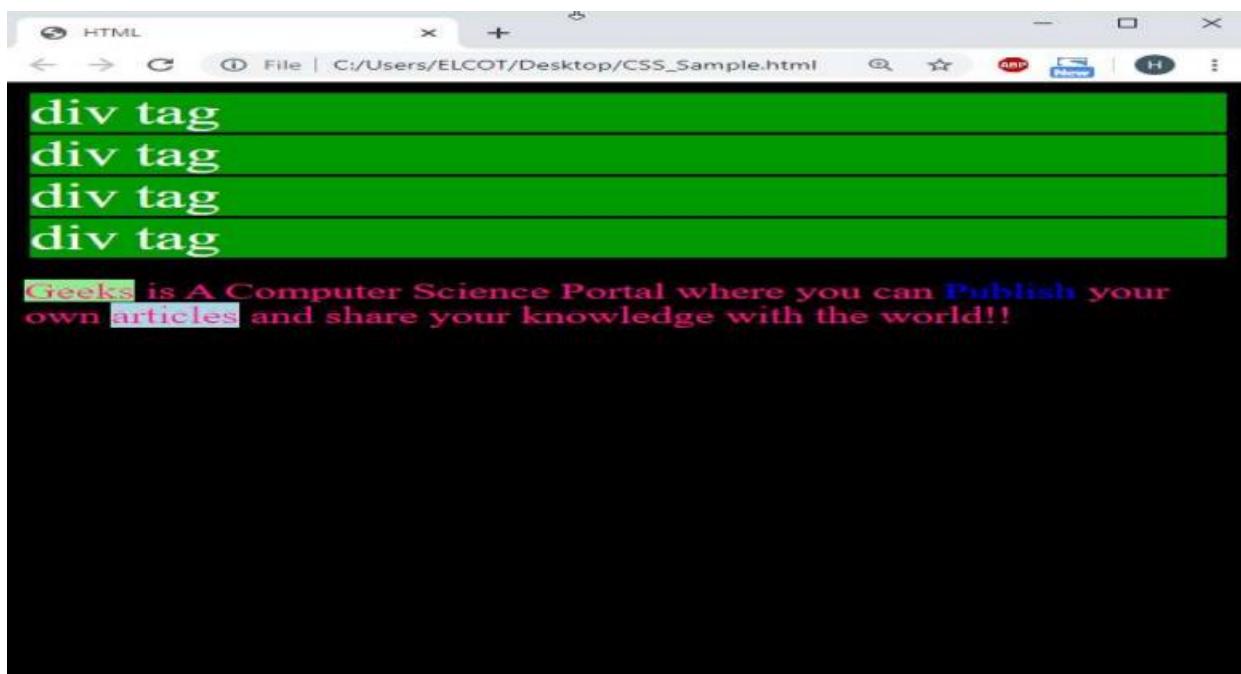


The screenshot shows a Sublime Text editor window with the following details:

- Title Bar:** C:\Users\ELCOT\Desktop\CSS\_Sample.html - Sublime Text (UNREGISTERED)
- Menu Bar:** File Edit Selection Find View Goto Tools Project Preferences Help
- File List:** CSS\_Sample.html
- Code Content:** An HTML file with embedded CSS. The code is as follows:

```
1 <html>
2 <head>
3 <title>HTML</title>
4 <style>
5     div {
6         color: white;
7         background-color: #009900;
8         margin: 2px;
9         font-size: 25px;
10    }
11   </style>
12 </head>
13 <body bgcolor="black" text="#FF1493">
14 <div> div tag </div>
15   <div> div tag </div>
16   <div> div tag </div>
17   <div> div tag </div>
18
19   <p><span style="background-color:lightgreen">
20     Geeks</span> is A Computer Science Portal
21     where you can<span style="color:blue;">
22       Publish</span> your own <span
23         style="background-color:lightblue;">articles</span
24       >
25     and share your knowledge with the world!!
26   </p>
27 </body>
28 </html>
```

Output:



<DIV>	<SPAN>
The <div> tag is a block level element.	The <span> tag is an inline element.
It is best to attach it to a section of a web page.	It is best to attach a CSS to a small section of a line in a web page.
It accepts align attribute.	It does not accept align attribute.
This tag should be used to wrap a section, for highlighting that section.	This tag should be used to wrap any specific word that you want to highlight in your webpage.

Image 44: External and internal links  
Reference: <https://www.geeksforgeeks.org/difference-between-div-and-span-tag-in-html/>

## Lists

HTML List Tags are used to specify information in the form of list.

HTML Lists are very useful to group related information together. Often List items looks well-structured and they are easy to read for users. A list can contain one or more list elements. HTML

---

## Type of Lists

- Unordered HTML List
- Ordered HTML List
- Description Lists
- Nested HTML Lists

### Ordered lists

Ordered list is used to list related items in a numbered or other specific order. This is useful when you want to show counts of items in some way.

Ordered list is created using the HTML `<ol>` tag. Each item in the list start with the `<li>` tag

Example of Ordered List

```
<ul>  
<li>Red</li>  
<li>Green </li>  
<li>Blue</li>  
</ul>
```

Above example will list colors items with numbers by default. There are different list style available for ordered lists such as numbers, letters etc.

#### Ordered List Style Type Attribute

There are two attributes can be used to customize ordered list, they are

1. Type - changing numbering style
2. Start - changing numbering order

Type - is used to change the number style. The default number style is standard Arabic numerals (1,2,3, ...).

Start - is used to specify the number of letters with which start the list. The default starting point is 1. The value of the start attribute should be a decimal number, regardless of the numbering style being used.

## IHTML Ordered List Style Type Attribute

List Style Type	Description	Example and Syntax
Numbers	Starts a list using numbers (default)	<ol type="1">
uppercase letters	Starts a list using uppercase letters	<ol type="A">
Lowercase letters	Starts a list using lowercase letters	<ol type="a">
Uppercase roman numbers	Starts a list using uppercase roman numbers	<ol type="I">
Lowercase roman numbers	Starts a list using lowercase roman numbers	<ol type="i">

```
<ol type="1">
<li>Banana</li>
<li>Apple</li>
<li>Grapes </li> </ol>
```

### Unordered lists

Unordered lists are used to list sets of items when they have no special order or sequence. It is also called a bulleted list.

Unordered list is created using the HTML `<ul>` tag. Each item in the list start with the `<li>` tag

### Unordered List Style Type Attribute

Like an ordered list, type attribute is used to customize bullet style for the list of elements. By default, a solid circle is used as bullets.

1. Type value: Numbering style
2. Disc: A solid circle

3. Square: A solid square
4. Circle: An unfilled circle

## HTML List Types

List Style Type	Description	Example & Syntax
Disc	Starts a list using discs type bullets (default)	<code>&lt;ul type="disc"&gt;</code>
Circle	Starts a list using circle type bullets	<code>&lt;ul type="circle"&gt;</code>
square	Starts a list using square type bullets	<code>&lt;ul type="square"&gt;</code>
None	Starts a list without bullets	<code>&lt;ul type="type:none"&gt;</code>

```
<ul type="disc">
  <li>Apple</li>
  <li>Banana</li>
  <li>Mango</li>
</ul>
```

## Definition Lists

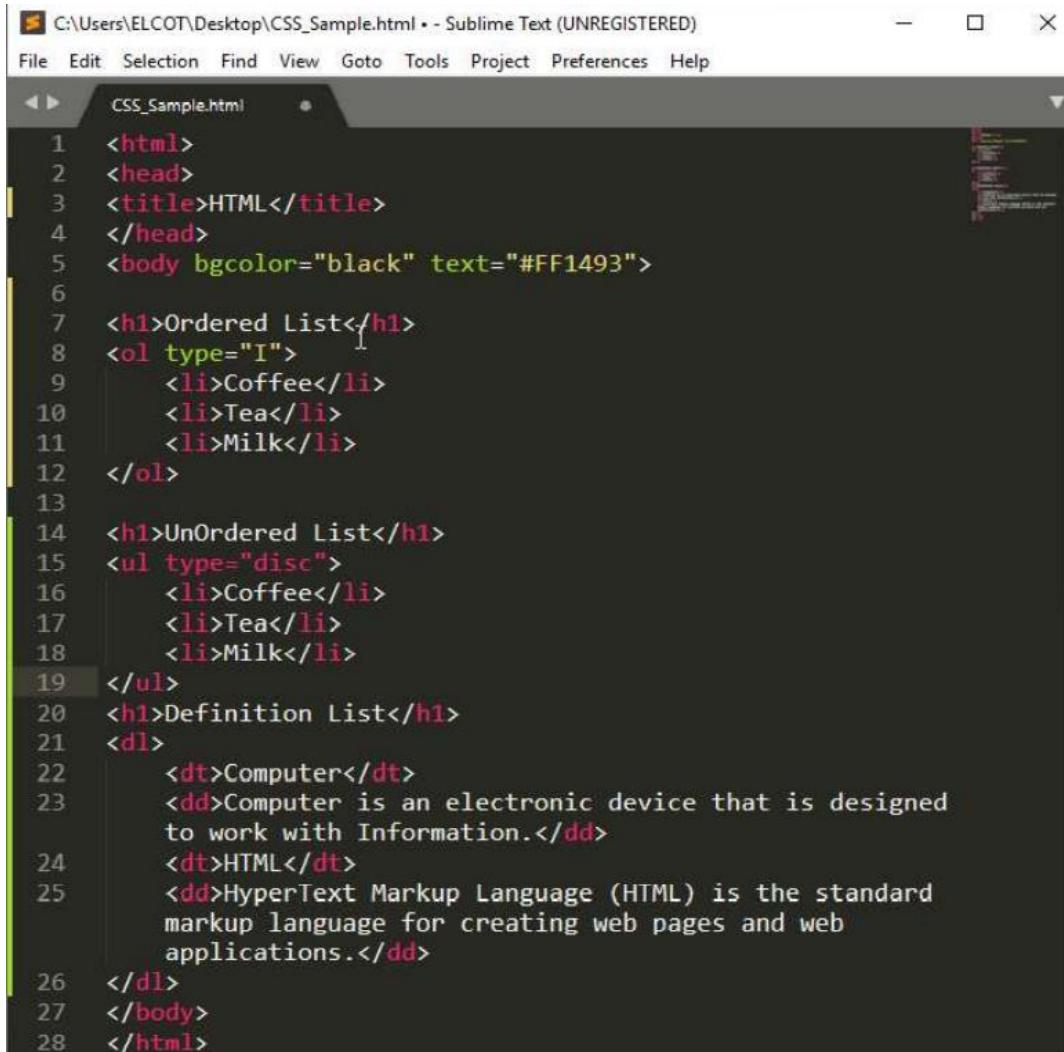
Definition list is different from the other two types of list. No bullet or number is provided for the list items. In this list type, the list element has two parts.

1. A definition terms
2. The definition description
  - Definition list is surrounded within `<DL>.....</DL>` tags.
  - Definition term is presented in between `<DT>.....<IDT>` tag and
  - Definition description should be surrounded within `<DD>.....</DD>` tag.

```
<dl>
  <dt>Computer </dt>
```

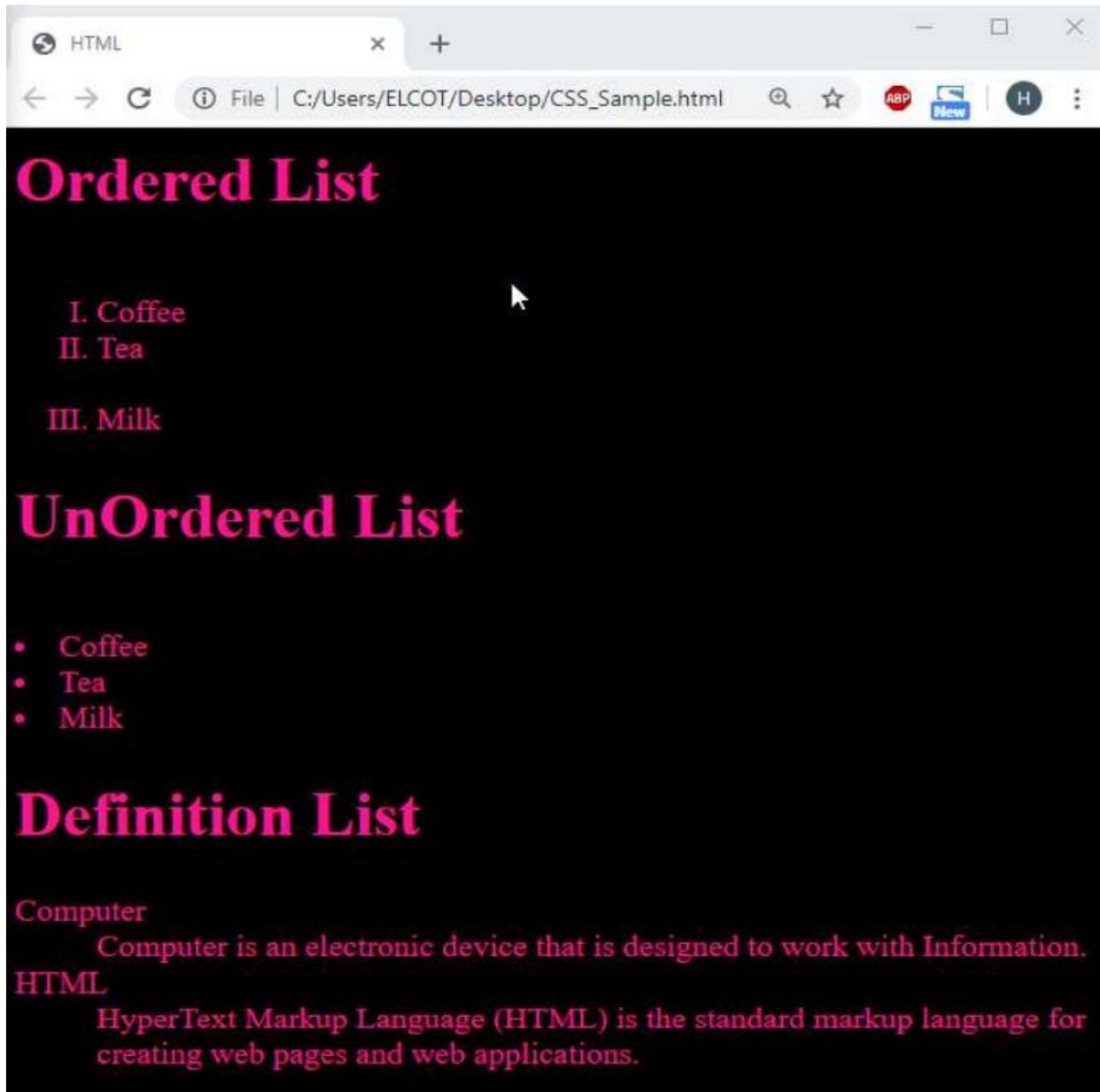
<dd>Computer is an electronic device that is designed to work with Information. </dd>  
</dl>

Source:



```
C:\Users\ELCOT\Desktop\CSS_Sample.html - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
CSS_Sample.html
1 <html>
2 <head>
3 <title>HTML</title>
4 </head>
5 <body bgcolor="black" text="#FF1493">
6
7 <h1>Ordered List</h1>
8 <ol type="I">
9   <li>Coffee</li>
10  <li>Tea</li>
11  <li>Milk</li>
12 </ol>
13
14 <h1>UnOrdered List</h1>
15 <ul type="disc">
16   <li>Coffee</li>
17   <li>Tea</li>
18   <li>Milk</li>
19 </ul>
20 <h1>Definition List</h1>
21 <dl>
22   <dt>Computer</dt>
23   <dd>Computer is an electronic device that is designed
24     to work with Information.</dd>
25   <dt>HTML</dt>
26   <dd>HyperText Markup Language (HTML) is the standard
27     markup language for creating web pages and web
28     applications.</dd>
29 </dl>
30 </body>
31 </html>
```

Output:



The screenshot shows a web browser window with the title 'HTML'. The address bar indicates the file is located at 'C:/Users/ELCOT/Desktop/CSS\_Sample.html'. The page content is as follows:

## Ordered List

- I. Coffee
- II. Tea
- III. Milk

## UnOrdered List

- Coffee
- Tea
- Milk

## Definition List

**Computer**  
Computer is an electronic device that is designed to work with Information.

**HTML**  
HyperText Markup Language (HTML) is the standard markup language for creating web pages and web applications.

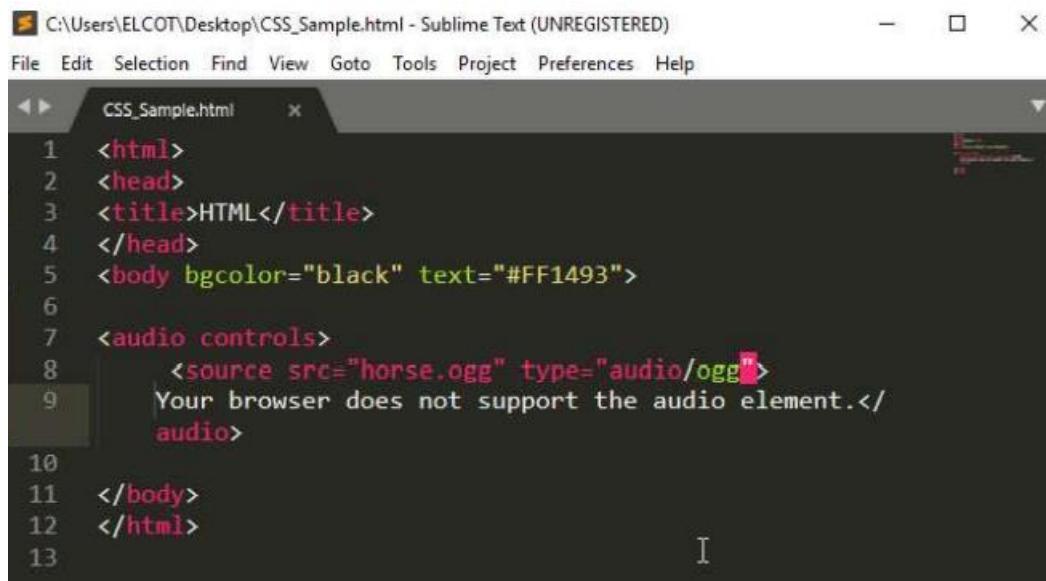
## Audio and Videos

- <audio> element specifies a standard way to embed audio in a web page.

The HTML audio tag that used to define sounds such as music and other audio clips are:

- a) mp3
- b) wav
- c) ogg

Source:



```
C:\Users\ELCOT\Desktop\CSS_Sample.html - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
CSS_Sample.html ×
1 <html>
2 <head>
3 <title>HTML</title>
4 </head>
5 <body bgcolor="black" text="#FF1493">
6
7 <audio controls>
8   <source src="horse.ogg" type="audio/ogg">
9   Your browser does not support the audio element.</
10  audio>
11 </body>
12 </html>
13
```

Output:



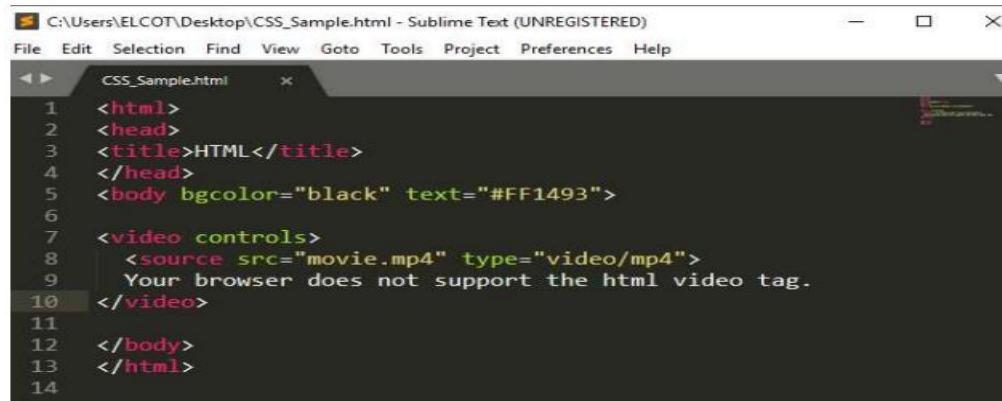
## Video

- <video> element specifies a standard way to embed a video in a web page
- The HTML video tag is used for streaming video files such as a movie clip, song clip on the web page.

Currently, there are three video formats supported for HTML video tag:

- a) mp4
- b) webM
- c) ogg

Source:



C:\Users\ELCOT\Desktop\CSS\_Sample.html - Sublime Text (UNREGISTERED)

```
File Edit Selection Find View Goto Tools Project Preferences Help
CSS_Sample.html x

1 <html>
2 <head>
3 <title>HTML</title>
4 </head>
5 <body bgcolor="black" text="#FF1493">
6
7 <video controls>
8   <source src="movie.mp4" type="video/mp4">
9   Your browser does not support the html video tag.
10 </video>
11
12 </body>
13 </html>
14
```

Output:



## HTML Forms and Input

An HTML form is a section of a document which contains controls such as text fields, password fields, checkboxes, radio buttons, submit button, menus etc.

An HTML form facilitates the user to enter data that is to be sent to the server for processing such as name, email address, password, phone number, etc.

- HTML Form Tags
- HTML <form> element

---

The HTML <form> element provide a document section to take input from user. It provides various interactive controls for submitting information to web server such as text field, text area, password field, etc.

```
<form>
//Form elements
</form>
```

Tag	Description
<form>	It defines an HTML form to enter inputs by the user side.
<input>	It defines an input control.
<textarea>	It defines a multi-line input control.
<label>	It defines a label for an input element.
<fieldset>	It groups the related element in a form.
<legend>	It defines a caption for a <fieldset> element.
<select>	It defines a drop-down list.
<optgroup>	It defines a group of related options in a drop-down list.
<option>	It defines an option in a drop-down list.
<button>	It defines a clickable button.

## HTML <input> element

The HTML <input> element is fundamental form element. It is used to create form fields, to take input from user. We can apply different input filed to gather different information form user.

```
<body>
<form>
Enter your name <br>
<input type = "-text" name="username" >
</form>
</body>
```

- Label Tag in Form

It is considered better to have label in form. As it makes the code parser/browser/user friendly.

If you click on the label tag, it will focus on the text control. To do so, you need to have for attribute in label tag that must be same as id attribute of input tag.

```
<form>
  <label for="firstname">First Name: </label> <br/>
    <input type="text" id="firstname" name="firstname"/> <br/>
  <label for="lastname">Last Name: </label>
    <input type="text" id="lastname" name="lastname"/> <br/>
</form>
```

**First Name:**

**Last Name:**

## Radio Button Control

The radio button is used to select one option from multiple options. It is used for selection of gender, quiz questions etc.

If you use one name for all the radio buttons, only one radio button can be selected at a time.

- Using radio buttons for multiple options, you can only choose a single option at a time.

```
<form>
  <label for="gender">Gender: </label>
  <input type="radio" id="gender" name="gender" value="male"/>Male
  <input type="radio" id="gender" name="gender" value="female"/>Female <br/>
</form>
```

Gender:  Male  Female

## Checkbox Control

The checkbox control is used to check multiple options from given checkboxes.

```
<form>
  Hobby:<br>
  <input type="checkbox" id="cricket" name="cricket" value="cricket"/>
  <label for="cricket">Cricket</label> <br>
  <input type="checkbox" id="football" name="football" value="football"/>
  <label for="football">Football</label> <br>
  <input type="checkbox" id="hockey" name="hockey" value="hockey"/>
  <label for="hockey">Hockey</label>
</form>
```

**Hobby:**  
 Cricket  
 Football  
 Hockey

## Submit button control

HTML `<input type="submit" >` are used to add a submit button on web page. When user clicks on submit button, then form get submit to the server

```
<form>
  <label for="name">Enter name</label><br>
  <input type="text" id="name" name="name"><br>
  <label for="pass">Enter Password</label><br>
  <input type="Password" id="pass" name="pass"><br>
  <input type="submit" value="submit">
</form>
```



HTML Form Example:

```
<form>
<fieldset>
  <legend>User personal information</legend>
  <label>Enter your full name</label><br>
  <input type="text" name="name"><br>
  <label>Enter your email</label><br>
  <input type="email" name="email"><br>
  <label>Enter your password</label><br>
  <input type="password" name="pass"><br>
  <label>confirm your password</label><br>
  <input type="password" name="pass"><br>
  <br><label>Enter your gender</label><br>
  <input type="radio" id="gender" name="gender" value="male"/>Male <br>
  <input type="radio" id="gender" name="gender" value="female"/>Female <br/>
  <input type="radio" id="gender" name="gender" value="others"/>others <br/>
  <br>Enter your Address:<br>
  <textarea></textarea><br>
  <input type="submit" value="sign-up">
</fieldset>
</form>
```

Output:

### Registration form

User personal information

Enter your full name

Enter your email

Enter your password

confirm your password

Enter your gender

Male

Female

others

Enter your Address:



### Markup Validation Service

An HTML validator is a quality assurance program used to check Hypertext Markup Language (HTML) markup elements for syntax errors.

A validator can be a useful tool for an HTML user who receives data electronically from a variety of input sources.

Validating a web page is a process of ensuring that it conforms to the norms or standards defined by the World Wide Web Consortium (W3C) which is the authority to maintain HTML standards.

There are several specific reasons for validating a web page, some of them are:

- It helps to create web pages that are cross-browser, cross-platform compatible. It is also likely to be compatible with the future version of web browsers and web standards.
- Standards compliant web pages increase the search engine spiders and crawlers' visibility, as a result your web pages will more likely appear in search results. It will reduce unexpected errors and make your web pages more accessible to the visitor.



Image 46: W3C Validation

Reference: <https://support.modernretail.com/hc/en-us/articles/201127998-W3C-Markup-Validation-Service>

## HTML5

### Introduction to HTML5

HTML5 is the latest standard for browsers to display and interact with web pages.

The latest versions of Apple Safari, Google Chrome, Mozilla Firefox, and Opera all support many HTML5 features and Internet Explorer 9.0 will also have support for some HTML5 functionality.

The mobile web browsers that come pre-installed on iPhones, iPads, and Android phones all have excellent support for HTML5.

### Features

- New Semantic Elements - These are like `<header>`, `<footer>`, and `<section>`
- Forms 2.0 - Improvements to HTML web forms where new attributes have been introduced for `<input>` tag.
- Persistent Local Storage - To achieve without resorting to third-party plugins.
- WebSocket - A next-generation bidirectional communication technology for web applications.
- Server-Sent Events - HTML5 introduces events which flow from web server to the web browsers and they are called Server-Sent Events (SSE).

- Canvas - This supports a two-dimensional drawing surface that you can program with JavaScript
- Audio & Video - You can embed audio or video on your webpages without resorting to third-party plugins.
- Geolocation - Now visitors can choose to share their physical location with your web application.
- Microdata - This lets you create your own vocabularies beyond HTML5 and extend your web pages with custom semantics.
- Drag and drop - Drag and drop the items from one location to another location on the same webpage.



Image 47:HTML

Reference: <https://www.markupbox.com/blog/wp-content/uploads/2017/02/html-benefits1.png>

## Page Layout Semantic Elements

A semantic element clearly describes its meaning to both the browser and the developer.

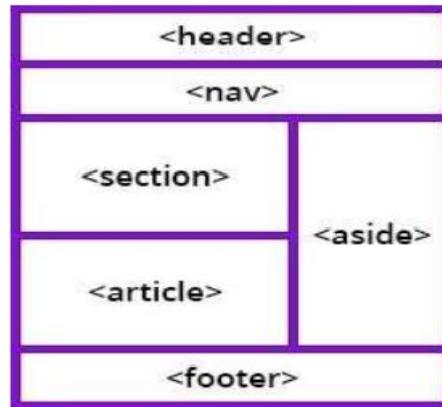
The following is the list of semantic page elements with a brief definition of each. Be sure to find an HTML5 reference you can trust for a complete list of valid attributes and child elements for each semantic element.

Block elements:

- 
- section - a generic page division used to help break up a larger block of content
  - article - content that is originally from an outside source, sometimes added dynamically on page load by using an aggregator script
  - header and footer -blocks that appear at the top and bottom of each page
  - hgroup - a grouping of multiple related headers, such as a title and subtitle
  - menu - a list of commands (see the command element), and attributes that specify the menu's behavior
  - nav -identifies a block that's strictly for Web site navigation, typically some unordered list of links to other pages on the site
  - address - includes contact information for the author of the content within a block, such as an article, section or entire page body
  - aside - indicates that the content should be treated as a sidebar

Inline element:

- summary and details -toggle between a teaser/summary and full details for the same content
- figure and figcaption elements used to apply common behavior to images, no matter what media elements (img, svg or canvas) are used to include them
- time -text representing a calendar date, clock time or both, formatted so that the browser can adjust for time zone differences if necessary
- command - a label and a behavior associated with that label when you use the keyboard or mouse to interact with it, typically used inside a menu block
- dfn -a term that's being defined within the content
- wbr -a tag indicating an acceptable place to break text within a word when it wraps across multiple lines



Source:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Using the section tag</title>
  </head>
  <body>
    <section>
      <h1>Hypertext markup language HTML</h1>
      <p>HTML is the standard markup language for creating web pages and web applications. Browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.</p>
    </section>
    <section>
      <h1>CSS</h1>
      <p>A formal language that is used as a description zone, formatting the appearance of a web page written with the help of markup languages HTML and XHTML but it can be applied to any XML-document, for example, to SVG or XUL.</p>
    </section>
  </body>
</html>
```

Output:

## Hypertext markup language HTML

HTML is the standard markup language for creating web pages and web applications. Browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

## CSS

A formal language that is used as a description zone, formatting the appearance of a web page written with the help of markup languages HTML and XHTML but it can be applied to any XML-document, for example, to SVG or XUL.

## Page Layout

Page layout is the part of graphic design that deals with the arrangement of visual elements on a page.

Source:

```
5  <body bgcolor="black" text="#FF1493">
6  <header>
7      <nav>
8          <ul>
9              <li>Your menu</li>
10         </ul>
11     </nav>
12 </header>
13
14 <section>
15
16     <article>
17         <header>
18             <h2>Article title</h2>
19             <p>Posted on <time datetime="2009-09-04T16:31:24+02:00">
20                 September 4th 2009</time> by <a href="#">Writer</a> - <a href="#"
21                 comments>6 comments</a></p>
22
23             <p>Pellentesque habitant morbi tristique senectus et netus et
24                 malesuada fames ac turpis egestas.</p>
25         </header>
26         <article>
27             <header>
28                 <h2>Article title</h2>
29                 <p>Posted on <time datetime="2009-09-04T16:31:24+02:00">
30                     September 4th 2009</time> by <a href="#">Writer</a> - <a href="#"
31                     comments>6 comments</a></p>
32
33             <section>
34                 <h3>About section</h3>
35                 <p>Donec eu libero sit amet quam egestas semper. Aenean ultricies mi
36                     vitae est. Mauris placerat eleifend leo.</p>
37         </section>
38
39     <aside>
40         <h3>About section</h3>
41         <p>Donec eu libero sit amet quam egestas semper. Aenean ultricies mi
42             vitae est. Mauris placerat eleifend leo.</p>
43     </aside>
44
45     <footer>
46         <p>Copyright 2009 Your name</p>
```

Output:

• Your menu

**Article title**

Posted on September 4th 2009 by [Writer](#) - [6 comments](#)

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

**Article title**

Posted on September 4th 2009 by [Writer](#) - [6 comments](#)

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

**About section**

Donec eu libero sit amet quam egestas semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo.

Copyright 2009 Your name

## HTML5 Web Forms

The evolution of HTML5 is to make a better World Wide Web. Writing less code is good (especially JavaScript), at the end of the day, those repetitive works of web developers should be taken over by web browsers. After all, this is the main reason man invented the machine.

What you can see below is a list of new input attributes and input types, we shall go through each and every one.

New Input attributes <input type="text" "new input attribute"/>

- Placeholder
- Autofocus
- Required
- DataList

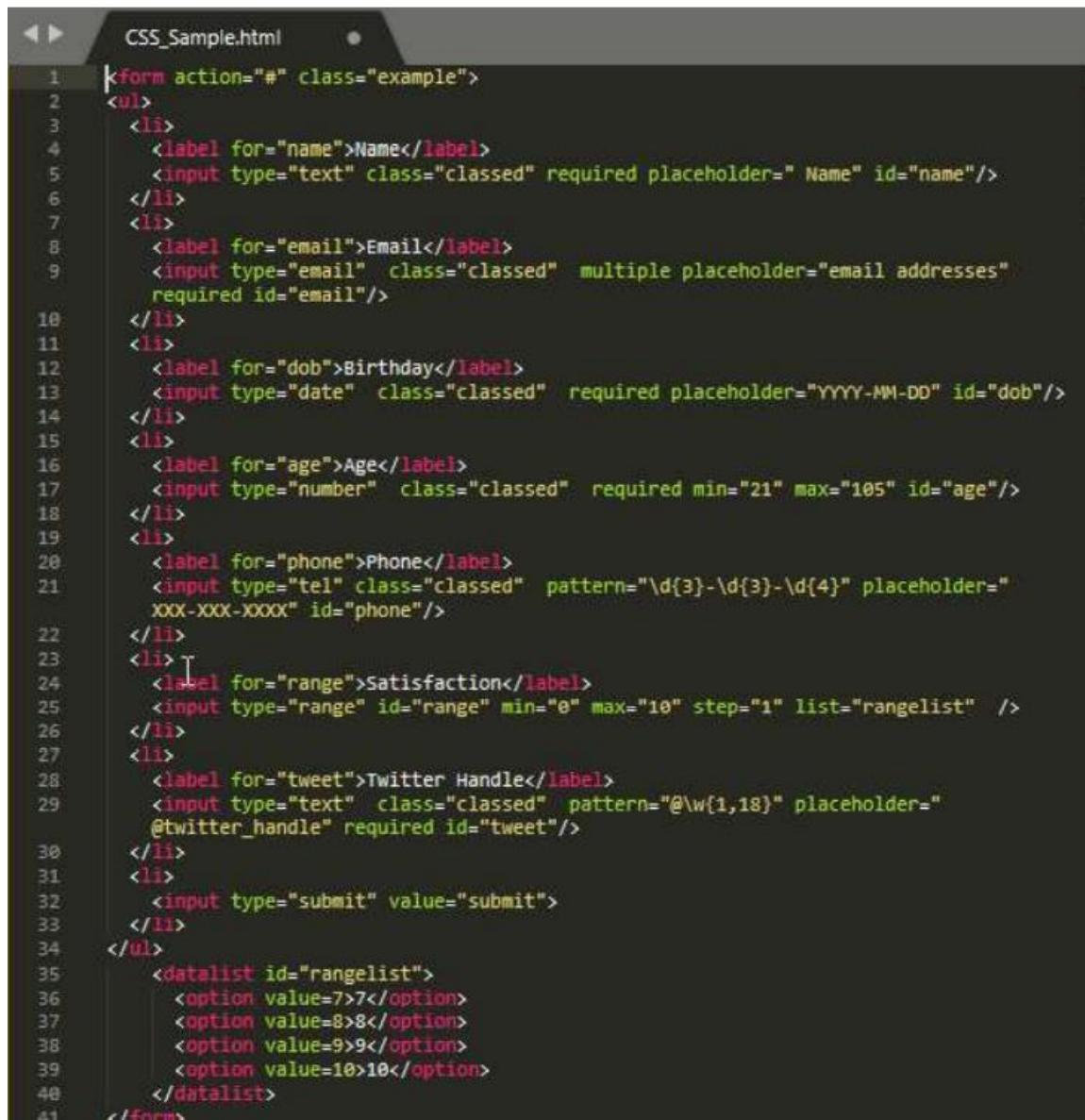
For years, we are only given a handful of "Type"s to be used in Input element, such as "text", "radio", "checkbox", "password", "file" and "submit", In HTML5, we will be having more fun with the new kids on the block.

New input types <input type="new input type"/>

- Search

- Email, URL and Phone
- Range as Slider
- Number as Spinner
- Date and Times
- Color picker

Source:



```
1 <form action="#" class="example">
2   <ul>
3     <li>
4       <label for="name">Name</label>
5       <input type="text" class="classed" required placeholder=" Name" id="name"/>
6     </li>
7     <li>
8       <label for="email">Email</label>
9       <input type="email" class="classed" multiple placeholder="email addresses"
10        required id="email"/>
11     </li>
12     <li>
13       <label for="dob">Birthday</label>
14       <input type="date" class="classed" required placeholder="YYYY-MM-DD" id="dob"/>
15     </li>
16     <li>
17       <label for="age">Age</label>
18       <input type="number" class="classed" required min="21" max="105" id="age"/>
19     </li>
20     <li>
21       <label for="phone">Phone</label>
22       <input type="tel" class="classed" pattern="\d{3}-\d{3}-\d{4}" placeholder="XXX-XXX-XXXX" id="phone"/>
23     </li>
24     <li>
25       <label for="range">Satisfaction</label>
26       <input type="range" id="range" min="0" max="10" step="1" list="rangelist" />
27     </li>
28     <li>
29       <label for="tweet">Twitter Handle</label>
30       <input type="text" class="classed" pattern="@\w{1,18}" placeholder="@twitter_handle" required id="tweet"/>
31     </li>
32     <li>
33       <input type="submit" value="submit">
34     </li>
35   </ul>
36   <datalist id="rangelist">
37     <option value=7>7</option>
38     <option value=8>8</option>
39     <option value=9>9</option>
40     <option value=10>10</option>
41   </datalist>
42 </form>
```

Output:

## HTML5 Form

Name	<input type="text"/>
Email	<input type="text"/>
Birthday	<input type="text"/>
Age	<input type="text"/>
Phone	<input type="text"/>
Satisfaction	<input type="text"/>
Twitter	<input type="text"/>
<input type="button" value="submit"/>	

## SVG

SVG defines vector-based graphics in XML format. SVG stands for Scalable Vector Graphics

Every element and every attribute in SVG files can be animated

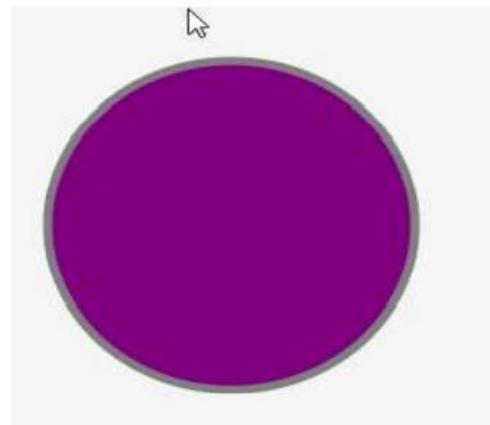
<circle> element

The SVG <circle> element creates circles, based on a center point and a radius. The coordinates of the circle's center are specified by the "cx" and "cy" attributes. And the radius of the circle is specified by the "r" attribute.

Source:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Title of the document</title>
  </head>
  <body>
    <svg height="300" width="300">
      <circle cx="150" cy="150" r="100" stroke="grey" stroke-width="5"
fill="purple" />
      Sorry, inline SVG isn't supported by your browser.
    </svg>
  </body>
</html>
```

Output:



## **HTML5 Media (Video & Audio)**

### **HTML Plug-ins**

- Plug-ins can be added to web pages with the `<object>` tag or the `<embed>` tag.
- Plug-ins can be used for many purposes: display maps, scan for viruses, verify your bank id, etc.

Source:

```
<!DOCTYPE html>
<html>
<body>
<embed width="400" height="50" src="bookmark.swf">
</body>
</html>
```

Output:



## **YouTube**

The easiest way to play videos in HTML, is to use YouTube.

- YouTube Video Id

- YouTube will display an id (like tgbNymZ7vqY), when you save (or play) a video. You can use this id, and refer to your video in the HTML code.
- Playing a YouTube Video in HTML

To play your video on a web page, do the following:

- Upload the video to YouTube
- Take a note of the video id
- Define an `<iframe>` element in your web page
- Let the `src` attribute point to the video URL
- Use the `width` and `height` attributes to specify the dimension of the player
- Add any other parameters to the URL (see below)

Source:

```
<!DOCTYPE html>
<html>
<body>
<iframe width="420" height="345"
src="https://www.youtube.com/embed/tgbNymZ7vqY">
</iframe>
</body>
</html>
```

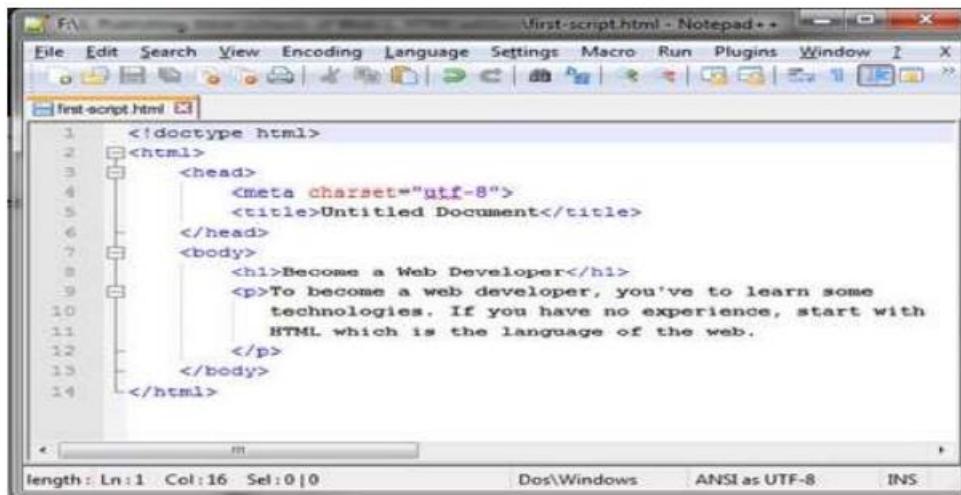
Output:



## Different editors used for Web Page Developing

### HTML Editors

HTML text editors are used to create and modify web pages. HTML codes can be written in any text editors including the notepad. One just needs to write HTML in any text editor and save the file with an extension ".html".



### Common features of HTML Code Editors

Text editors commonly used for HTML typically include either built-in functions or integration with external tools for such tasks as version control, link-checking and validation, code cleanup and formatting, spell-checking, uploading by FTP or WebDAV, and structuring as a project.

- Auto-completion.
- Adding a library for HTML entities.
- With the help of Site Explorer, you can view the files in a hierarchical pattern.
- Some editors have built-in FTP to upload the files faster.
- Advanced HTML editors provide support for other languages like CSS and JavaScript.
- highlighting syntax errors

Key Features of an HTML Editor
Interactive Text, HTML & Source Code Editor
Cleaving of Messy Code
Word to HTML Conversion
Find and Replace Tools for Texts Replacements
Table to DIV Conversions

Image 48: Key Features

Reference: <https://www.goodfirms.co/blog/best-free-open-source-html-editors-software>

## Different editors used for Web Page Developing

Some of the Popular Html Editors

1. Phase 5 HTML Editor
2. Notepad++
3. Sublime Text
4. jEdit HTML Editor
5. AdobeBrackets
6. SynWrite Editor
7. Visualcode Editor

## Phase 5 HTML Editor

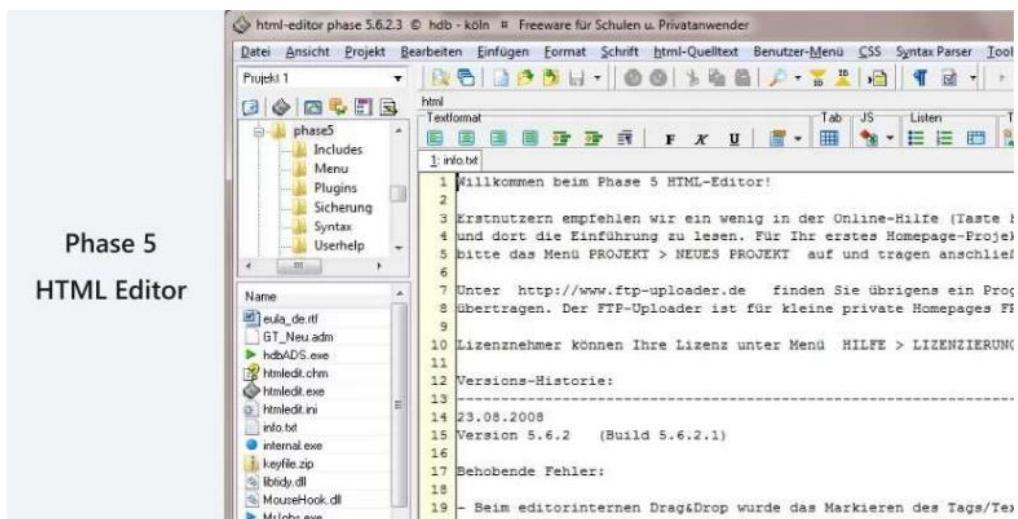


Image 49: Phase5 Editor

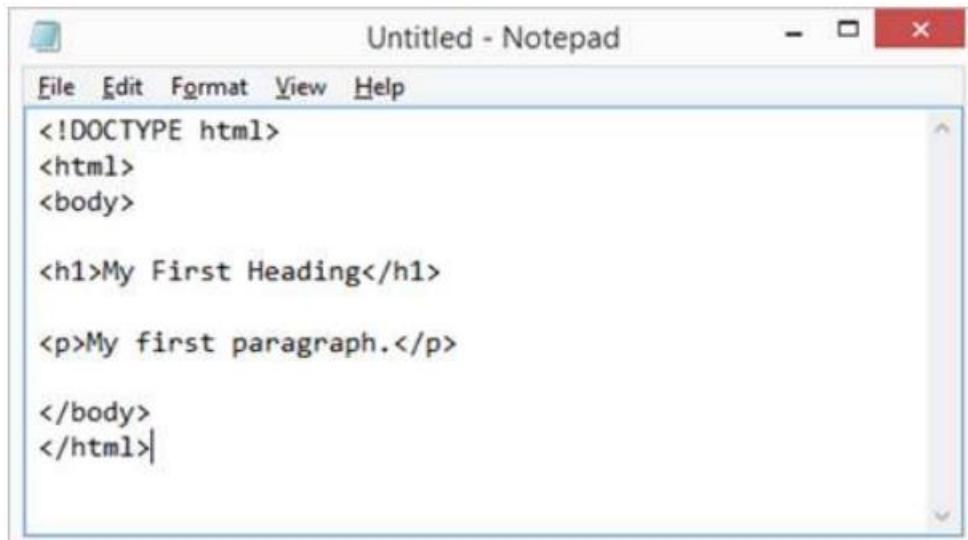
Reference: <https://blog.templateaster.com/best-free-html-editors/>

## Features

- Phase 5 is an impressive German HTML editor.
- It is freeware but only for Schools and Home users. If you run a big organization or a Company then you are required to buy the license key to run the program.
- Phase 5 is compatible with Windows only.
- Phase 5 HTML editors support different languages such as HTML, PHP, Java, JavaScript, Pearl, and VBScript.
- It has a crisp and clear Menu arrangement.
- Integrated file management makes the switching between different documents easy.
- Phase 5 has a tidy interface to work with.

## Notepad

Notepad is a simple text editor. It is an inbuilt desktop application available in Windows OS.

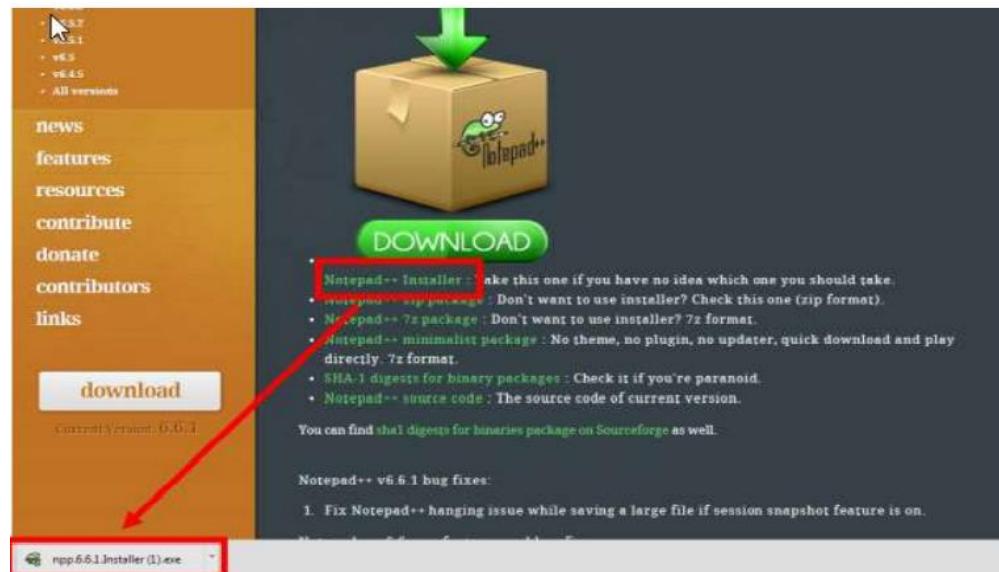


## Notepad++

### Installation

Step 1: - Go to the following website: - <http://notepad-plus-plus.org/download/v6.6.1.html>

Step 2: - Click on 'Notepad++ Installer'.



Step 3: -A 'account user control' window opens. Click 'yes'.

Step 4: - Click 'Ok'



Step 5: - Click 'Next'.



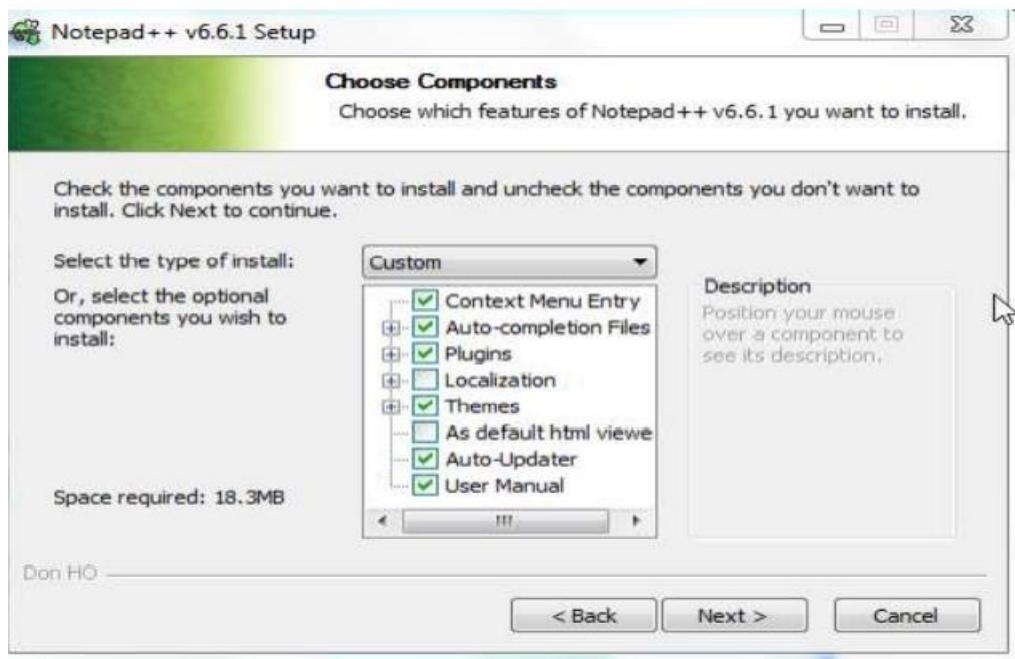
Step 6: - Click 'I Agree'.



Step 7: -Click 'Next'.



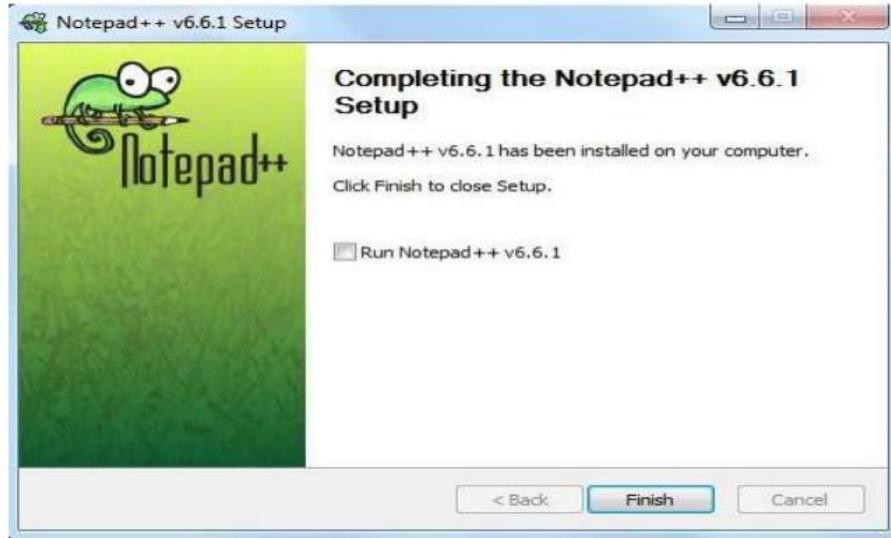
Step 8: - Click 'Next'.



Step 9: - Click 'Install'.



Step 10:- Click 'Finish'



#### Features:

- Autosave.
- Finding and replacing strings of text with regular expressions.
- Guided indentation.
- Line bookmarking.
- Macros.
- Simultaneous editing.
- Split screen editing and synchronized scrolling.

#### Pros

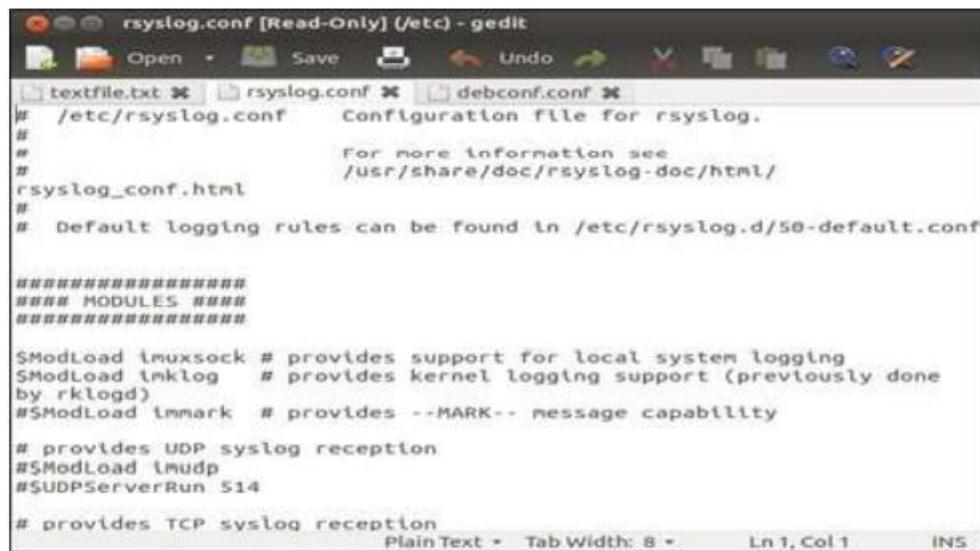
- Light and fast
- Portable
- Free
- Collaborative editing

## Cons

- Single platform support - Despite the software being as good as it can to be offered for free, it has limited support.
- Notepad++ is only available on Windows leaving out macOS and Linux.

## geditor

geditor provides a simple interface from which you have access to a full text editor with programming functions and is compatible with most languages



```
rsyslog.conf [Read-Only] (/etc) - gedit
Open Save Undo Redo Find Replace Cut Copy Paste Select All Find in Files
textfile.txt • rsyslog.conf • debconf.conf •
# /etc/rsyslog.conf      Configuration file for rsyslog.
#
# For more information see
# /usr/share/doc/rsyslog-doc/html/
rsyslog_conf.html
#
# Default logging rules can be found in /etc/rsyslog.d/50-default.conf

#####
##### MODULES #####
#####

$ModLoad imuxsock # provides support for local system logging
$ModLoad imklog   # provides kernel logging support (previously done
by rklogd)
$ModLoad immark  # provides --MARK-- message capability

# provides UDP syslog reception
$ModLoad imudp
$UDPServerRun 514

# provides TCP syslog reception
Plain Text • Tab Width: 8 • Ln 1, Col 1 INS
```

Image 50: gEdit  
Reference: <https://geek-university.com/linux/gedit-text-editor>

## Features

Some of the text-editing platform's features include

- Display line-number feature: This makes the software ideal for managing large source codes and it complements the 'Go to Line' function.
- The platform supports internationalized texts and handles Arabic characters without any problem.
- It comes with an integrated syntax highlighting feature.
- The integrated highlighting syntax feature makes the software ideal for editing scripts and configuring files.

- 
- It comes with an integrated spell checker: This allows users to autosave the document they are working on and access its copy in case it is damaged.
  - Supports GNOME: The software supports different types of document formats and allows users to open any ASCII file when using GNOME

### Pros of gedit

Some of the benefits of using this software are

- It comes with a plugin for the document statistics: This makes the software able to show statistics on selections with information such as the number of lines, characters, words, and bytes included.
- It is a light text-editing platform that is easy to use and highly efficient.
- The software ensures that users get access to a beautiful and user-friendly interface.
- It allows software developers to modify the code and analyze existing codes easily.
- The software is compatible with remote editing and a variety of languages.

### Cons of gedit

- Some of the limitations of the platform are
- It does not come with the integrated find and replace feature.
- Users might be concerned that the platform does not include a code-folding feature.

### Sublime Text 3:

Sublime is a cross platform code editor tool. It supports all markup languages

Step 1 - Download the .exe package from the official website as shown below  
<https://www.sublimetext.com/3>

Step 2 - Now, run the executable file. This defines the environment variables. When you run the executable file, you can observe the following window on your screen. Click Next.

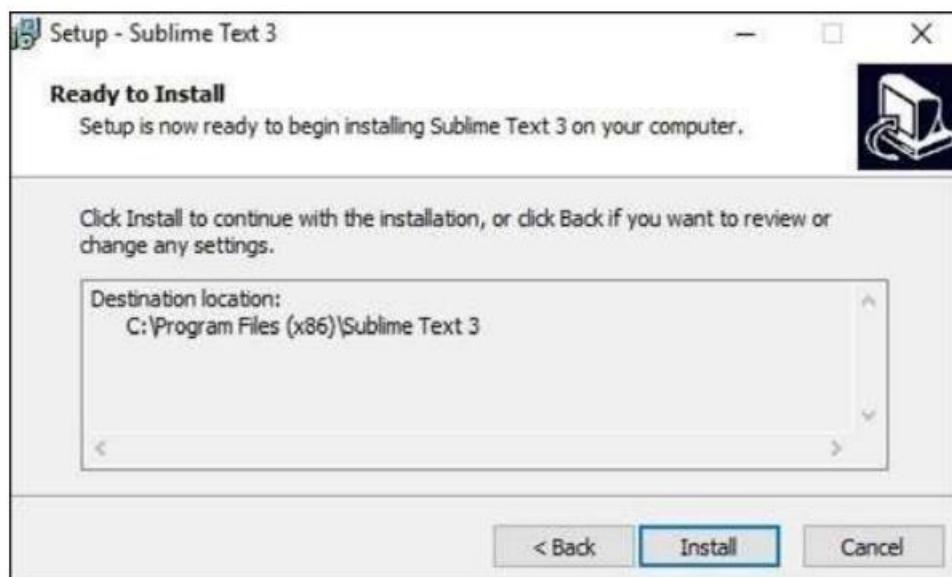


Image 51: Sublime

Reference: <https://www.thapatechnical.com/2018/09/html-css-auto-complete-plugin-in-sublime-Text-3.html>

Step 3 - Now, choose a destination location to install Sublime Text3 and click Next.

Step 4 - Verify the destination folder and click Install.



Step 5 - Now, click Finish to complete the installation.



Step 6 - Upon a successful installation, your editor will appear as shown below

#### Features

- Syntax Highlight
- Auto Indentation
- File Type Recognition
- Sidebar with files of mentioned directory
- Macros
- Plug-in and Packages

#### Pros

- Performance: all the operations like opening, closing, searching is fast
- Package: lots of packages and themes

- Customize: looks nice, also has many themes to choose from, and is able to configure using JSON file.
- Reliable: no need to do anything else once everything is installed and set up.

### Cons

- Cost: free download, but require \$70 license fee, otherwise frequent "buy license" pop-up.
- Package: package control is not installed ahead, so some searching needed; powerful only with plugins.
- Suggestions: only based on snippet, no context or language-based suggestions

### Bracket:

Bracket is an open-source software primarily used for Web development. It provides live HTML, CSS, JavaScript editing functionality.

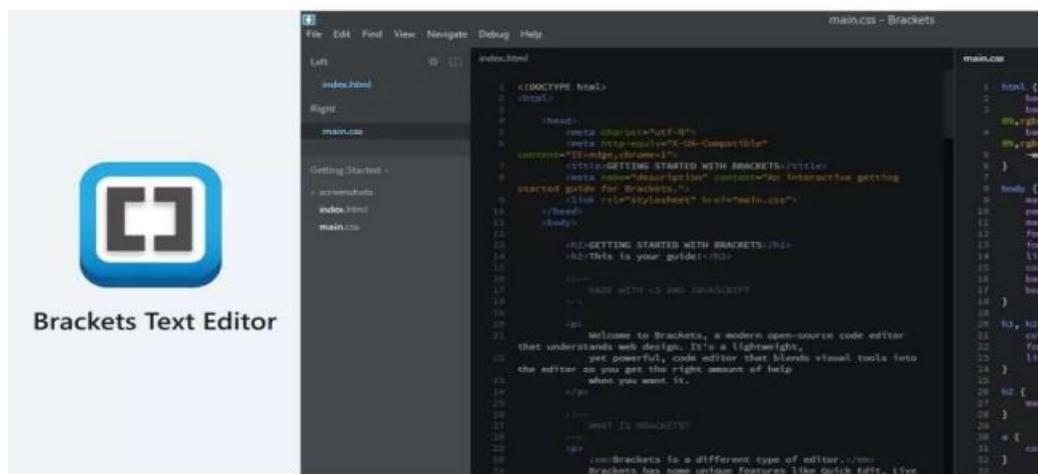


Image 52: Bracket

Reference: <https://blog.templatetoaster.com/best-free-html-editors/>

### Features

- Brackets is a top-notch Text editor for mac.
- Open source and free to download a text editor.
- Brackets support multiple platforms like macOS, Windows, Linux.
- The major feature which makes Brackets different from the rest of the HTML editors is the "Extract" feature.

- 
- Extract feature lets you extract information straight from PSD(s) like fonts, colors, and measurements with a clear CSS without contextual code references.
  - Simple to implement JavaScript, HTML, and CSS.
  - Easy to customize. It is a free HTML Editor.

#### Pros

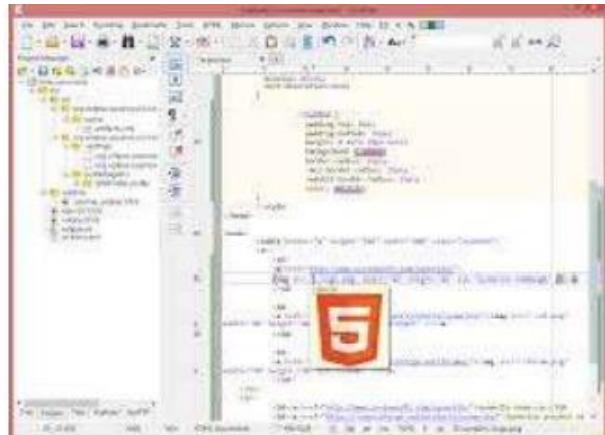
- Free to use for everyone
- The open-source nature of the software allows you to add your own extensions
- Includes standard features that are must-haves for web developers and code editors
- Unique features, such as Live Preview and Quick Edit, make code editing even easier
- Built using CSS, HTML, and Javascript, making it easy to hack and extend
- Gives you the ability to quickly download user-created extensions

#### Cons

- Mostly made for front-end developers who use HTML, Javascript, or CSS, with little functionality for those using server-side coding languages
- The extension registry does not have a filtering option making it hard to find what you are looking for

## SynWrite Editor

SynWrite combines great ideas from many well-known editors into a single, freely available product. It's a complete environment for Web workers, coders and writers.

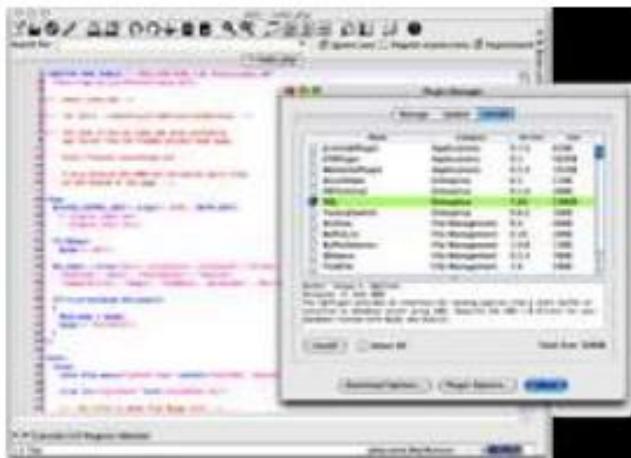


## Features

- macro recording
  - code highlighting
  - code folding,
  - multi-caret editing
  - regular expressions.
  - PlainEdit.NET
  - Portable PlainEdit.NET is an intuitive word processor with advanced settings and support for syntax highlighting.

## jEdit HTML Editor

jEdit is a mature programmer's text editor with hundreds (counting the time developing plugins) of person-years of development behind it.



## Features

- Written in Java, so it runs on Mac OS X, OS/2, Unix, VMS and Windows.
- Built-in macro language; extensible plugin architecture....
- Plugins can be downloaded and installed from within jEdit using the "plugin manager" feature.
- Auto indent, and syntax highlighting for more than 200 languages.

## Visual Studio Code

Visual Studio Code is a code editor redefined and optimized for building and debugging modern web and cloud applications.

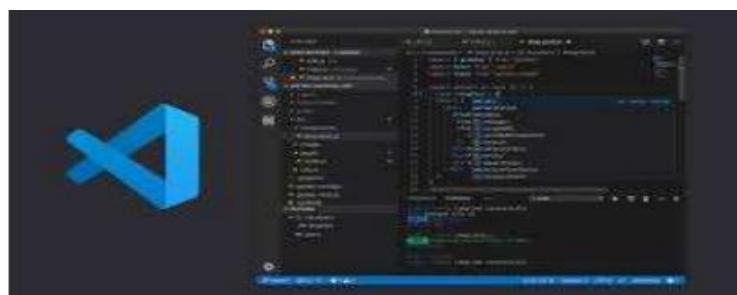


Image 53: Visual Studio  
Reference: <https://code.visualstudio.com/download>

## Installation

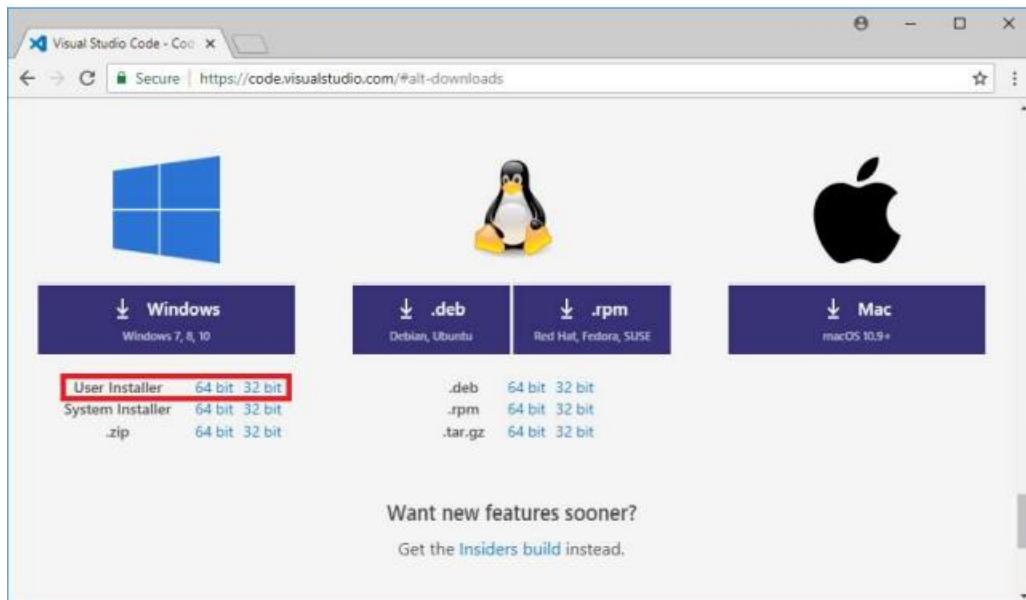
### Step 1- Download

Head over to the Visual Studio Code downloads page. There are three Windows installer versions available:

User installer: installs in your User folder and does not need Administrator privileges.  
System installer: installs for all users on the system and needs Administrator privileges.

.zip installer: a portable version.

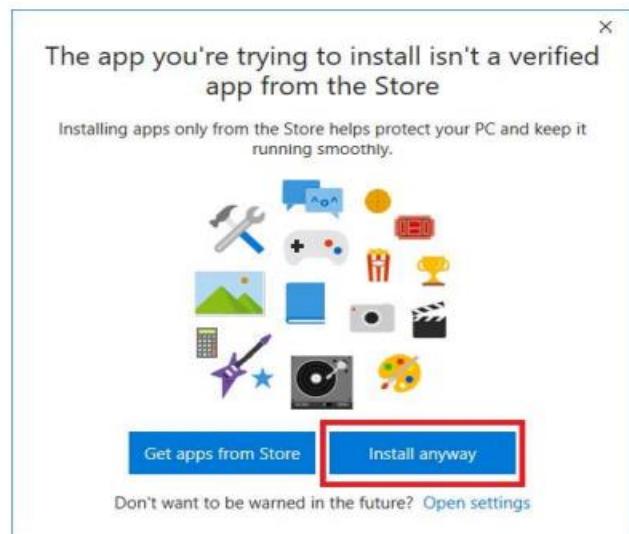
Check your windows bit version and click on the corresponding link.visual studio code download user installer Wait for the download to complete.



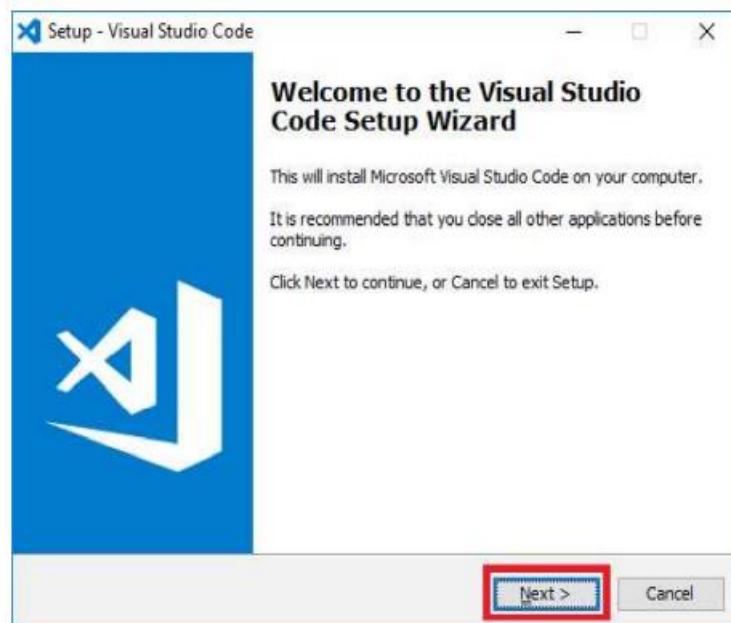
### Step 2- Install

Open the location of the downloaded executable.visual studio code downloaded installer Double-click it to run the installer.

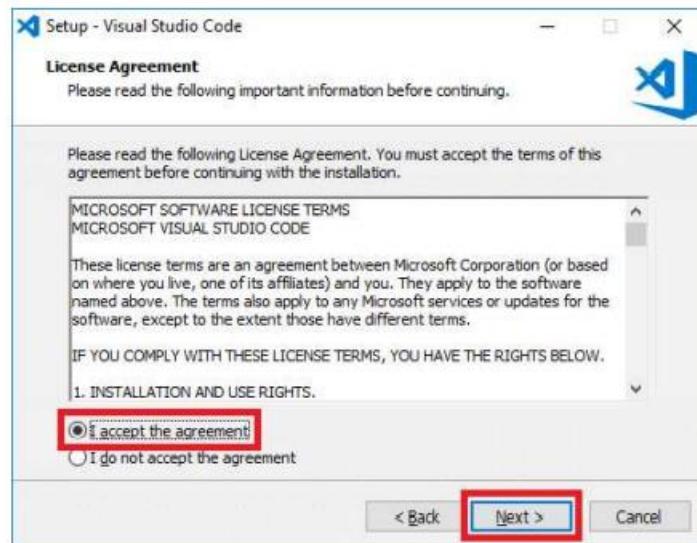
On Windows 10 a pop-up window will appear: The app you're trying to install isn't a verified app from the StoreClick on Install anyway.



Step3-The installer setup wizard will open.Click Next.

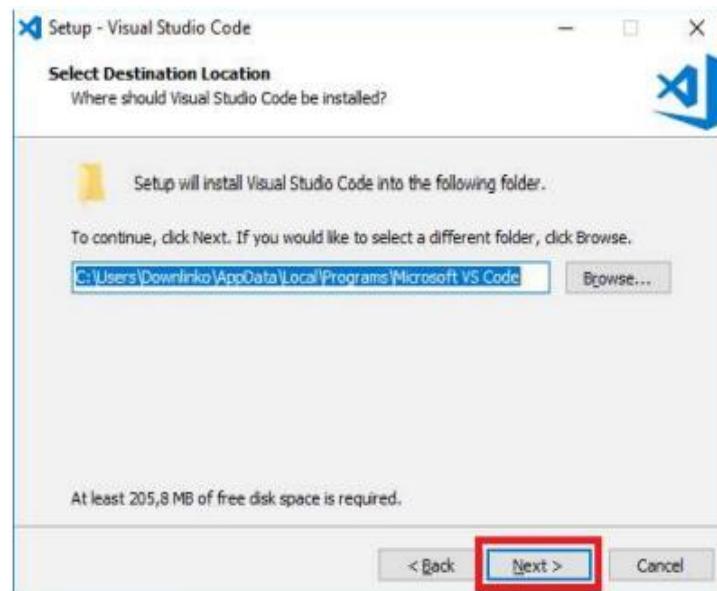


Step4- Click on the radio button next to I accept the agreement.Click Next.

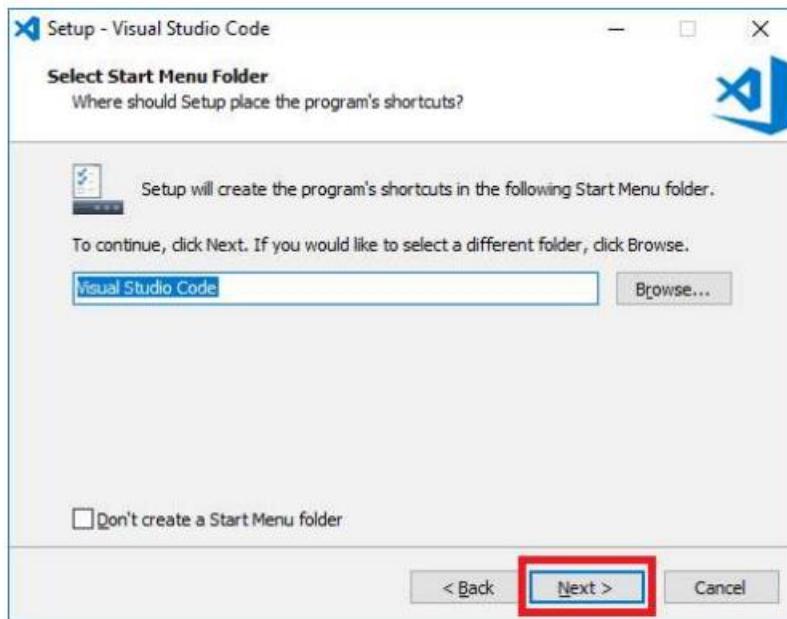


Step5-You can change the installation location by clicking on the Browse... button.In this example, we keep the default install location.Click Next.

Step6- Keep the default Start Menu Folder.Click Next.

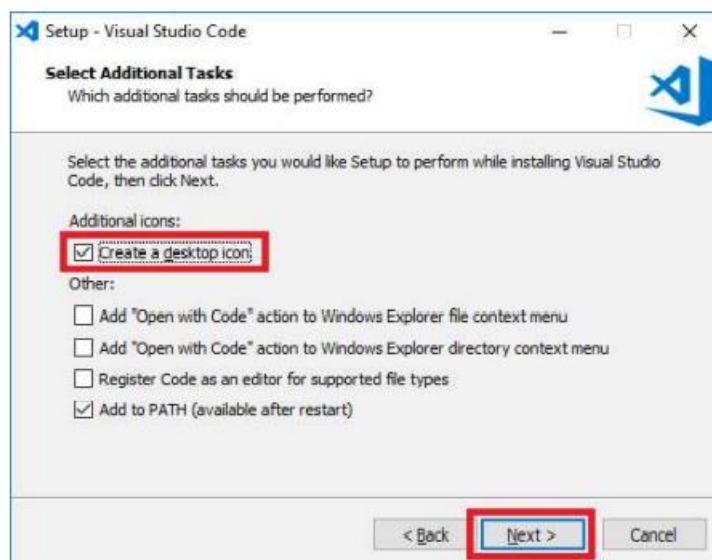


Step7- Select the Create a desktop icon checkbox.Click Next.



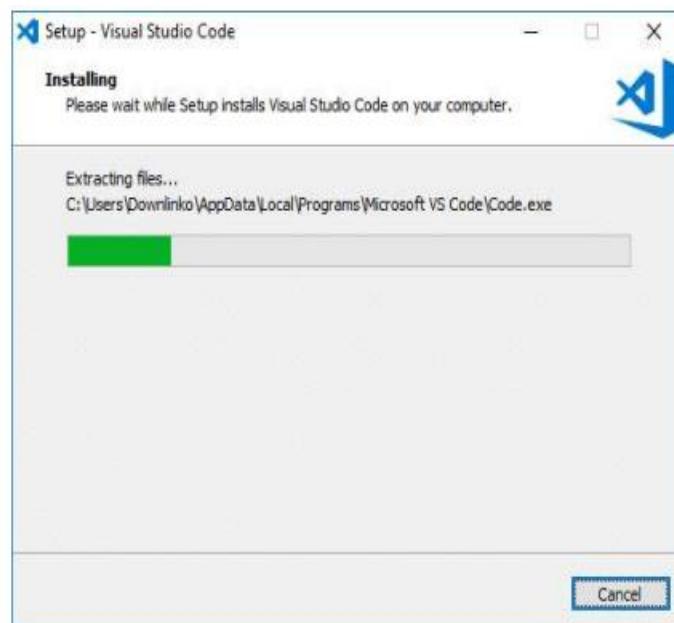
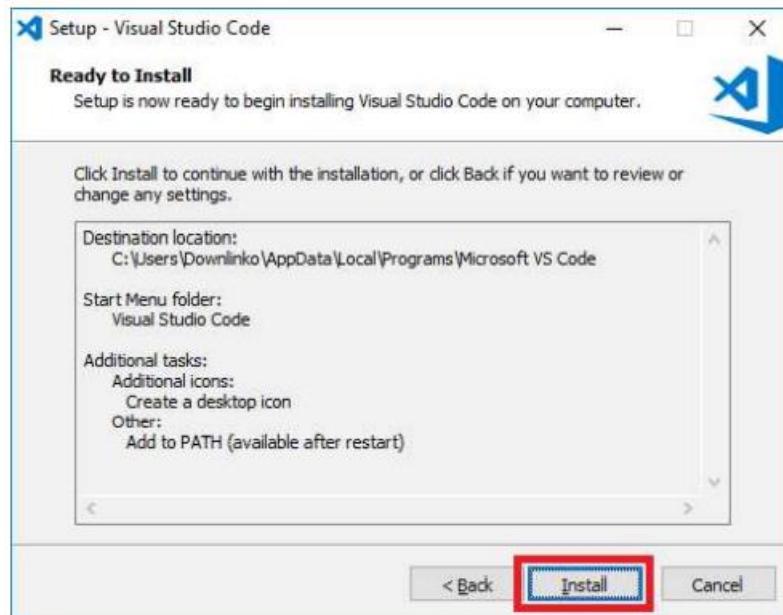
Step8-An overview of the selected installation settings is shown.

Click Install to start the installation.

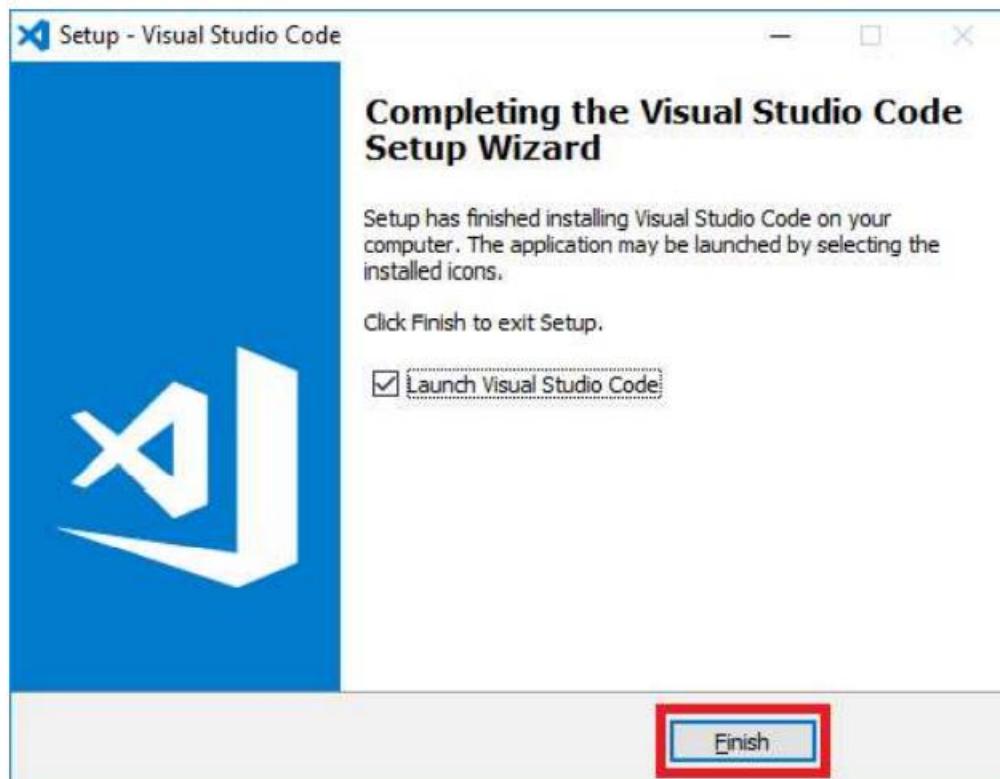


Step9-The Visual Studio Code installation will now start.

A progress bar shows the various steps that are executed.



Step 10-Once the installation is complete, click Finish.



### Step 11: Run

To start Visual Studio Code, double-click on the desktop shortcut.



### Features

- Supports the use of multiple split windows and horizontal/vertical orientation
- Includes VS Code IntelliSense that supports many different languages
- Command Palette that provides instant access to popular VS Code components

## Pros

- User friendly layout and visually stunning interface
- Easily customizable
- Thousands of plugins/extensions available through the VS Code Marketplace

## Cons

- Lacks most features of a full IDE suite
- Slow File Search
- Longer launch time than most of its competitors

## Comparison



The chart compares five free HTML editors: CoffeeCup HTML Editor, Komodo Edit, NetBeans, Notepad++, and VS Code. The features compared are: Free (all are checked), Open Source (Komodo Edit and NetBeans are checked), Number of Users (1 user for all), Upgrade Cost (\$29 once for Komodo Edit, \$7 monthly for NetBeans, Free for others), Support (Online for all), Text Editor (checked for all), WYSIWYG Editor (checked for Komodo Edit), Training (Documentation for all), and Deployment (Installed (Windows) for all).

	CoffeeCup HTML Editor	Komodo Edit	NetBeans	Notepad++	VS CODE
Free	✓	✓	✓	✓	✓
Open Source		✓	✓		
Number of Users	1 user	1 user	1 user	1 user	1 user
Upgrade Cost	\$29 once	\$7 monthly	Free	Free	Not available
Support	Online	Online	Online	Forum	Online
Text Editor	✓	✓	✓	✓	✓
WYSIWYG Editor	✓		With plugin		
Training	Documentation	Documentation	Documentation	Documentation	Documentation
Deployment	Installed (Windows)	Installed (Mac & Windows)	Cloud, SaaS, Web	Installed (Windows)	Installed (Mac & Windows)

Image 54: Comparison of Editors  
Reference: <https://blog.capterra.com/best-free-html-editors/>

## Applications of HTML

### How is its industry using HTML?

The World Wide Web Consortium (W3C) is an international community where Member organizations, a full-time staff, and the public work together to develop Web standards.

If you are learning about Web technology, you may wish to start with the introduction below,

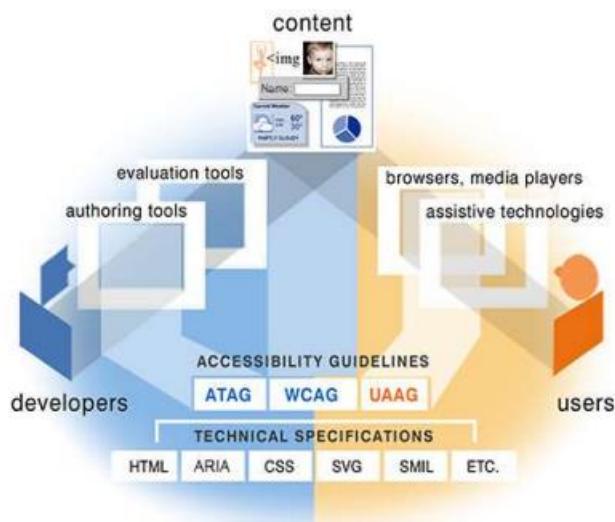


Image 55: Application in Industry  
Reference: <https://www.w3.org/WAI/standards-guide-lines/>

## What is a Static Website?

### Website

Website is a collection of related web pages that may contain text, images, audio and video. The first page of a website is called a home page. Each website has a specific internet address (URL) that you need to enter in your browser to access a website.

Website is hosted on one or more servers and can be accessed by visiting its homepage using a computer network. A website is managed by its owner that can be an individual, company or an organization.



Image 56: Website -www  
Reference: <https://www.javatpoint.com/website-static-vs-dynamic>

A website can be of two types:

1. Static Website
2. Dynamic Website

### Static website

Static website is the basic type of website that is easy to create. You don't need the knowledge of web programming and database design to create a static website. Its web pages are coded in HTML.

The codes are fixed for each page so the information contained in the page does not change and it looks like a printed page.

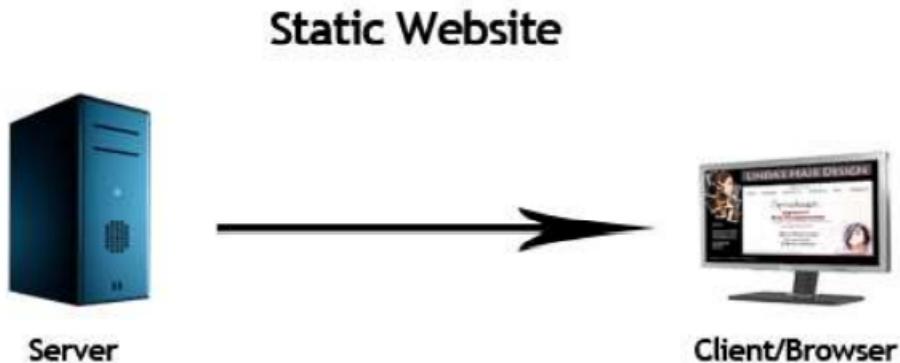


Image 56: Static Website  
Reference: <https://www.javatpoint.com/website-static-vs-dynamic>

### Dynamic Website

Dynamic website is a collection of dynamic web pages whose content changes dynamically. It accesses content from a database or Content Management System (CMS). Therefore, when you alter or update the content of the database, the content of the website is also altered or updated.

Dynamic website uses client-side scripting or server-side scripting, or both to generate dynamic content.

Client-side scripting generates content at the client computer on the basis of user input. The web browser downloads the web page from the server and processes the code within the page to render information to the user.

In server-side scripting, the software runs on the server and processing is completed on the server then plain pages are sent to the user.

## Dynamic Website

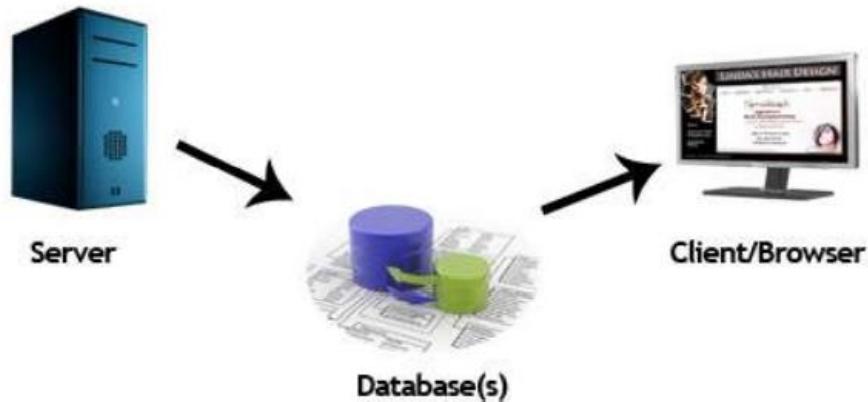


Image 57: Dynamic Website  
Reference: <https://www.javatpoint.com/website-static-vs-dynamic>

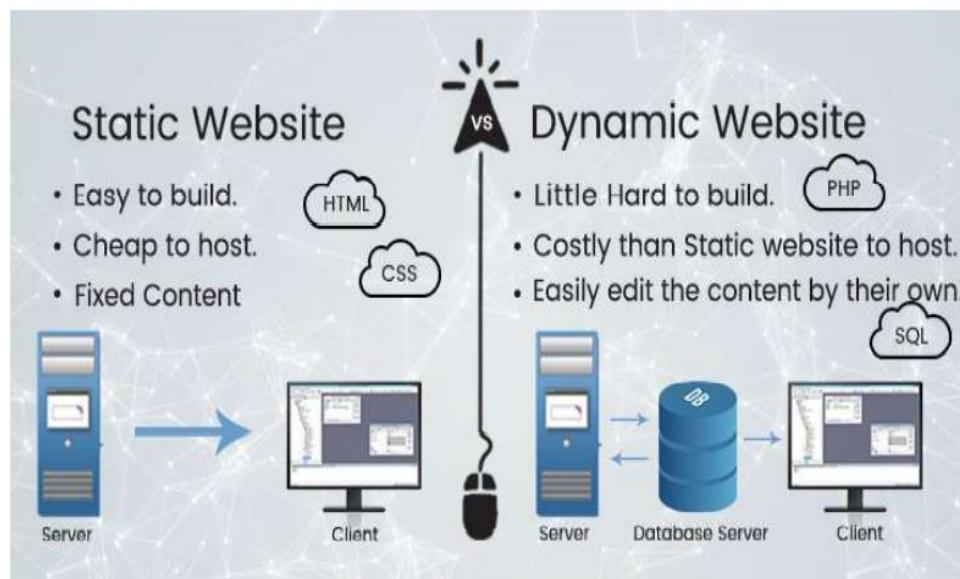
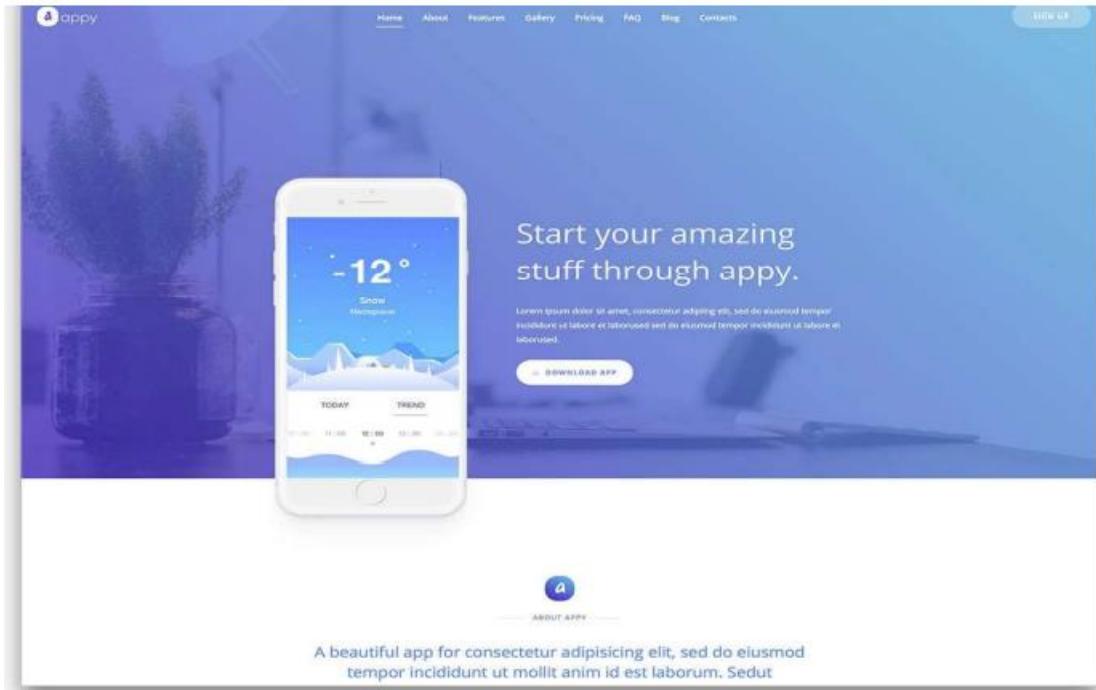


Image 58: Difference between static and dynamic  
Reference: <https://www.jeewangarg.com/blog/difference -between-static-and-dynamic-website>

## Practical Application



## Source Code

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<link rel="stylesheet" type="text/css" href="style.css"/>

<title>Landscape Titles florida web design </title>

</head>

<body>

<div id="container">

<div id="header" >

<h1><span class="off ">Landscape </span>Titles </h1>

<div id="links">

```

---

```
<a href="#">Home </a>
<a href="#">About</a>
<a href="#">Products </a>
<a href="#">Services </a>
<a href="#">Portfolio</a>
<a href="#">Contact </a>
</div>
</div>
<div id="content" >

<div class="contenttext">
<h2>Template Information</h2>
<p>You may use this template in any way you would like, please just leave the link at the bottom to our site in place. In order to add your own pictures, simply insert an image as usual, and just add class="picture" to each image. The shadow is automatically added to the images. Make sure your image is exactly 750px wide for best results. </p>
</div>

<div class="contenttext">
<h2>Template Information</h2>
<p>You may use this template in any way you would like, please just leave the link at the bottom to our site in place. In order to add your own pictures, simply insert an image as usual, and just add class="picture" to each image. The shadow is automatically added to the images. Make sure your image is exactly 750px wide for best results. </p>
</div>

<div class="contenttext">
```

## <h2>Template Information</h2>

<p>You may use this template in any way you would like, please just leave the link at the bottom to our site in place. In order to add your own pictures, simply insert an image as usual, and just add class="picture" to each image. The shadow is automatically added to the images. Make sure your image is exactly 750px wide for best results. </p>

</div>

</div>

<div id="footer"><a href= "<http://www.google.com>">web design florida</a></div>

</div>

</body>

</html>

## Style.css

```
body {  
margin: 0;  
padding: 0; text-align: left;  
font: Arial, Helvetica, sans-serif; font-size: 13px;  
color: #000000; background: #1364A6;  
background-image:url(images/background.png); background-repeat:repeat-x;  
}  
*  
{  
margin: 0 auto 0 auto; text-align:left;} #container  
{  
display: block; height:auto; position: relative; width: 750px;  
background-image:url(images/background.jpg); background-repeat:repeat-y;
```

```
padding-left:17px; padding-right:17px; overflow:hidden;  
}  
  
#links  
{  
  
display:block; float:right; margin-top:25px;  
  
margin-right: 10px;  
}  
  
#links a, #links a:visited  
{  
  
font-size:16px; font-weight:bold; margin-left:4px; margin-right:4px; color: #1652B4;  
  
text-decoration:none;  
}  
  
#links a:hover  
{  
  
border-bottom:dotted 1px #1652B4;  
}  
  
#header  
{  
  
height:60px;  
}  
  
#header h1  
{  
  
margin-top: 18px; margin-left:10px; font-size:33px; color:#OA2645; float:left;  
}  
  
.off
```

```
{  
color:#1652B4;  
}  
.picture  
{  
background-image:url(images/imagebackground.jpg); background-repeat:repeat-x;  
background-position:bottom; padding-bottom:8px;  
margin-top:22px;  
}  
.contenttext h2  
{  
color:#123F83; font-size: 24px;  
margin-bottom:10px;  
}  
.contenttext  
{  
padding-left: 10px; padding-right:10px;  
}  
#footer  
{  
font-size:12px; color:#519FEE; margin-top:18px; width:750px; text-align:center;  
}  
#footer a, #footer a:visited  
{  
text-align:center; color:#ADCCF1;
```

text-decoration:none;

}

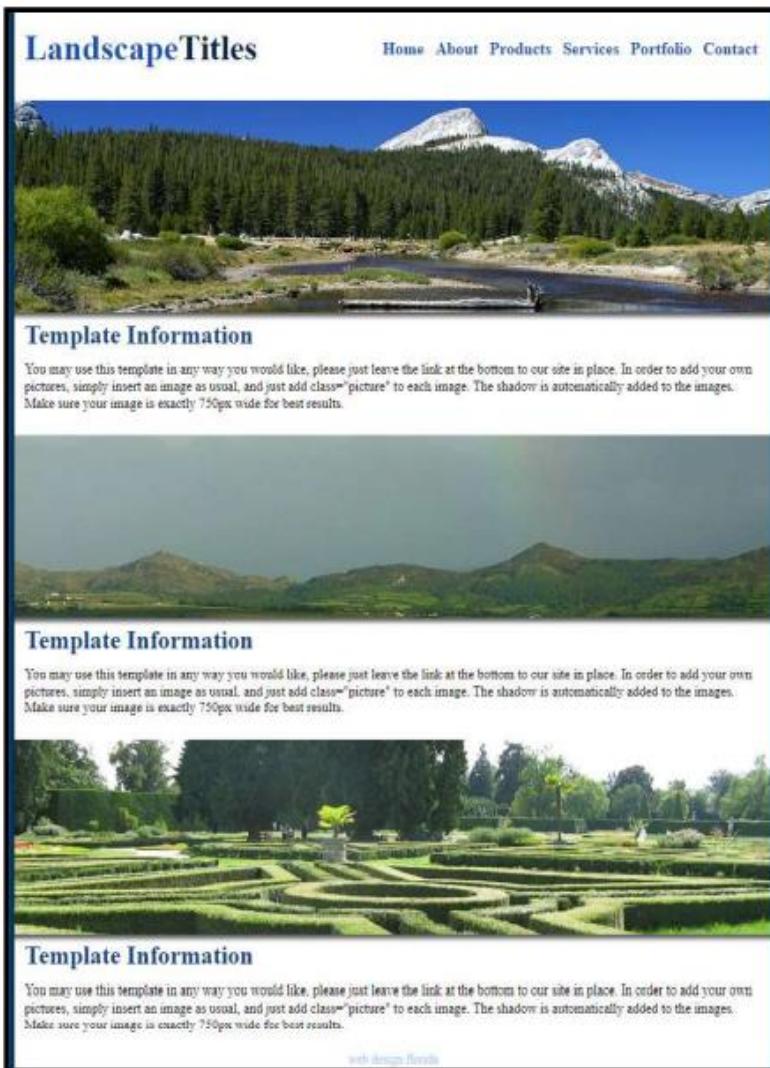
html, body {

text-align: center;

}

p {text-align: left;}

## Output:



The screenshot shows a website template with a header "LandscapeTitles" and a navigation bar with links: Home, About, Products, Services, Portfolio, Contact. Below the header is a large landscape image of a mountain range and a lake. A "Template Information" section is overlaid on the image, containing text about using the template and a placeholder image. This pattern repeats for two more landscape images: a rainbow over a lake and a garden labyrinth.

**LandscapeTitles**

Home About Products Services Portfolio Contact

**Template Information**

You may use this template in any way you would like, please just leave the link at the bottom to our site in place. In order to add your own pictures, simply insert an image as usual, and just add class="picture" to each image. The shadow is automatically added to the images. Make sure your image is exactly 750px wide for best results.

**Template Information**

You may use this template in any way you would like, please just leave the link at the bottom to our site in place. In order to add your own pictures, simply insert an image as usual, and just add class="picture" to each image. The shadow is automatically added to the images. Make sure your image is exactly 750px wide for best results.

**Template Information**

You may use this template in any way you would like, please just leave the link at the bottom to our site in place. In order to add your own pictures, simply insert an image as usual, and just add class="picture" to each image. The shadow is automatically added to the images. Make sure your image is exactly 750px wide for best results.

work designs florals

## TOP 10 USES OF HTML

### 1. Web pages development

HTML is heavily used for creating pages that are displayed on the world wide web. Every page contains a set of HTML tags including hyperlinks which are used for connecting to other pages. Every page that we witness, on the world wide web, is written using a version of HTML code.

### 2. Web document creation

Document creation on the internet is dominated by HTML and its basic concept via tag and DOM i.e., document object model. HTML tags are inserted before and afterward or phrases to locate their format and location on the page. A web document consists of three sections:

title, head, and body. Head includes the information to identify the document, including title and any other important keyword. A title can be seen on the browser's bar and the body section is the main portion of the website visible to the viewer. All three segments are designed and created by the use of HTML tags. Every section has its own specific set of tags, which are dedicatedly rendered keeping the head, title and body concepts in a loop.

### 3. Internet navigation

This is one of the most important uses of HTML which is revolutionary. This navigation is possible by utilizing the concept of Hypertext. It is basically a text which refers to other web pages or text and when the user clicks on it, would navigate to the referenced text or page. HTML is heavily used to embed the hyperlink within the web pages. A user can easily navigate within the web pages and between websites as well, which are located on different servers.

### 4. Cutting edge feature

HTML5 with its set of standards and API is being used to introduce some of the latest trends in website creation business. Like polyfill libraries, which are supported by old browsers equally well. Browsers like Google Chrome are the perfect choice when it comes to implementing an HTML5 latest set of standards and APIs. There is a JavaScript library available called Modernizr, which can detect features that let the developer dynamically load polyfill libraries as required.

### 5. Responsive images on web pages

At the elementary level in applications of HTML, queries can be set to utilize the images, which are responsive in nature. With the srcset attribute of img element in HTML, and combining it with picture elements, a developer can fully control how the user will render an image. Now different types of an image with size variation can be loaded by using the

---

img element. Rules can be easily set with the picture element, we can declare an img element with default source and then for every case, a source can be provided.

## **6. Client-side storage**

Earlier, a user could not save the user's browser data that would persist across sessions. To meet this requirement, server-side infrastructure has to be built or user's cookies can be used. But with HTML5, client-side storage is feasible using localStorage and IndexDB. These two strategies have their own standards and features. localStorage basically gives string-based hash-table storage. Its API is very simple and provides the developer with setItem, getItem, and removeItem methods. IndexDB, on the other hand, is a larger and better client-side data store. IndexDB database can be expanded with the user's permission.

## **7. Offline capabilities usage**

Once data can be stored in the browser, the developer can think of a strategy to make the application work, when a user is disconnected. HTML5 has its application cache mechanism which would define how the browser manages the offline situation. Application cache, responsible for offline ability actually comprises different components, which includes API methods that create an update, read manifest file and events. By using the certain property in HTML5, a developer can check if the application is online or not. A developer can also specify in a website's application cache manifest file the information like, what browser manages resources for offline use. In the manifest file, resources which are available offline can also be specified.

## **8. Data Entry support with HTML**

HTML5 standard and set of APIs can be used to support data entry level of work. As browsers implement new HTML5 standards, developers can simply add the attributes to the tag which indicate required fields, text, data format etc. HTML5 has come up with several new attributes to drive on-screen keyboards, validation, and other data-entry experiences so that end-users can have a better data-entry.

## **9. Game development usage**

Before the advent of HTML5, game development was an exclusive domain of Flash and Silverlight. Since browsers support new specifications for HTML5 including CSS3 and light-fast JavaScript engine to drive a new rich experience, HTML5 can bring the reality of game development possible, which was earlier the forte of Flash and Silverlight. Every

---

single feature of APIs need not be implemented, but most appropriate ones can be utilized while eliminating the rest of the features.

## **10. Native APIs usage to enrich website**

HTML5 adds so many new abilities and tools, which was just an imagination in the past. A large set of new APIs regarding file system, Geolocation, drag, and drop, event handling, client-storage etc. are the capabilities which make usage of HTML5 easier than ever before. Application experience can be enhanced with other APIs like Fullscreen, Visibility and Media Capture. A modern web application has asynchronous nature which can be fostered using Websockets and Web workers like APIs.

---

### Learning Outcome

- Create Styles of web pages using CSS

## Cascaded Style Sheet (CSS)

### Introduction to CSS

CSS stands for Cascading Style Sheets. It is the language for describing the presentation of Web pages, including colours, layout, and fonts, thus making our web pages presentable to the users.

CSS describes how HTML elements are to be displayed on screen, paper, or in other media

It is used to change the style (color, height, width, padding, margin, etc.) of the HTML element. By using CSS, we can design a very good user interface for your website or web application.

Styling your web page to make it look appealing is also an essential element of web designing and development. CSS helps in giving styles for describing the presentation of the markup-based documents.

### What is CSS

- **Cascading Style Sheets**
  - Contains the rules for the **presentation** of HTML.
  - CSS was introduced to keep the **presentation** information **separate** from **HTML** markup (content).
- 
- The diagram illustrates the relationship between HTML, CSS, and a Web Page. It shows a stack of paper labeled 'HTML' with various code snippets like 'TITLE', 'HEAD', 'BODY', 'H1', 'P', 'IMG', 'SCRIPT', and 'STYLE'. This is followed by a plus sign ('+') and another stack of paper labeled 'CSS' with code snippets like 'COLOR', 'FONTSIZE', 'FONTCOLOR', 'BACKGROUND', 'HEIGHT', 'WIDTH', 'MARGIN', 'PADDING', and 'CLEAR'. An equals sign ('=') leads to a final stack of paper labeled 'Web Page' which displays a sample web page with a blue header and text.

Image 1: CSS

Reference: <https://www.slideshare.net/webdevninja/introduction-to-css-13603389>

### World Wide Web Consortium(W3C)

CSS is developed, updated, and maintained by a faction of people of the W3C, and the group is named "CSS Working Group". This group produces documents called specifications. When these specifications in the form of documents are released and approved by the developers' community, they are set officially so that other users can use it as a standard mechanism of using CSS.

### History of CSS

CSS was first proposed by **Hakon Wium Lie** on October 10, 1994. At the time, Lie was working with Tim Berners-Lee (father of Html) at CERN. The European Organization for Nuclear Research is known as CERN. Hakon wium lie is known as father of css.

CSS was proposed in 1994 as a web styling language, to solve some of the problems of Html 4. There were other styling languages proposed at this time, such as Style Sheets for Html and JSSS but CSS won.



Image 2: Founder of CSS

Reference: : <https://www.shecodes.io/workshops/shecodes-online-workshop-46-0/projects/197698>

## Include properties in CSS2

CSS level 2 specification was developed by the W3C and published as a recommendation in May 1998. CSS 2 includes a number of new capabilities like below;

- absolute
- relative
- fixed
- positioning
- z-index
- concept of media type
- bidirectional text
- new font properties such as shadows.

CSS3 was started in 1998 but it has never been completed. Some parts are still being developed and some components work on some browsers. It published in June 1999. CSS 3 is divided into several separate documents called "modules". Each module adds new capabilities or extends features defined in CSS 2.

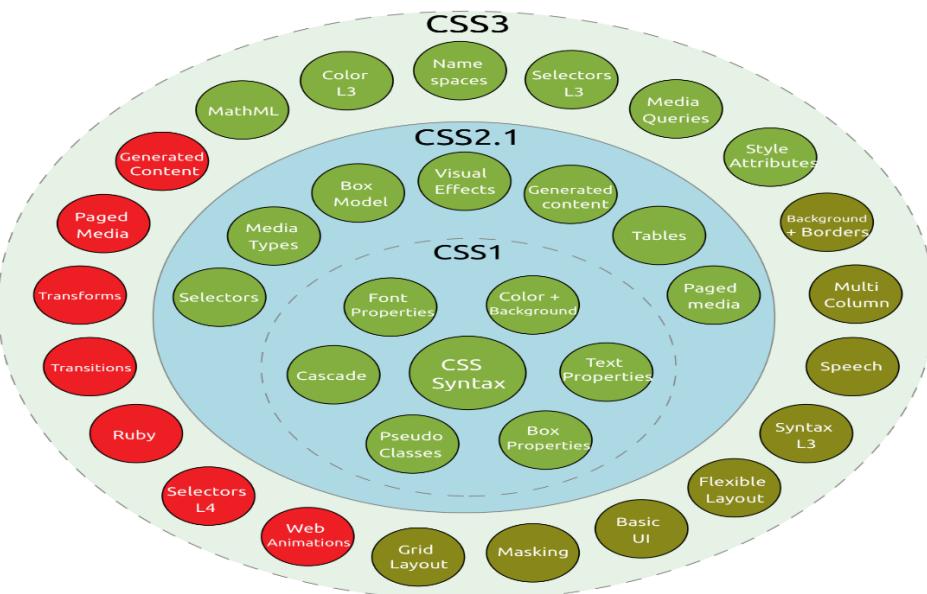


Image 3: History of CSS  
Reference: <https://www.shecodes.io/workshops/shecodes-online-workshop-46-0/projects/197698>

## Why CSS?

### CSS saves time:

You can write CSS once and reuse same sheet in multiple HTML pages.

### Easy Maintenance:

To make a global change simply change the style, and all elements in all the webpages will be updated automatically.

### Search Engines:

CSS is considered as clean coding technique, which means search engines won't have to struggle to "read" its content.

### Superior styles to HTML:

CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.

### Offline Browsing:

CSS can store web applications locally with the help of offline cache. Using of this we can view offline websites.

## **Limitations of CSS**

### **Confusion due to many CSS Versions:**

Beginners are more vulnerable to this issue. They might get confused while opting to learn CSS as there are many levels of CSS such as CSS2, CSS3, etc.

### **Cross-Browser Issues:**

Different browsers work differently. So, you have to check that changes implemented in the website via CSS codes are reflected properly among all the browsers.

### **Security Issues:**

Security is important in today's world driven by technology and data. One of the major disadvantages of CSS is that it has limited security.

### **Extra Work for Developers:**

Design services are required to consider and test all CSS codes across different browsers for compatibility. Due to developers testing compatibility for different browsers, their workload increases.

## **Advantages of CSS**

### **CSS saves time**

Once you write a CSS code you can use it in more than one HTML page. User can define a style for each HTML element and it can apply as many web pages. By doing this we save a lot of time.

### **Pages load faster**

You don't need to write HTML tag attributes every time if you are using CSS. Just write a CSS Rule of a tag and apply it to every web pages. By doing this, the download time of your website will be greatly reduced.

Maintenance of a website becomes very easy using Easy maintenance –CSS. If you want to change the style of your website completely, then you have to change the simple style code, all the other page elements will be changed automatically.

### **Superior styles to HTML**

CSS has a lot more detailed features than HTML, allowing you to give your HTML page a much better look than HTML features.

## Multiple Device Compatibility

Style sheets allow the material to be optimized for more than one type of device. Using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.

## Global web standards

Now recommended to use CSS as required by HTML attributes. So, it is a good idea to start using CSS in all HTML pages to make them compatible with future browsers.

## Offline Browsing

On using CSS, we can view offline websites. The cache is also very helpful in making a website load faster and ensures better performance of the website.

## Platform Independence

CSS Script provides consistent platform independence and also supports every latest Web Browser.

## CSS Syntax

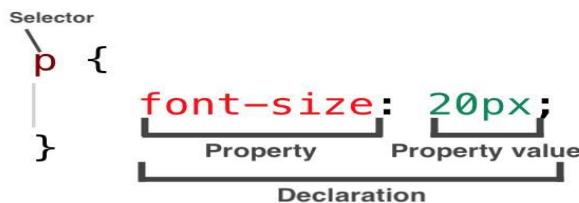


Image 4: CSS Syntax

**Selector:** selects the element you want to target

**Keys:** properties(attributes) like color, font-size, background, width, height, etc

**Value:** values associated with these properties

There are few basic selectors like tags, id's, and classes. All forms this key-value pair.

## CSS Comments

Comments don't render on the browser

- Helps to understand our code better and makes it readable.
- Helps to debug our code

---

Comments are used to explain the code, and may help when you edit the source code at a later date.

Comments are ignored by browsers.

### **Two ways to comment:**

#### **Single line**

```
/*<h6> This represents the most/ least important line of the doc. </h6>*/
```

#### **Multiple lines:**

```
/*
  h1
  {
    color: red;
    text-align: center;
  }
*/
```

### **White Spaces in CSS**

White spaces are special characters that can be an actual space, tab, or newline (carriage return). These whitespaces are used to construct your stylesheets extra readable. Like that of HTML, the browser usually ignores almost all of the whitespaces within your CSS code; and is meant to make the code human-readable.

### **Three ways to integrate CSS**

There are three ways of inserting a style sheet in any Html documents.

- Inline style sheet
- Internal style sheet (Embedded)
- External style sheet

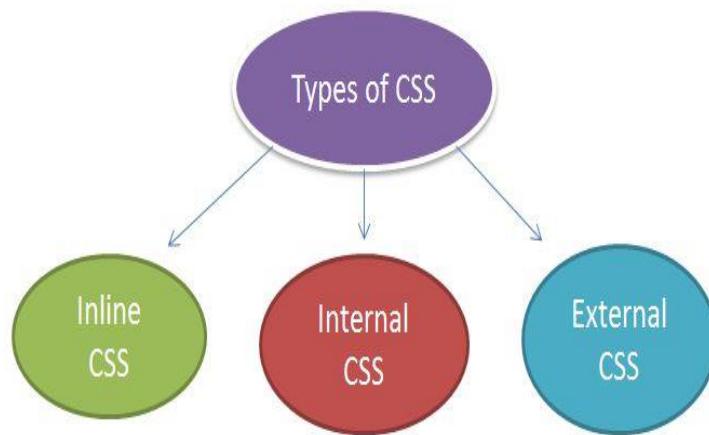


Image 5: Types of CSS  
Reference: <https://www.bitdegree.org/learn/inline-css>

### Inline Styles:

- Inline styles are placed within an HTML element in the code.
- When you use inline style, your styling will only affect the element you selected.
- Inline styles do not have selectors because it's written inside the html element.

Example:

```
<p style="color: red; margin-left:20px">This is a paragraph. </p>
```

### Internal Styles:

- are placed in head section of the web page you are writing via style tag `<style type="text/css"></style>`.
- The styles you have written will only be used for the web page you used it in.
- Internal Styles are also called “Embedded styles”

Example:

```
<style>
hr {color:red;}
p {margin-left:20px;}
</style>
```

## External Styles:

External Styles can be reused to apply on more than one page by only linking the style sheet to the web page.

Write your CSS codes via any of the code editors and then save it as a .css file. then link the style sheet to the HTML page by adding this code in the head section:

Example:

```
<head>
<link rel="stylesheet" type="text/css" href="name of the Css file">
</head>
p{color: red; }    //css file
```

## Merits and demerits of - external Style Sheets, Embedded Style Sheets

### Embedded Style Sheets

#### Merits

- No need to upload multiple files as the CSS code is added to the same HTML file corresponding to the web page.
- Class and ID selectors can be used.

#### Demerits

- Adding CSS code to the HTML file results in increasing the page size and therefore, reducing the loading speed.
- Using it for multiple web pages is ineffective as it is required to add the same CSS rules for every web page

### External Style Sheets

#### Merits

- A single external CSS file can be used for styling several web pages.
- HTML files leveraging external CSS have a cleaner structure and are smaller in size.

#### Demerits

- Linking to or uploading several external CSS files might decrease a website's download speed and affect its performance.

- 
- Web pages requiring the external CSS file might not be rendered accurately until the same is fully loaded.

## Inline CSS

```
<p style="color: blue;">This is a paragraph.</p>
```

## Internal CSS

```
<head>
  <style type = text/css>
    body {background-color: blue;}
    p { color: yellow;}
  </style>
</head>
```

## External CSS

```
<head>
  <link rel="stylesheet" type="text/css" href="style.css">
</head>
```

Image 6: Types of CSS

## CSS Units and Values

CSS has several different units for expressing a length. Many CSS properties take "length" values, such as `width`, `margin`, `padding`, `font-size`, etc.

**Length** is a number followed by a length unit, such as `10px`, `2em`, etc.

### Example

Set different length values, using px (pixels):

```
h1 {
  font-size: 60px;
}
```

```
p {
  font-size: 25px;
  line-height: 50px;
}
```

There are two types of length units:

- **absolute**
- **relative**

## Absolute Lengths

The absolute length units are fixed and a length expressed in any of these will appear as exactly that size.

Absolute length units are not recommended for use on screen, because screen sizes vary so much. However, they can be used if the output medium is known, such as for print layout.

Unit	Description
cm	centimeters
mm	millimeters
in	inches (1in = 96px = 2.54cm)
px *	pixels (1px = 1/96th of 1in)
pt	points (1pt = 1/72 of 1in)
pc	picas (1pc = 12 pt)

\* Pixels (px) are relative to the viewing device. For low-dpi devices, 1px is one device pixel (dot) of the display. For printers and high-resolution screens 1px implies multiple device pixels.

## Relative Lengths

Relative length units specify a length relative to another length property. Relative length units scale better between different rendering medium.

Unit	Description

em	Relative to the font-size of the element (2em means 2 times the size of the current font)
ex	Relative to the x-height of the current font (rarely used)
ch	Relative to the width of the "0" (zero)
rem	Relative to font-size of the root element
vw	Relative to 1% of the width of the viewport*
vh	Relative to 1% of the height of the viewport*
vmin	Relative to 1% of viewport's* smaller dimension
vmax	Relative to 1% of viewport's* larger dimension
%	Relative to the parent element

## CSS Styling Text

CSS has a lot of properties for formatting text.

### Text Formatting

This text is styled with some of the text formatting properties. The heading uses the text-align, text-transform, and color properties. The paragraph is indented, aligned, and the space between characters is specified.

## Text Color

The `color` property is used to set the color of the text.

The color is specified by:

- a color name - like "red"
- a HEX value - like "#ff0000"
- an RGB value - like "rgb(255,0,0)"

Look at [CSS Color Values](#) for a complete list of possible color values.

The default text color for a page is defined in the body selector.

### Example

```
body {  
  color: blue;  
}
```

```
h1 {  
  color: green;  
}
```

## Text Color and Background Color

In this example, we define both the `background-color` property and the `color` property:

### Example

```
body {  
  background-color: lightgrey;  
  color: blue;  
}
```

```
h1 {  
  background-color: black;  
  color: white;  
}  
div { background-color: blue; color: white; }
```

## The CSS Text Color Property

Property	Description
<u>color</u>	Specifies the color of text

## CSS Text Alignment and Text Direction

In this chapter you will learn about the following properties:

- text-align
- text-align-last
- direction
- Unicode-bidi
- vertical-align

### Text Alignment

The text-align property is used to set the horizontal alignment of a text. A text can be left or right aligned, centered, or justified.

The following example shows center aligned, and left and right aligned text (left alignment is default if text direction is left-to-right, and right alignment is default if text direction is right-to-left):

#### Example

```
h1 {  
  text-align: center;  
}
```

```
h2 {  
  text-align: left;  
}
```

```
h3 {  
  text-align: right;  
}
```

When the `text-align`, property is set to "justify", each line is stretched so that every line has equal width, and the left and right margins are straight (like in magazines and newspapers):

### **Example**

```
div {  
  text-align: justify;  
}
```

### **Text Align Last**

The `text-align-last` property specifies how to align the last line of a text.

### **Example**

Align the last line of text in three `<p>` elements:

```
p.a {  
  text-align-last: right;  
}
```

```
p.b {  
  text-align-last: center;  
}
```

```
p.c {  
  text-align-last: justify;  
}
```

### **Text Direction**

The `direction` and `unicode-bidi` properties can be used to change the text direction of an element:

### **Example**

```
p {  
  direction: rtl;  
  unicode-bidi: bidi-override;  
}
```

## Vertical Alignment

The `vertical-align` property sets the vertical alignment of an element.

### Example

Set the vertical alignment of an image in a text:

```
img.a {  
    vertical-align: baseline;  
}
```

```
img.b {  
    vertical-align: text-top;  
}
```

```
img.c {  
    vertical-align: text-bottom;  
}
```

```
img.d {  
    vertical-align: sub;  
}
```

```
img.e {  
    vertical-align: super;  
}
```

## The CSS Text Alignment/Direction Properties

Property	Description
<u>direction</u>	Specifies the text direction/writing direction
<u>text-align</u>	Specifies the horizontal alignment of text
<u>text-align-last</u>	Specifies how to align the last line of a text

<b><u>unicode-bidi</u></b>	Used together with the <code>direction</code> property to set or return whether the text should be overridden to support multiple languages in the same document
<b><u>vertical-align</u></b>	Sets the vertical alignment of an element

## CSS Text Decoration

In this chapter you will learn about the following properties:

- `text-decoration-line`
- `text-decoration-color`
- `text-decoration-style`
- `text-decoration-thickness`
- `text-decoration`

### Add a Decoration Line to Text

The `text-decoration-line` property is used to add a decoration line to text.

**Tip:** You can combine more than one value, like overline and underline to display lines both over and under a text.

#### Example

```
h1 {  
  text-decoration-line: overline;  
}  
  
h2 {  
  text-decoration-line: line-through;  
}  
  
h3 {  
  text-decoration-line: underline;  
}  
  
p {
```

```
text-decoration-line: overline underline;  
}
```

## Specify a Color for the Decoration Line

The `text-decoration-color` property is used to set the color of the decoration line.

### Example

```
h1 {  
  text-decoration-line: overline;  
  text-decoration-color: red;  
}
```

```
h2 {  
  text-decoration-line: line-through;  
  text-decoration-color: blue;  
}
```

```
h3 {  
  text-decoration-line: underline;  
  text-decoration-color: green;  
}
```

```
p {  
  text-decoration-line: overline underline;  
  text-decoration-color: purple;  
}
```

## Specify a Style for the Decoration Line

The `text-decoration-style` property is used to set the style of the decoration line.

### Example

```
h1 {  
  text-decoration-line: underline;  
  text-decoration-style: solid;  
}
```

```
h2 {  
  text-decoration-line: underline;  
  text-decoration-style: double;
```

```
}
```

```
h3 {  
  text-decoration-line: underline;  
  text-decoration-style: dotted;  
}
```

```
p.ex1 {  
  text-decoration-line: underline;  
  text-decoration-style: dashed;  
}
```

```
p.ex2 {  
  text-decoration-line: underline;  
  text-decoration-style: wavy;  
}
```

```
p.ex3 {  
  text-decoration-line: underline;  
  text-decoration-color: red;  
  text-decoration-style: wavy;  
}
```

## Specify the Thickness for the Decoration Line

The `text-decoration-thickness` property is used to set the thickness of the decoration line.

### Example

```
h1 {  
  text-decoration-line: underline;  
  text-decoration-thickness: auto;  
}
```

```
h2 {  
  text-decoration-line: underline;  
  text-decoration-thickness: 5px;  
}
```

```
h3 {  
  text-decoration-line: underline;  
  text-decoration-thickness: 25%;
```

```
}
```

```
p {  
  text-decoration-line: underline;  
  text-decoration-color: red;  
  text-decoration-style: double;  
  text-decoration-thickness: 5px;  
}
```

## The Shorthand Property

The `text-decoration` property is a shorthand property for:

- `text-decoration-line` (required)
- `text-decoration-color` (optional)
- `text-decoration-style` (optional)
- `text-decoration-thickness` (optional)

### Example

```
h1 {  
  text-decoration: underline;  
}
```

```
h2 {  
  text-decoration: underline red;  
}
```

```
h3 {  
  text-decoration: underline red double;  
}
```

```
p {  
  text-decoration: underline red double 5px;  
}
```

## All CSS text-decoration Properties

Property	Description
<u><a href="#">text-decoration</a></u>	Sets all the text-decoration properties in one declaration

<u>text-decoration-color</u>	Specifies the color of the text-decoration
<u>text-decoration-line</u>	Specifies the kind of text decoration to be used (underline, overline, etc.)
<u>text-decoration-style</u>	Specifies the style of the text decoration (solid, dotted, etc.)
<u>text-decoration-thickness</u>	Specifies the thickness of the text decoration line

## Text Transformation

The `text-transform` property is used to specify uppercase and lowercase letters in a text.

It can be used to turn everything into uppercase or lowercase letters, or capitalize the first letter of each word:

### Example

```
p.uppercase {  
  text-transform: uppercase;  
}
```

```
p.lowercase {  
  text-transform: lowercase;  
}
```

```
p.capitalize {  
  text-transform: capitalize;  
}
```

### The CSS Text Transformation Property

Property	Description
<u>text-transform</u>	Controls the capitalization of text

## CSS Text Indentation, Letter Spacing, Line Height, Word Spacing, and White Space

In this chapter you will learn about the following properties:

- text-indent
- letter-spacing
- line-height
- word-spacing
- white-space

### Text Indentation

The `text-indent` property is used to specify the indentation of the first line of a text:

#### Example

```
p {  
  text-indent: 50px;  
}
```

### Letter Spacing

The `letter-spacing` property is used to specify the space between the characters in a text. The following example demonstrates how to increase or decrease the space between characters:

#### Example

```
h1 {  
  letter-spacing: 5px;  
}
```

```
h2 {  
  letter-spacing: -2px;  
}
```

### Line Height

The `line-height` property is used to specify the space between lines:

#### Example

```
p.small {  
  line-height: 0.8;  
}
```

```
p.big {  
  line-height: 1.8;  
}
```

## Word Spacing

The `word-spacing` property is used to specify the space between the words in a text. The following example demonstrates how to increase or decrease the space between words:

### Example

```
p.one {  
  word-spacing: 10px;  
}
```

```
p.two {  
  word-spacing: -2px;  
}
```

## White Space

The `white-space` property specifies how white-space inside an element is handled. This example demonstrates how to disable text wrapping inside an element:

### Example

```
p {  
  white-space: nowrap;  
}
```

## The CSS Text Spacing Properties

Property	Description
<u>letter-spacing</u>	Specifies the space between characters in a text
<u>line-height</u>	Specifies the line height
<u>text-indent</u>	Specifies the indentation of the first line in a text-block

<u>white-space</u>	Specifies how to handle white-space inside an element
<u>word-spacing</u>	Specifies the space between words in a text

## Text Shadow

The `text-shadow` property adds shadow to text. In its simplest use, you only specify the horizontal shadow (2px) and the vertical shadow (2px):

Text shadow effect!

### Example

```
h1 {  
  text-shadow: 2px 2px;  
}
```

Next, add a color (red) to the shadow:

Text shadow effect!

### Example

```
h1 {  
  text-shadow: 2px 2px red;  
}
```

Then, add a blur effect (5px) to the shadow:

## Text shadow effect

### Example

```
h1 {  
  text-shadow: 2px 2px 5px red;  
}
```

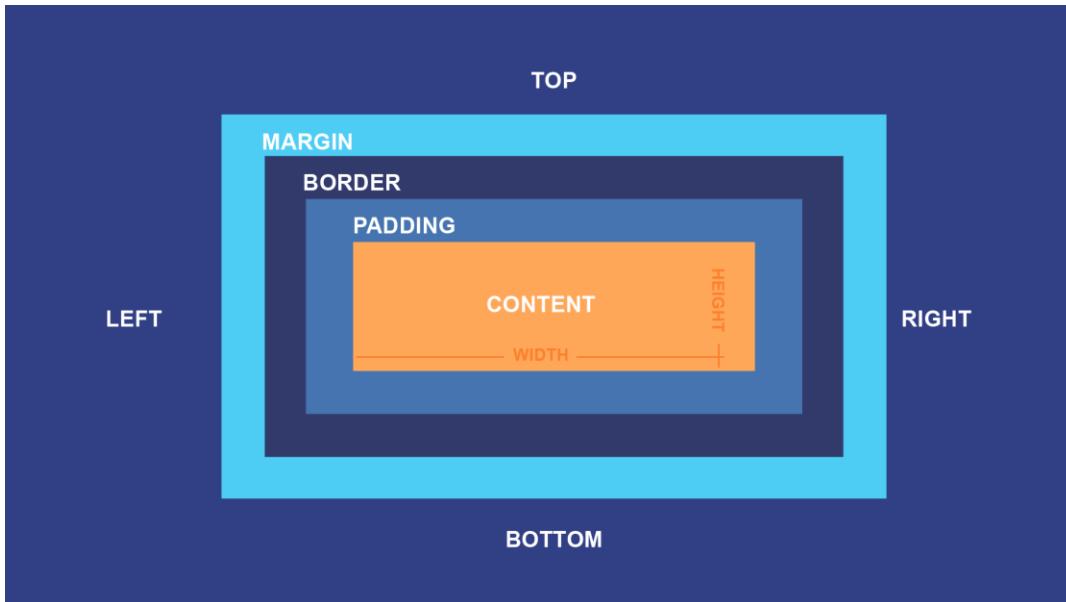
## Styling Box

### CSS Box Model

In CSS, the term "box model" is used when talking about design and layout.

The CSS box model is essentially a box that wraps around every HTML element. It consists of: margins, borders, padding, and the actual content. The image below illustrates the box model:

Explanation of the different parts:



- **Content** - The content of the box, where text and images appear
- **Padding** - Clears an area around the content. The padding is transparent
- **Border** - A border that goes around the padding and content
- **Margin** - Clears an area outside the border. The margin is transparent

The box model allows us to add a border around elements, and to define space between elements.

### Example

Demonstration of the box model:

```
div
```

```
{  
width: 300px;  
border: 15px solid green;  
padding: 50px;  
margin: 20px;  
}
```

## Width and Height of an Element

In order to set the width and height of an element correctly in all browsers, you need to know how the box model works.

Important: When you set the width and height properties of an element with CSS, you just set the width and height of the content area. To calculate the full size of an element, you must also add padding, borders and margins.

### Example

This `<div>` element will have a total width of 350px:

```
div {  
    width: 320px;  
    padding: 10px;  
    border: 5px solid gray;  
    margin: 0;  
}
```

### Here is the calculation:

320px (width)  
+ 20px (left + right padding)  
+ 10px (left + right border)  
+ 0px (left + right margin)  
**= 350px**

The total width of an element should be calculated like this:

Total element width = width + left padding + right padding + left border + right border + left margin + right margin

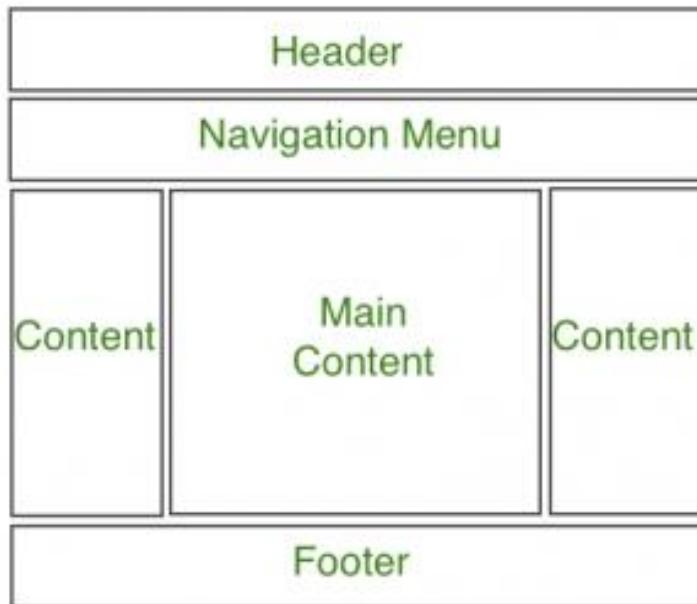
The total height of an element should be calculated like this:

Total element height = height + top padding + bottom padding + top border + bottom border + top margin + bottom margin

## Website Layout

A website can be divided into various sections comprising of header, menus, content and footer based on which there are many different layout designs available for developer. Different layouts

can be created by using div tag and use CSS property to style it.  
The most common structure of website layout is given below:



Notice: Header section contains a website logo, a search bar and profile of user. The navigation menu contains link to various categories of articles available and content section is divided into 3 parts(columns) with left and right sidebar containing links to other articles and advertisements whereas the main content section is the one containing this article, then at the bottom there is a footer section which contains address, links, contacts etc.

**Header Section:** The header section is generally placed either at the top of the Website or just below a top navigation menu. It often comprises of the name of the Website or the logo of the Website.

**Example:**

```
<!-- This code describes the header section  
of website layout -->  
<!DOCTYPE html>  
<html>
```

```
<head>

    <title>
        Website Layouts
    </title>

    <style>
        .header {
            background-color: green;
            padding: 15px;
            text-align: center;
        }
    </style>

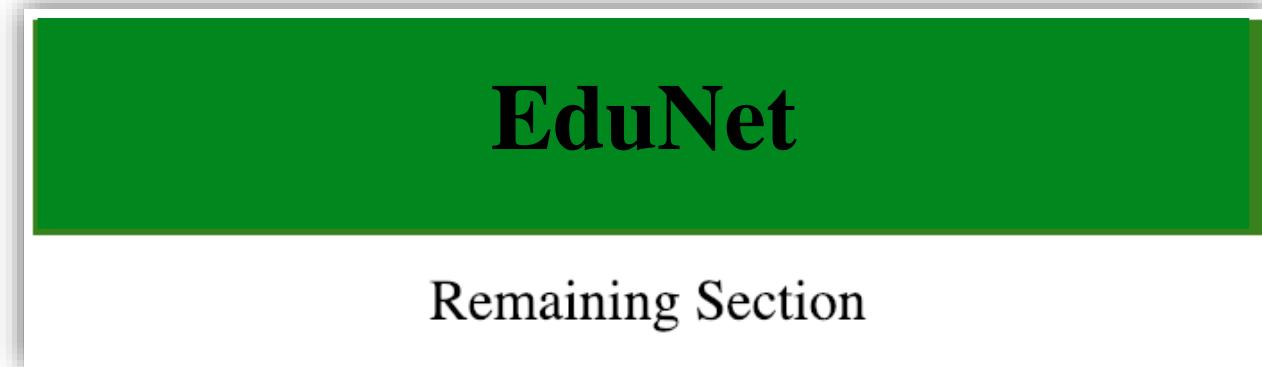
</head>

<body>

    <div class = "header">
        <h2 style = "color:white;">
            EduNet
        </h2>
    </div>
    <br>

    <center style="font-size:200%; ">
        Remaining Section
    </center>
</body>

</html>
```

**Output:**

**Navigation Menu:** A Navigation Bar/Menu is basically a list of links that allows visitor to navigate through the website comfortably with easy access.

**Example:**

```
<!DOCTYPE html>
<html>
  <head>
    <title>
      Website Layout
    </title>

    <style>

      /* CSS property for header section */
      .header {
        background-color: green;
        padding: 15px;
        text-align: center;
      }

      /* CSS property for navigation menu */
      .nav_menu {
        overflow: hidden;

        background-color: #333;
      }

    </style>
  </head>
  <body>
    <div class="header">
      <h1>EduNet</h1>
    </div>
    <div class="nav_menu">
      <ul>
        <li>Home</li>
        <li>About</li>
        <li>Services</li>
        <li>Contact</li>
      </ul>
    </div>
    <div>
      <h2>Remaining Section</h2>
    </div>
  </body>
</html>
```

```
}

.nav_menu a {

    float: left;

    display: block;

    color: white;

    text-align: center;

    padding: 14px 16px;

    text-decoration: none;

}

.nav_menu a:hover {

    background-color: white;

    color: green;

}

</style>

</head>

<body>

<!-- header of website layout -->

<div class = "header">

<h2 style = "color:white;font-size:200%;">

    EduNet


```

```
</h2>

</div>

<!-- navigation menu for website layout -->

<div class = "nav_menu">

    <a href = "#">Algo</a>

    <a href = "#">DS</a>

    <a href = "#">Language</a>

</div><br>

<center style = "font-size:200%;">

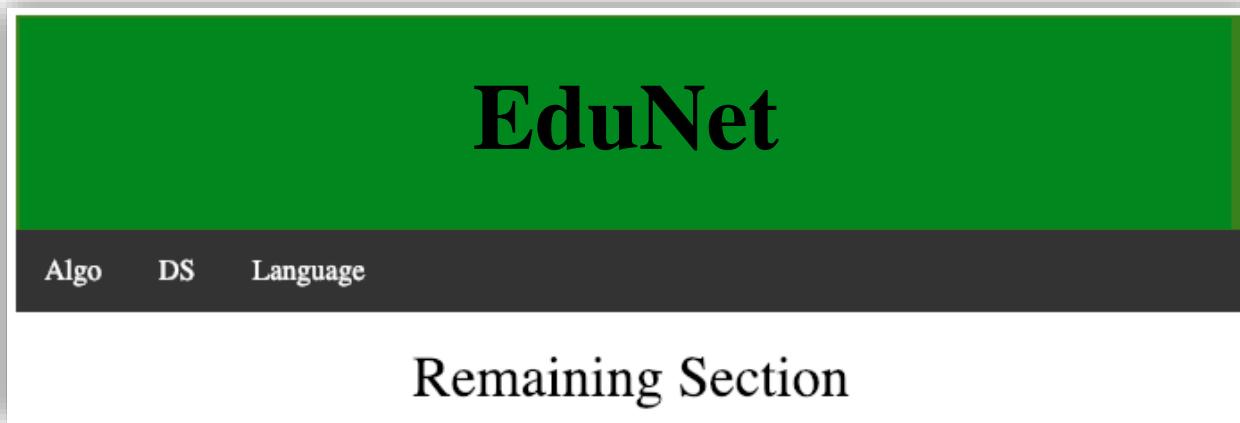
    Remaining Section

</center>

</body>

</html>
```

**Output:**

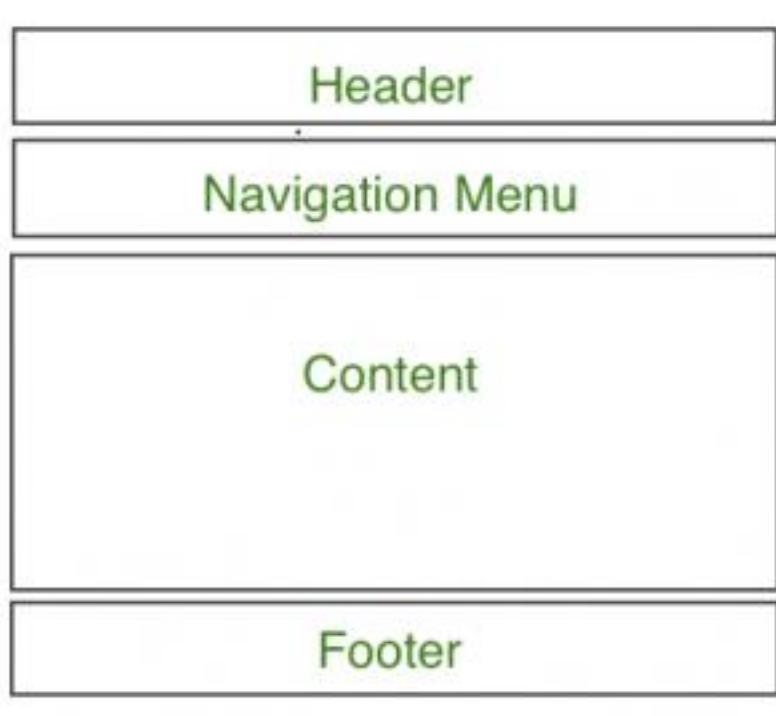


**Content Section:** The content section is the main body of the website.

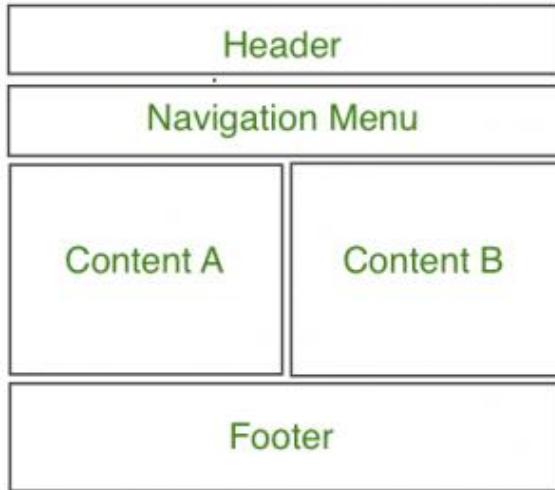
The user can divide content section in n-column layout.

The most common layouts are:

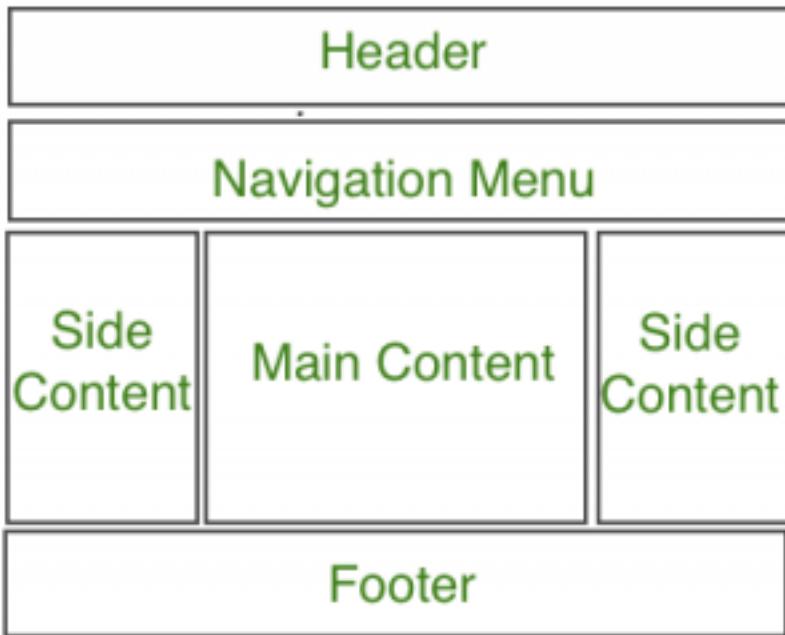
**1-Column Layout:** It is mostly used for mobile layout.



**2-Column Layout:** This website layout is mostly used for tablets or laptops.



**3-Column Layout:** This website layout is mostly used for desktops.



The user can also create a responsive layout where the layout will get changed as per screen size. Consider the below example where if width of screen is more than 600px then there will be 3-column layout and if width of screen is between 400px to 600px then there will be 2-column layout and if screen size less than 400px then 1-column layout will display.

## Example:

```
<!DOCTYPE html>

<html>

<head>

<title>

    Website Layout

</title>

<style>

    * {

        box-sizing: border-box;
    }

    /* CSS property for header section */

    .header {

        background-color: green;
        padding: 15px;
        text-align: center;
    }

    /* CSS property for navigation menu */

    .nav_menu {

        overflow: hidden;
    }
}
```

```
background-color: #333;  
}  
  
.nav_menu a {  
  
    float: left;  
  
    display: block;  
  
    color: white;  
  
    text-align: center;  
  
    padding: 14px 16px;  
  
    text-decoration: none;  
}  
  
.nav_menu a:hover {  
  
    background-color: white;  
  
    color: green;  
}  
  
/* CSS property for content section */  
  
.columnA, .columnB, .columnC {  
  
    float: left;  
  
    width: 31%;  
  
    padding: 15px;  
  
    text-align: justify;  
}
```

```
h2 {  
    color:green;  
    text-align:center;  
}  
  
/* Media query to set website layout  
according to screen size */  
  
@media screen and (max-width:600px) {  
    .columnA, .columnB, .columnC {  
        width: 50%;  
    }  
}  
  
@media screen and (max-width:400px) {  
    .columnA, .columnB, .columnC {  
        width: 100%;  
    }  
}  
    }  
    </style>  
    </head>  
  
<body>  
    <!-- header of website layout -->
```

```
<div class = "header">

    <h2 style = "color:white;font-size:200%">
        EduNet
    </h2>

</div>

<!-- navigation menu of website layout -->

<div class = "nav_menu">

    <a href = "#">Algo</a>
    <a href = "#">DS</a>
    <a href = "#">Language</a>

</div>

<!-- Content section of website layout -->

<div class = "row">

    <div class = "columnA">
        <h2>Column A</h2>
    </div>

    <div class = "columnB">
        <h2>Column B</h2>
    </div>

</div>

<p>Prepare for the Recruitment drive of product
    based companies like Microsoft, Amazon, Adobe
</p>
```

etc with a free online placement preparation course. The course focuses on various MCQ's & Coding question likely to be asked in the interviews & make your upcoming placement season efficient and successful.</p>

</div>

<div class = "columnB">  
<h2>Column B</h2>

<p>Prepare for the Recruitment drive of product based companies like Microsoft, Amazon, Adobe etc with a free online placement preparation course. The course focuses on various MCQ's & Coding question likely to be asked in the interviews & make your upcoming placement season efficient and successful.</p>

</div>

```
<div class = "columnC">  
    <h2>Column C</h2>  
  
    <p>Prepare for the Recruitment drive of product  
        based companies like Microsoft, Amazon, Adobe  
        etc with a free online placement preparation  
        course. The course focuses on various MCQ's  
        & Coding question likely to be asked in the  
        interviews & make your upcoming placement  
        season efficient and successful.</p>  
  
    </div>  
    </div>  
    </body>  
</html>
```

**Output:****The width of screen size greater then 700px:****Column A**

Prepare for the Recruitment drive of product based companies like Microsoft, Amazon, Adobe etc with a free online placement preparation course. The course focuses on various MCQ's & Coding question likely to be asked in the interviews & make your upcoming placement season efficient and successful.

**Column B**

Prepare for the Recruitment drive of product based companies like Microsoft, Amazon, Adobe etc with a free online placement preparation course. The course focuses on various MCQ's & Coding question likely to be asked in the interviews & make your upcoming placement season efficient and successful.

**Column C**

Prepare for the Recruitment drive of product based companies like Microsoft, Amazon, Adobe etc with a free online placement preparation course. The course focuses on various MCQ's & Coding question likely to be asked in the interviews & make your upcoming placement season efficient and successful.

**The width of screen size greater then 400px and less then 600px:**

# EduNet

Algo   DS   Language

Column A	Column B
Prepare for the Recruitment drive of product based companies like Microsoft, Amazon, Adobe etc with a free online placement preparation course. The course focuses on various MCQ's & Coding question likely to be asked in the interviews & make your upcoming placement season efficient and successful.	Prepare for the Recruitment drive of product based companies like Microsoft, Amazon, Adobe etc with a free online placement preparation course. The course focuses on various MCQ's & Coding question likely to be asked in the interviews & make your upcoming placement season efficient and successful.
Column C	Prepare for the Recruitment drive of product based companies like Microsoft, Amazon, Adobe etc with a free online placement preparation course. The course focuses on various MCQ's & Coding question likely to be asked in the interviews & make your upcoming placement season efficient and successful.

**The width of screen size less then 400px:**

# EduNet

Algo      DS      Language

## Column A

Prepare for the Recruitment drive of product based companies like Microsoft, Amazon, Adobe etc with a free online placement preparation course. The course focuses on various MCQ's & Coding question likely to be asked in the interviews & make your upcoming placement season efficient and successful.

## Column B

Prepare for the Recruitment drive of product based companies like Microsoft, Amazon,

**Footer Section:** A footer section is placed at the bottom of the webpage and it generally consists of information like contact info, copyrights, About us etc.

**Example:**

```
<!DOCTYPE html>

<html>

<head>

    <title>

        CSS Website Layout

    </title>

    <style>

        /* Style for footer section */

        .footer {

            background-color: green;

            padding: 15px;

            text-align: center;

        }

    </style>

</head>

<body>

    <center style = "font-size:200%;">

        Upper section

    </center>

    <!-- footer Section -->

    <div class = "footer">

        <a href = "#">About</a><br>


```

```
<a href = "#">Career</a><br>
<a href = "#">Contact Us</a>
</div>
</body>
</html>
```

**Output:**

## Upper section

[About](#)  
[Career](#)  
[Contact Us](#)

### Learning Outcome

After completing this module, a student will be able to:

- Summarize variable naming rules and JavaScript data types.
- Identify expressions and operators.
- Summarize flow control.
- Demonstrate objects and arrays usage
- Define functions and methods.

## Introduction to JavaScript

### What is JavaScript?

JavaScript is a very powerful client-side scripting language. JavaScript is used mainly for enhancing the interaction of a user with the webpage. In other words, you can make your webpage livelier and more interactive, with the help of JavaScript. JavaScript is also being used widely in game development and Mobile application development.

JavaScript is a scripting language that is used to create and manage dynamic web pages, basically anything that moves on your screen without requiring you to refresh your browser. It can be anything from animated graphics to an automatically generated Facebook timeline.

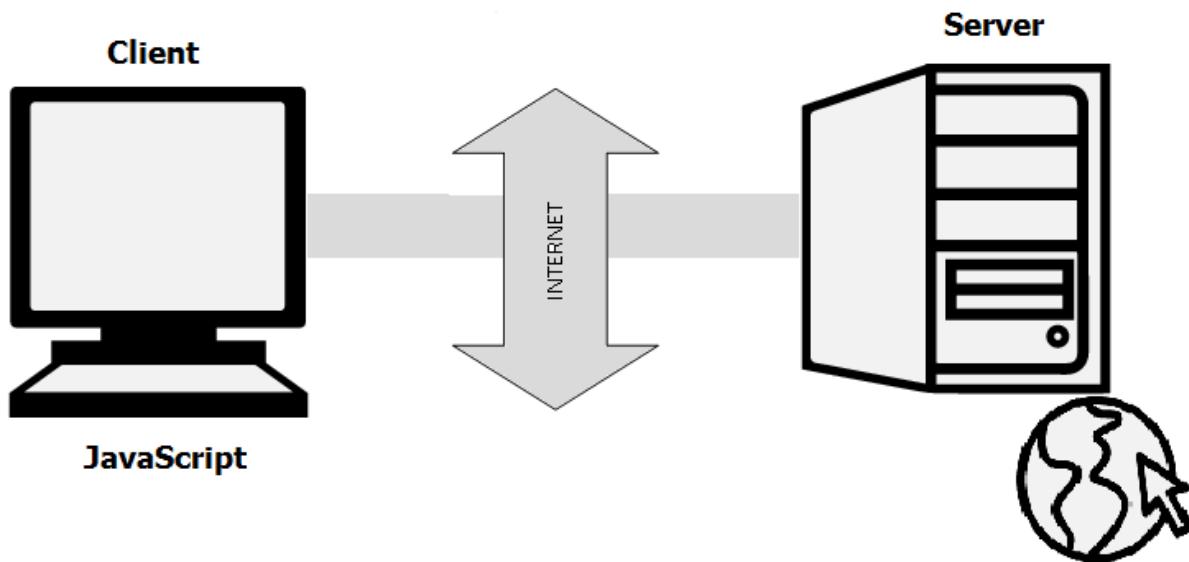


Image 1: what is java script

Reference: [https://www.guru99.com/images/JavaScript/javascript1\\_1.png](https://www.guru99.com/images/JavaScript/javascript1_1.png)

When most people get interested in web development, they start with good old HTML and CSS. From there, they move on to JavaScript, which makes sense, because, these three elements together form the backbone of web development.

**HTML** is the structure of your page like the headers, the body text, any images you want to include. It basically defines the contents of a web page.

**CSS** controls how that page looks (it's what you'll use to customize fonts, background colors, etc.).

**JavaScript** is the third element. Once you've created your structure (HTML) and your aesthetic vibe (CSS), JavaScript makes your site dynamic (automatically updateable).

## Why JavaScript?

JavaScript is an essential programming language, almost compulsory to learn for students or software developers that are gravitated towards web development. Wondering why? Here's the answer:

- JavaScript is the most popular programming language in the world and that makes it a default choice for web development. There are many frameworks available which you can use to create web applications once you have learned JavaScript.
- JavaScript offers lots of flexibility. You can create stunning and fast web applications with tons of customizations to provide users with the most relevant graphical user interface.
- JavaScript is now also used in mobile app development, desktop app development, and game development. This opens many possibilities for you as a JavaScript developer.
- Due to the high demand in the industry, there are tons of job growth opportunities and high pay for those who know JavaScript.
- The incredible thing about JavaScript is that you can find tons of frameworks and libraries already developed, which can be used directly in web development. That reduces the development time and enhances the graphical user interface.

## What is JavaScript Used For?

JavaScript is used in various fields from the web to servers, and here's a quick list of the significant areas it's used in:



Image 2: JavaScript Application

Reference: [https://www.simplilearn.com/ice9/free\\_resources\\_article\\_thumb/js-app.JPG](https://www.simplilearn.com/ice9/free_resources_article_thumb/js-app.JPG)

- **Web Applications:** JavaScript is used for adding interactivity and automation to websites. So, if you want your web application to be anything more than just a static page of contents, you'll probably need to do some "JavaScripting."
- **Mobile Applications:** JavaScript isn't just for developing web applications; it is also used for developing applications for phones and tablets. With frameworks like React Native, you can develop full-fledged mobile applications with all those fancy animations.
- **Web-based Games:** If you've ever played a game directly on the web browser, JavaScript was probably used to make that happen.

- **Back-end Web Development:** JavaScript has traditionally been used for developing the front-end parts of a web application. However, with the introduction of NodeJS, a prevalent back-end JavaScript framework, things have changed. And now, JavaScript is used for developing the back-end structure also.

## Features of JavaScript

There are following features of JavaScript:

- All popular web browsers support JavaScript as they provide built-in execution environments.
- JavaScript follows the syntax and structure of the C programming language. Thus, it is a structured programming language.
- JavaScript is a weakly typed language, where certain types are implicitly cast (depending on the operation).
- JavaScript is an object-oriented programming language that uses prototypes rather than using classes for inheritance.
- It is a light-weighted and interpreted language.
- It is a case-sensitive language.
- JavaScript is supportable in several operating systems including, Windows, macOS, etc.
- It provides good control to the users over the web browsers.

## Application of JavaScript

JavaScript is used to create interactive websites. It is mainly used for:

- Client-side validation,
- Dynamic drop-down menus,
- Displaying date and time,
- Displaying pop-up windows and dialog boxes (like an alert dialog box, confirm dialog box and prompt dialog box),
- Displaying clocks etc.

## JavaScript Example

```
<script>  
document.write("Hello JavaScript by JavaScript");  
</script>
```

JavaScript example is easy to code. JavaScript provides 3 places to put the JavaScript code: within body tag, within head tag and external JavaScript file.

Let's create the first JavaScript example.

```
<script type="text/javascript">  
document.write("JavaScript is a simple language for javatpoint learners");  
</script>
```

The script tag specifies that we are using JavaScript.

The text/javascript is the content type that provides information to the browser about the data. The document.write() function is used to display dynamic content through JavaScript. We will learn about document object in detail later.

### 3 Places to put JavaScript code

- i. Between the body tag of html
- ii. Between the head tag of html
- iii. In .js file (external javaScript)

#### I. JavaScript Example: code between the body tag

In the above example, we have displayed the dynamic content using JavaScript. Let's see the simple example of JavaScript that displays alert dialog box.

```
<script type="text/javascript">  
alert("Hello Javatpoint");  
</script>
```

#### II. JavaScript Example: code between the head tag

Let's see the same example of displaying alert dialog box of JavaScript that is contained inside the head tag.

In this example, we are creating a function msg(). To create function in JavaScript, you need to write function with function\_name as given below.

To call function, you need to work on event. Here we are using onclick event to call msg() function.

```
<html>  
<head>  
<script type="text/javascript">  
function msg(){  
alert("Hello Javatpoint");
```

```
}

</script>

</head>

<body>

<p>Welcome to JavaScript</p>

<form>

<input type="button" value="click" onclick="msg()" />

</form>

</body>

</html>
```

### III. External JavaScript file

We can create external JavaScript file and embed it in many html page. It provides code reusability because single JavaScript file can be used in several html pages.

An external JavaScript file must be saved by .js extension. It is recommended to embed all JavaScript files into a single file. It increases the speed of the webpage.

Let's create an external JavaScript file that prints Hello Javatpoint in a alert dialog box.

message.js

```
function msg(){

  alert("Hello Javatpoint");

}
```

Let's include the JavaScript file into html page. It calls the JavaScript function on button click.

index.html

```
<html>

<head>

<script type="text/javascript" src="message.js"></script>

</head>

<body>
```

```
<p>Welcome to JavaScript</p>
<form>
<input type="button" value="click" onclick="msg()" />
</form>
</body>
</html>
```

## JavaScript Syntax

The syntax of JavaScript is the set of rules that define a correctly structured JavaScript program.

A JavaScript consists of JavaScript statements that are placed within the `<script></script>` HTML tags in a web page, or within the external JavaScript file having `.js` extension.

The following example shows how JavaScript statements look like:

```
var x = 5;
var y = 10;
var sum = x + y;
document.write(sum); // Prints variable value
```

## JavaScript Variables

### What is Variable?

Variables are fundamental to all programming languages. Variables are used to store data, like string of text, numbers, etc. The data or value stored in the variables can be set, updated, and retrieved whenever needed. In general, variables are symbolic names for values.

You can create a variable with the `var` keyword, whereas the assignment operator (`=`) is used to assign value to a variable, like this: `var varName = value;`

### Example:

```
var name = "Peter Parker";
var age = 21;
var isMarried = false;
```

In the above example we have created three variables, first one has assigned with a string value, the second one has assigned with a number, whereas the last one assigned with a boolean value. Variables can hold different types of data, we'll learn about them in later chapter.

In JavaScript, variables can also be declared without having any initial values assigned to them. This is useful for variables which are supposed to hold values like user inputs.

**Example:**

```
// Declaring Variable
var userName;
// Assigning value
userName = "Clark Kent";
```

### Declaring Multiple Variables at Once

In addition, you can also declare multiple variables and set their initial values in a single statement. Each variable are separated by commas, as demonstrated in the following example:

**Example**

```
// Declaring multiple Variables
var name = "Peter Parker", age = 21, isMarried = false;

/* Longer declarations can be written to span
multiple lines to improve the readability */
var name = "Peter Parker",
    age = 21,
    isMarried = false;
```

### Naming Conventions for JavaScript Variables

These are the following rules for naming a JavaScript variable:

- A variable name must start with a letter, underscore (\_), or dollar sign (\$).
- A variable name cannot start with a number.
- A variable name can only contain alpha-numeric characters (A-z, 0-9) and underscores.
- A variable name cannot contain spaces.
- A variable name cannot be a JavaScript keyword or a JavaScript reserved word.

**Note:** Variable names in JavaScript are case sensitive, it means \$myvar and \$myVar are two different variables. So be careful while defining variable names.

## JavaScript Datatype

JavaScript includes data types similar to other programming languages like Java or C#. JavaScript is dynamic and loosely typed language. It means you don't require to specify a type of a variable. A variable in JavaScript can be assigned any type of value, as shown in the following example.

Example: Loosely Typed Variables

```
var myvariable = 1; // numeric value
myvariable = 'one'; // string value
myvariable = 1.1; // decimal value
myvariable = true; // Boolean value
myvariable = null; // null value
```

In the above example, different types of values are assigned to the same variable to demonstrate loosely typed characteristics of JavaScript. Different values 1, 'one', 1.1, true are examples of different data types.

JavaScript includes primitive and non-primitive data types.

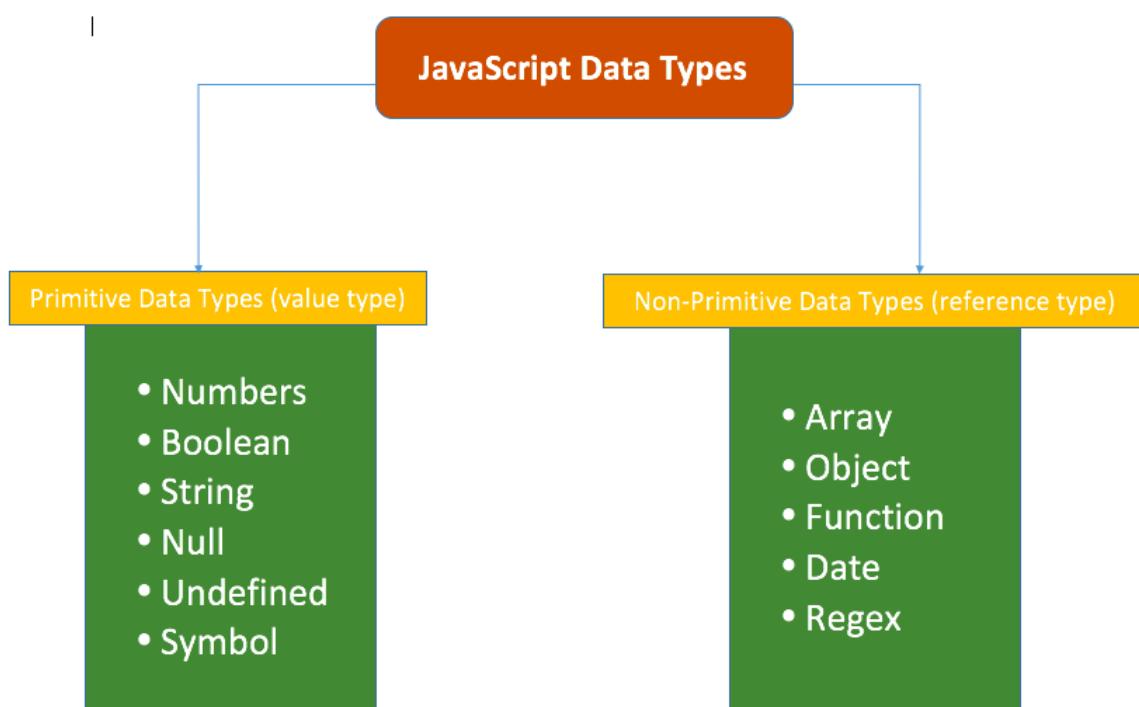


Image 3: JS Datatype  
Reference: [https://miro.medium.com/max/839/1\\*EJVSM8Mtm6kPmPYMTP5gug.png](https://miro.medium.com/max/839/1*EJVSM8Mtm6kPmPYMTP5gug.png)

## Primitive Data Types

The primitive data types are the lowest level of the data value in JavaScript. The `typeof` operator can be used with primitive data types to know the type of a value.

The followings are primitive data types in JavaScript:

Data Type	Description
<b>String</b>	String is a textual content wrapped inside '' or " " or `` (tick sign).  Example: 'Hello World!', "This is a string", etc.
<b>Number</b>	Number is a numeric value.  Example: 100, 4521983, etc.
<b>BigInt</b>	BigInt is a numeric value in the arbitrary precision format.  Example: 453889879865131n, 200n, etc.
<b>Boolean</b>	Boolean is a logical data type that has only two values, true or false.
<b>Null</b>	A null value denotes an absence of value.  Example: var str = null;
<b>Undefined</b>	undefined is the default value of a variable that has not been assigned any value.  Example: In the variable declaration, var str;, there is no value assigned to str. So, the type of str can be check using <code>typeof(str)</code> which will return undefined.

## Non-primitive data types

The non-primitive data types contain some kind of structure with primitive data.

Data Type	Description
Object	<p>An object holds multiple values in terms of properties and methods.</p> <p>Example:</p> <pre>var person = {     firstName: "James",     lastName: "Bond",     age: 15 };</pre>
Date	<p>Date object represents date &amp; time including days, months, years, hours, minutes, seconds and milliseconds.</p> <p>Example: var today = new Date("25 July 2021");</p>
Array	<p>An array stores multiple values using special syntax.</p> <p>Example: var nums = [1, 2, 3, 4];</p>

## JavaScript Operators

JavaScript includes operators same as other languages. An operator performs some operation on single or multiple operands (data value) and produces a result. For example, in  $1 + 2$ , the  $+$  sign is an operator and 1 is left side operand and 2 is right side operand. The  $+$  operator performs the addition of two numeric values and returns a result.

### Syntax:

<Left operand> operator <right operand>

<Left operand> operator

JavaScript includes following categories of operators:

1. Arithmetic Operators
2. Comparison (Relational) Operators
3. Bitwise Operators
4. Logical Operators
5. Assignment Operators
6. Special Operators

## 1. Arithmetic Operators

Arithmetic operators are used to perform arithmetic operations on the operands. The following operators are known as JavaScript arithmetic operators.

Operator	Description	Example
+	Addition	$10+20 = 30$
-	Subtraction	$20-10 = 10$
*	Multiplication	$10*20 = 200$
/	Division	$20/10 = 2$
%	Modulus (Remainder)	$20 \% 10 = 0$
++	Increment	<code>var a=10; a++; Now a = 11</code>
--	Decrement	<code>var a=10; a--; Now a = 9</code>

## 2. Comparison Operators

The JavaScript comparison operator compares the two operands. The comparison operators are as follows:

Operator	Description	Example
==	Is equal to	$10==20 = \text{false}$
==	Identical (equal and of same type)	$10==20 = \text{false}$
!=	Not equal to	$10!=20 = \text{true}$
!=	Not Identical	$20!==20 = \text{false}$
>	Greater than	$20>10 = \text{true}$
>=	Greater than or equal to	$20>=10 = \text{true}$
<	Less than	$20<10 = \text{false}$

<code>&lt;=</code>	Less than or equal to	$20 <= 10 = \text{false}$
--------------------	-----------------------	---------------------------

### 3. Bitwise Operators

The bitwise operators perform bitwise operations on operands. The bitwise operators are as follows:

Operator	Description	Example
<code>&amp;</code>	Bitwise AND	$(10 == 20 \& 20 == 33) = \text{false}$
<code> </code>	Bitwise OR	$(10 == 20   20 == 33) = \text{false}$
<code>^</code>	Bitwise XOR	$(10 == 20 ^ 20 == 33) = \text{false}$
<code>~</code>	Bitwise NOT	$(\sim 10) = -10$
<code>&lt;&lt;</code>	Bitwise Left Shift	$(10 << 2) = 40$
<code>&gt;&gt;</code>	Bitwise Right Shift	$(10 >> 2) = 2$
<code>&gt;&gt;&gt;</code>	Bitwise Right Shift with Zero	$(10 >>> 2) = 2$

### 4. Logical Operators

The following operators are known as JavaScript logical operators.

Operator	Description	Example
<code>&amp;&amp;</code>	Logical AND	$(10 == 20 \&& 20 == 33) = \text{false}$
<code>  </code>	Logical OR	$(10 == 20    20 == 33) = \text{false}$
<code>!</code>	Logical Not	$!(10 == 20) = \text{true}$

### 5. Assignment Operators

The following operators are known as JavaScript assignment operators.

Operator	Description	Example
<code>=</code>	Assign	$10 + 10 = 20$
<code>+=</code>	Add and assign	<code>var a=10; a+=20; Now a = 30</code>
<code>-=</code>	Subtract and assign	<code>var a=20; a-=10; Now a = 10</code>
<code>*=</code>	Multiply and assign	<code>var a=10; a*=20; Now a = 200</code>

<code>/=</code>	Divide and assign	<code>var a=10; a/=2; Now a = 5</code>
<code>%=</code>	Modulus and assign	<code>var a=10; a%-=2; Now a = 0</code>

## 6. Special Operators

The following operators are known as JavaScript special operators.

Operator	Description
<code>(?:)</code>	Conditional Operator returns value based on the condition. It is like if-else.
<code>,</code>	Comma Operator allows multiple expressions to be evaluated as single statement.
<code>delete</code>	Delete Operator deletes a property from the object.
<code>in</code>	In Operator checks if object has the given property
<code>instanceof</code>	checks if the object is an instance of given type
<code>new</code>	creates an instance (object)
<code>typeof</code>	checks the type of object.
<code>void</code>	it discards the expression's return value.
<code>yield</code>	checks what is returned in a generator by the generator's iterator.

## Javascript Condition

### JavaScript If-else

The JavaScript if-else statement is used to execute the code whether condition is true or false. There are three forms of if statement in JavaScript.

- i. If Statement
  - ii. If else statement
  - iii. if else if statement
- i. **If statement**  
It evaluates the content only if expression is true. The signature of JavaScript if statement is given below.

### Syntax:

```
if(expression){  
//content to be evaluated  
}
```

### Flowchart of JavaScript If statement

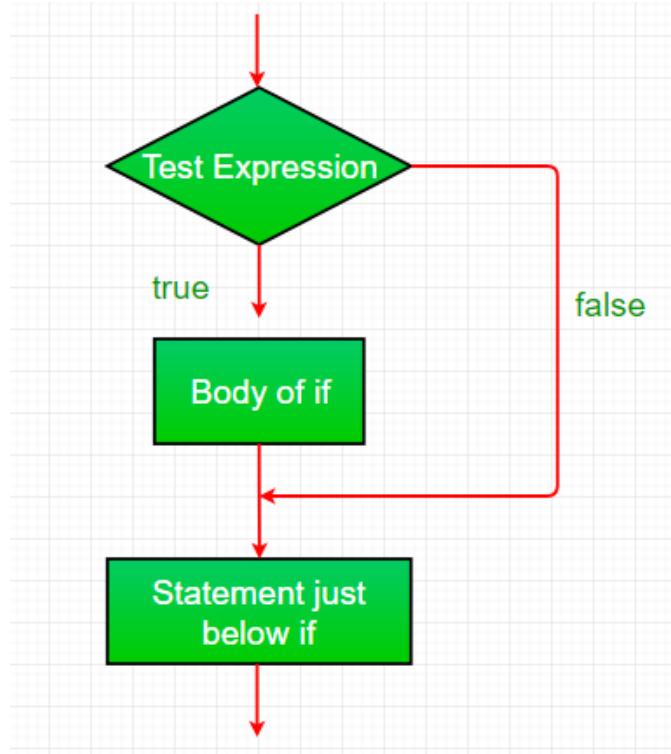


Image 4: if statement  
Reference: <https://media.geeksforgeeks.org/wp-content/uploads/if.png>

Let's see the simple example of if statement in javascript.

```
<script>  
var a=20;  
if(a>10){  
    document.write("value of a is greater than 10");  
}  
</script>
```

Output of the above example

value of a is greater than 10

## ii. If...else Statement

It evaluates the content whether condition is true or false. The syntax of JavaScript if-else statement is given below.

### Syntax

```
if(expression){  
    //content to be evaluated if condition is true  
}  
else{  
    //content to be evaluated if condition is false  
}
```

### Flowchart of JavaScript If...else statement

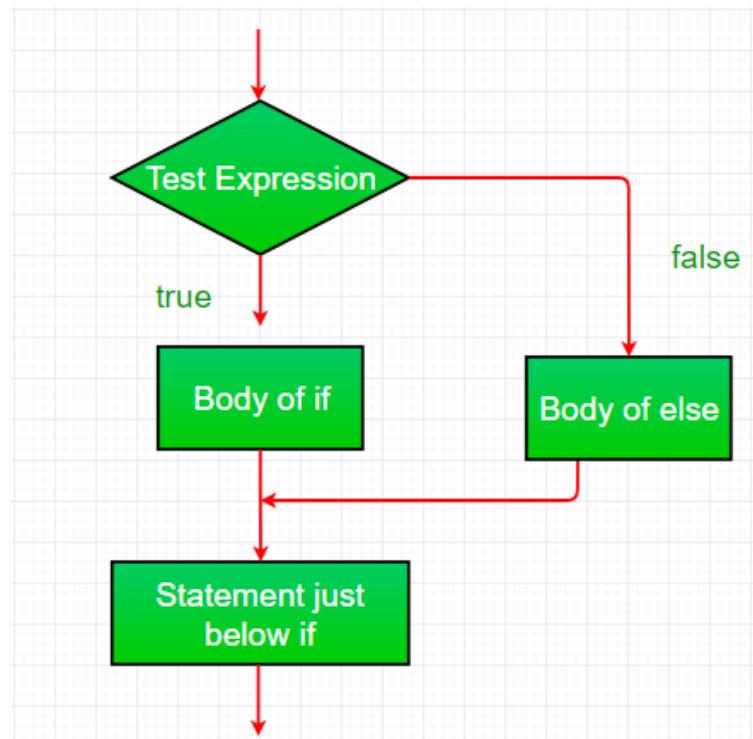


Image 5: if-else

Reference: <https://media.geeksforgeeks.org/wp-content/uploads/if-else.png>

**Let's see the example of if-else statement in JavaScript to find out the even or odd number.**

```
<script>  
var a=20;
```

```
if(a%2==0){  
    document.write("a is even number");  
}  
else{  
    document.write("a is odd number");  
}  
</script>
```

Output of the above example

a is even number

### iii. If...else if statement

It evaluates the content only if expression is true from several expressions. The signature of JavaScript if else if statement is given below.

**Syntax:**

```
if(expression1){  
    //content to be evaluated if expression1 is true  
}  
else if(expression2){  
    //content to be evaluated if expression2 is true  
}  
else if(expression3){  
    //content to be evaluated if expression3 is true  
}  
else{  
    //content to be evaluated if no expression is true  
}
```

**Flowchart of JavaScript If...else statement**

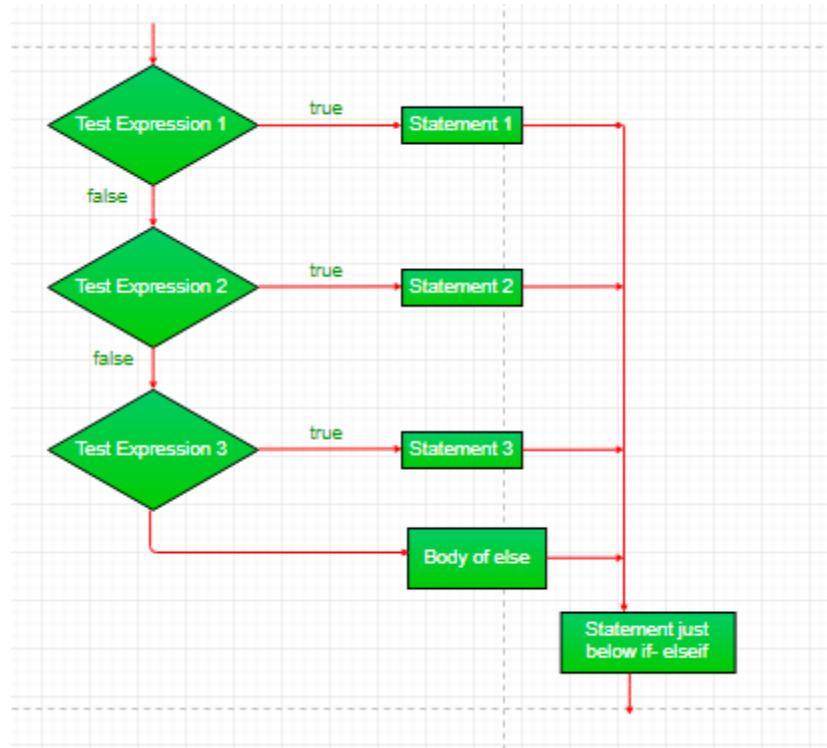


Image 5: if-else--if

Reference: <https://media.geeksforgeeks.org/wp-content/uploads/if-elseif.png>

Let's see the simple example of if else if statement in javascript.

```
<script>
var a=20;
if(a==10){
    document.write("a is equal to 10");
}
else if(a==15){
    document.write("a is equal to 15");
}
else if(a==20){
    document.write("a is equal to 20");
}
else{
```

```
document.write("a is not equal to 10, 15 or 20");
}
</script>
```

Output of the above example

a is equal to 20

### JavaScript Switch

The JavaScript switch statement is used to execute one code from multiple expressions. It is just like else if statement that we have learned in previous page. But it is convenient than if..else..if because it can be used with numbers, characters etc.

#### Syntax:

```
switch(expression){
    case value1:
        code to be executed;
        break;
    case value2:
        code to be executed;
        break;
    .....
    default:
        code to be executed if above values are not matched;
```

code to be executed if above values are not matched;

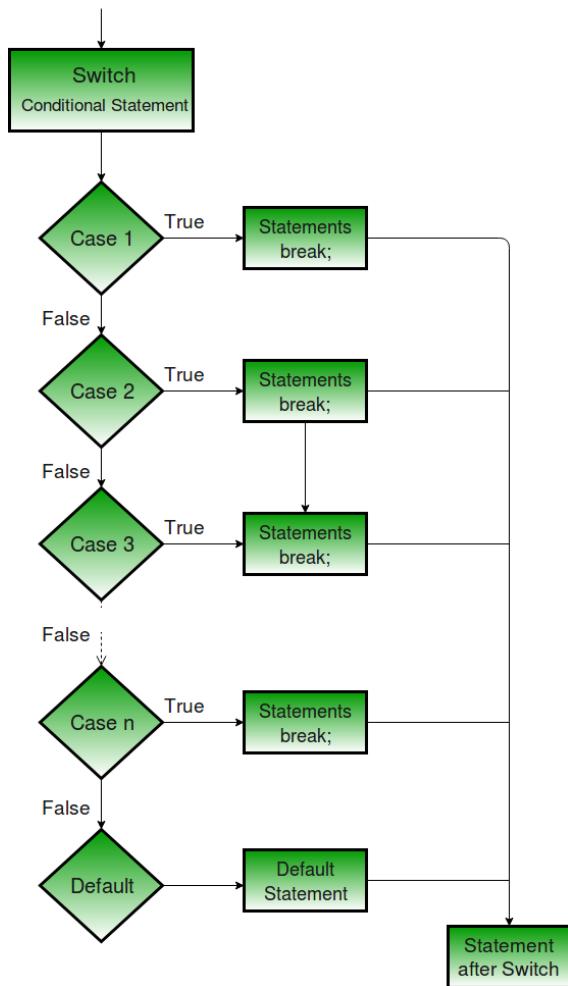


Image: Switch

Reference: <https://media.geeksforgeeks.org/wp-content/uploads/switch.png>

Let's understand the behavior of switch statement in JavaScript.

```
<script>

var grade='B';

var result;

switch(grade){

case 'A':

result+=" A Grade";

case 'B':

result+=" B Grade";

case 'C':
```

```
result+=" C Grade";  
default:  
result+=" No Grade";  
}  
document.write(result);  
</script>
```

Output of the above example

undefined B Grade C Grade No Grade

### **Loop Control statement**

The JavaScript loops are used to iterate the piece of code using for, while, do while or for-in loops. It makes the code compact. It is mostly used in array.

There are four types of loops in JavaScript.

- i. for loop
- ii. while loop
- iii. do-while loop
- iv. for-in loop

### **for loop**

The JavaScript for loop iterates the elements for the fixed number of times. It should be used if number of iterations is known. The syntax of for loop is given below.

#### **Syntax:**

```
for (initialization; condition; increment)  
{  
    code to be executed  
}
```

Let's see the simple example of for loop in javascript.

```
<script>  
for (i=1; i<=5; i++)
```

```
{  
document.write(i + "<br/>")  
}  
</script>
```

Output:

```
1  
2  
3  
4  
5
```

### While loop

The JavaScript while loop iterates the elements for the infinite number of times. It should be used if number of iteration is not known. The syntax of while loop is given below.

#### Syntax:

```
while (condition)  
{  
    code to be executed  
}
```

Let's see the simple example of while loop in javascript.

```
<script>  
var i=11;  
while (i<=15)  
{  
    document.write(i + "<br/>");  
    i++;
```

```
}
```

```
</script>
```

**Output:**

11  
12  
13  
14  
15

**do while loop**

The JavaScript do while loop iterates the elements for the infinite number of times like while loop. but, code is executed at least once whether condition is true or false. The syntax of do while loop is given below.

**Syntax:**

```
do{  
    code to be executed  
}while (condition);
```

Let's see the simple example of do while loop in javascript.

```
<script>  
var i=21;  
do{  
    document.write(i + "<br/>");  
    i++;  
}while (i<=25);  
</script>
```

**Output:**

21

22

23

24

25

## for in loop

JavaScript includes for loop like Java or C#. Use for loop to execute code repeatedly.

Syntax:

```
for(initializer; condition; iteration)
{
    // Code to be executed
}
```

The for loop requires following three parts.

**Initializer:** Initialize a counter variable to start with

**Condition:** specify a condition that must evaluate to true for next iteration

**Iteration:** increase or decrease counter

Let's see the simple example of do for loop in javascript.

```
<script>
// JavaScript program to illustrate for loop
var x;

// for loop begins when x=2
// and runs till x <=4
for (x = 2; x <= 4; x++)
{
    document.write("Value of x:" + x + "<br />");
}
</script>
```

### Output:

Value of x:2

Value of x:3

Value of x:4

## JavaScript Array

JavaScript array is an object that represents a collection of similar type of elements.

There are 3 ways to construct array in JavaScript

- i. By array literal
- ii. By creating instance of Array directly (using new keyword)
- iii. By using an Array constructor (using new keyword)

### i. JavaScript array literal

The syntax of creating array using array literal is given below:

```
var arrayname=[value1,value2.....valueN];
```

As you can see, values are contained inside [ ] and separated by , (comma).

Let's see the simple example of creating and using array in JavaScript.

```
<script>

var emp=["Sonoo","Vimal","Ratan"];

for (i=0;i<emp.length;i++){

document.write(emp[i] + "<br/>");

}

</script>
```

The .length property returns the length of an array.

### Output

Sonoo

Vimal

Ratan

### ii. JavaScript Array directly (new keyword)

The syntax of creating array directly is given below:

```
var arrayname=new Array();
```

Here, new keyword is used to create instance of array.

Let's see the example of creating array directly.

```
<script>

var i;

var emp = new Array();

emp[0] = "Arun";

emp[1] = "Varun";

emp[2] = "John";

for (i=0;i<emp.length;i++){

document.write(emp[i] + "<br>");

}

</script>
```

## Output

Arun

Varun

John

### iii. JavaScript array constructor (new keyword)

Here, you need to create instance of array by passing arguments in constructor so that we don't have to provide value explicitly.

The example of creating object by array constructor is given below.

```
<script>

var emp=new Array("Jai","Vijay","Smith");

for (i=0;i<emp.length;i++){

document.write(emp[i] + "<br>");

}
```

```
| </script>
```

## Output

Jai

Vijay

Smith

## JavaScript Array Methods

Let's see the list of JavaScript array methods with their description.

Methods	Description
<u><a href="#">concat()</a></u>	It returns a new array object that contains two or more merged arrays.
<u><a href="#">copywithin()</a></u>	It copies the part of the given array with its own elements and returns the modified array.
<u><a href="#">entries()</a></u>	It creates an iterator object and a loop that iterates over each key/value pair.
<u><a href="#">every()</a></u>	It determines whether all the elements of an array are satisfying the provided function conditions.
<u><a href="#">flat()</a></u>	It creates a new array carrying sub-array elements concatenated recursively till the specified depth.
<u><a href="#">flatMap()</a></u>	It maps all array elements via mapping function, then flattens the result into a new array.
<u><a href="#">fill()</a></u>	It fills elements into an array with static values.
<u><a href="#">from()</a></u>	It creates a new array carrying the exact copy of another array element.

<u><a href="#">filter()</a></u>	It returns the new array containing the elements that pass the provided function conditions.
<u><a href="#">find()</a></u>	It returns the value of the first element in the given array that satisfies the specified condition.
<u><a href="#">findIndex()</a></u>	It returns the index value of the first element in the given array that satisfies the specified condition.
<u><a href="#">forEach()</a></u>	It invokes the provided function once for each element of an array.
<u><a href="#">includes()</a></u>	It checks whether the given array contains the specified element.
<u><a href="#">indexOf()</a></u>	It searches the specified element in the given array and returns the index of the first match.
<u><a href="#">isArray()</a></u>	It tests if the passed value ia an array.
<u><a href="#">join()</a></u>	It joins the elements of an array as a string.
<u><a href="#">keys()</a></u>	It creates an iterator object that contains only the keys of the array, then loops through these keys.
<u><a href="#">lastIndexOf()</a></u>	It searches the specified element in the given array and returns the index of the last match.
<u><a href="#">map()</a></u>	It calls the specified function for every array element and returns the new array
<u><a href="#">of()</a></u>	It creates a new array from a variable number of arguments, holding any type of argument.
<u><a href="#">pop()</a></u>	It removes and returns the last element of an array.
<u><a href="#">push()</a></u>	It adds one or more elements to the end of an array.

<u>reverse()</u>	It reverses the elements of given array.
<u>reduce(function, initial)</u>	It executes a provided function for each value from left to right and reduces the array to a single value.
<u>reduceRight()</u>	It executes a provided function for each value from right to left and reduces the array to a single value.
<u>some()</u>	It determines if any element of the array passes the test of the implemented function.
<u>shift()</u>	It removes and returns the first element of an array.
<u>slice()</u>	It returns a new array containing the copy of the part of the given array.
<u>sort()</u>	It returns the element of the given array in a sorted order.
<u>splice()</u>	It add/remove elements to/from the given array.
<u>toLocaleString()</u>	It returns a string containing all the elements of a specified array.
<u>toString()</u>	It converts the elements of a specified array into string form, without affecting the original array.
<u>unshift()</u>	It adds one or more elements in the beginning of the given array.
<u>values()</u>	It creates a new iterator object carrying values for each index in the array.

## JavaScript Object

A JavaScript object is an entity having state and behavior (properties and method). For example: car, pen, bike, chair, glass, keyboard, monitor etc.

JavaScript is an object-based language. Everything is an object in JavaScript.

JavaScript is template based not class based. Here, we don't create class to get the object. But, we direct create objects.

### Creating Objects in JavaScript

There are 3 ways to create objects.

- i. By object literal
  - ii. By creating instance of Object directly (using new keyword)
  - iii. By using an object constructor (using new keyword)
- i. JavaScript Object by object literal**

The syntax of creating object using object literal is given below:

```
object={property1:value1,property2:value2.....propertyN:valueN}
```

As you can see, property and value is separated by : (colon).

Let's see the simple example of creating object in JavaScript.

```
<script>  
emp={id:102,name:"Shyam Kumar",salary:40000}  
document.write(emp.id+" "+emp.name+" "+emp.salary);  
</script>
```

#### Output

102 Shyam Kumar 40000

#### ii. By creating instance of Object

The syntax of creating object directly is given below:

```
var objectname=new Object();
```

Here, new keyword is used to create object.

Let's see the example of creating object directly.

```
<script>
```

```
var emp=new Object();
emp.id=101;
emp.name="Ravi Malik";
emp.salary=50000;
document.write(emp.id+" "+emp.name+" "+emp.salary);
</script>
```

## Output

101 Ravi 50000

### iii. By using an Object constructor

Here, you need to create function with arguments. Each argument value can be assigned in the current object by using this keyword.

This keyword refers to the current object.

The example of creating object by object constructor is given below.

```
<script>
function emp(id,name,salary){
this.id=id;
this.name=name;
this.salary=salary;
}
e=new emp(103,"Vimal Jaiswal",30000);
document.write(e.id+" "+e.name+" "+e.salary);
</script>
```

## Output

103 Vimal Jaiswal 30000

### Defining method in JavaScript Object

We can define method in JavaScript object. But before defining method, we need to add property in the function with same name as method.

The example of defining method in object is given below.

```
<script>
function emp(id,name,salary){
```

```
this.id=id;  
this.name=name;  
this.salary=salary;  
  
this.changeSalary=changeSalary;  
function changeSalary(otherSalary){  
this.salary=otherSalary;  
}  
}  
  
e=new emp(103,"Sonoo Jaiswal",30000);  
document.write(e.id+" "+e.name+" "+e.salary);  
e.changeSalary(45000);  
document.write("<br>"+e.id+" "+e.name+" "+e.salary);  
</script>
```

Output:

103 Sonoo Jaiswal 30000

103 Sonoo Jaiswal 45000

### JavaScript Object Methods

The various methods of Object are as follows:

Sr.No	Methods	Description
1	<a href="#"><u>Object.assign()</u></a>	This method is used to copy enumerable and own properties from a source object to a target object

2	<a href="#">Object.create()</a>	This method is used to create a new object with the specified prototype object and properties.
3	<a href="#">Object.defineProperty()</a>	This method is used to describe some behavioral attributes of the property.
4	<a href="#">Object.defineProperties()</a>	This method is used to create or configure multiple object properties.
5	<a href="#">Object.entries()</a>	This method returns an array with arrays of the key, value pairs.
6	<a href="#">Object.freeze()</a>	This method prevents existing properties from being removed.
7	<a href="#">Object.getOwnPropertyDescriptor()</a>	This method returns a property descriptor for the specified property of the specified object.
8	<a href="#">Object.getOwnPropertyDescriptors()</a>	This method returns all own property descriptors of a given object.
9	<a href="#">Object.getOwnPropertyNames()</a>	This method returns an array of all properties (enumerable or not) found.
10	<a href="#">Object.getOwnPropertySymbols()</a>	This method returns an array of all own symbol key properties.
11	<a href="#">Object.getPrototypeOf()</a>	This method returns the prototype of the specified object.
12	<a href="#">Object.is()</a>	This method determines whether two values are the same value.
13	<a href="#">Object.isExtensible()</a>	This method determines if an object is extensible

14	<a href="#">Object.isFrozen()</a>	This method determines if an object was frozen.
15	<a href="#">Object.isSealed()</a>	This method determines if an object is sealed.
16	<a href="#">Object.keys()</a>	This method returns an array of a given object's own property names.
17	<a href="#">Object.preventExtensions()</a>	This method is used to prevent any extensions of an object.
18	<a href="#">Object.seal()</a>	This method prevents new properties from being added and marks all existing properties as non-configurable.
19	<a href="#">Object.setPrototypeOf()</a>	This method sets the prototype of a specified object to another object.
20	<a href="#">Object.values()</a>	This method returns an array of values.

## JavaScript Function

A function is a block of code that performs a specific task.

Suppose you need to create a program to create a circle and color it. You can create two functions to solve this problem:

- a function to draw the circle
- a function to color the circle

Dividing a complex problem into smaller chunks makes your program easy to understand and reusable.

JavaScript also has a huge number of inbuilt functions. For example, `Math.sqrt()` is a function to calculate the square root of a number.

## User-defined functions.

### Declaring a Function

The syntax to declare a function is:

```
function nameOfFunction () {  
    // function body  
}
```

- A function is declared using the function keyword.
- The basic rules of naming a function are similar to naming a variable. It is better to write a descriptive name for your function. For example, if a function is used to add two numbers, you could name the function add or addNumbers.
- The body of function is written within {}.

For example,

```
// declaring a function named greet()  
  
function greet() {  
  
    console.log("Hello there");  
  
}
```

### Calling a Function

In the above program, we have declared a function named greet(). To use that function, we need to call it.

Here's how you can call the above greet() function.

```
// function call
```

```
greet();
```

```
function greet() { ←  
  // code  
}  
  
greet(); ←  
// code
```

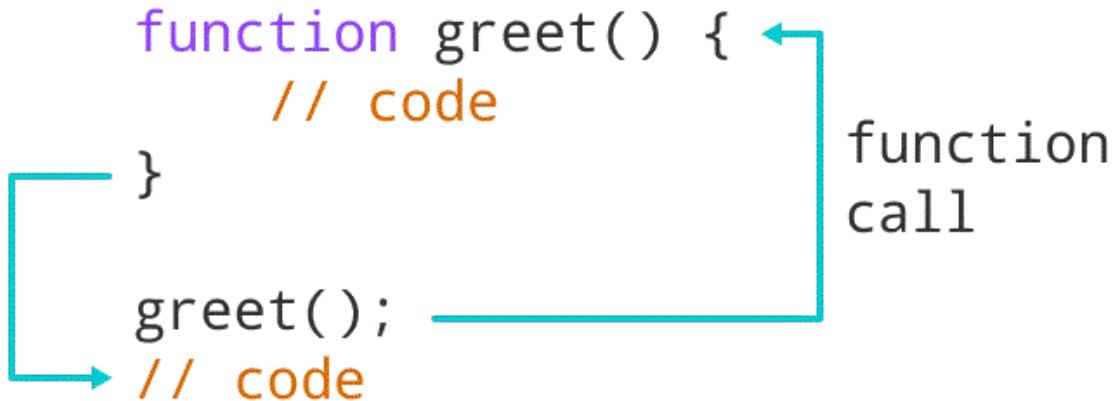


Image: Working of a Function in JavaScript

Reference:<https://cdn.programiz.com/cdn/farfuture/NdxxeWIRfoHMPgdcWPkeVyIwN9MwAgqoYqZkFQDMFQ/mtime:1591592059/sites/tutorial2program/files/javascript-function-example1.png>

### Example 1: Display a Text

```
// program to print a text  
  
// declaring a function  
  
function greet() {  
  
  console.log("Hello there!");  
  
}  
  
// calling the function  
  
greet();
```

Output

Hello there!

## Function Parameters

A function can also be declared with parameters. A parameter is a value that is passed when declaring a function.

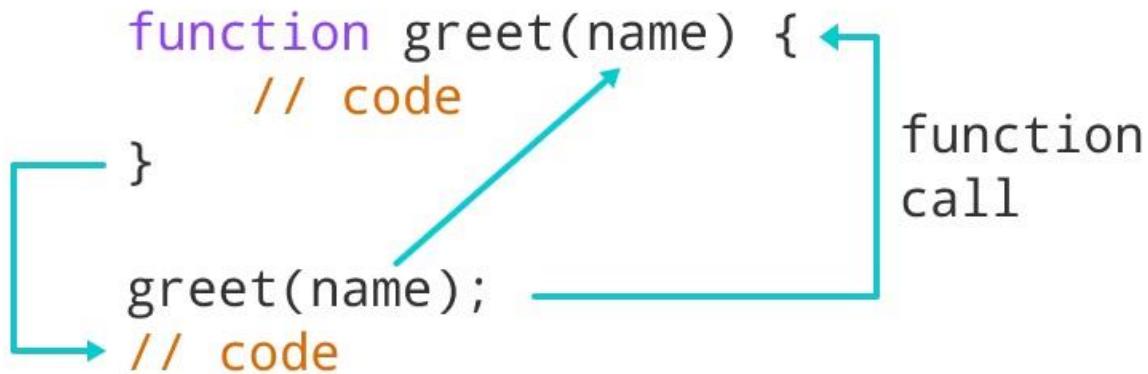


Image: Working of JavaScript Function with parameter

Reference: [https://cdn.programiz.com/cdn/farfuture/oAZVf3IqOKOYj\\_aJ-IoYQvbJ2CB-B3y4HXSLXBUmYcY/mtime:1591592163/sites/tutorial2program/files/javascript-function-with-parameter.png](https://cdn.programiz.com/cdn/farfuture/oAZVf3IqOKOYj_aJ-IoYQvbJ2CB-B3y4HXSLXBUmYcY/mtime:1591592163/sites/tutorial2program/files/javascript-function-with-parameter.png)

### Example 2: Function with Parameters

```
// program to print the text  
  
// declaring a function  
  
function greet(name) {  
  
    console.log("Hello " + name + ":");  
  
}
```

```
// variable name can be different  
  
let name = prompt("Enter a name: ");  
  
// calling function  
  
greet(name);
```

### Output

Enter a name: Simon  
Hello Simon :)

In the above program, the greet function is declared with a name parameter. The user is prompted to enter a name. Then when the function is called, an argument is passed into the function.

**Note:** When a value is passed when declaring a function, it is called parameter. And when the function is called, the value passed is called argument.

Example 3: Add Two Numbers

```
// program to add two numbers using a function
// declaring a function
function add(a, b) {
  console.log(a + b);
}
// calling functions
add(3,4);
add(2,9);
```

Output

7

11

In the above program, the add function is used to find the sum of two numbers.

- The function is declared with two parameters a and b.
- The function is called using its name and passing two arguments 3 and 4 in one and 2 and 9 in another.

Notice that you can call a function as many times as you want. You can write one function and then call it multiple times with different arguments.

## Function Return

The return statement can be used to return the value to a function call.

The return statement denotes that the function has ended. Any code after return is not executed.

If nothing is returned, the function returns an undefined value.

```
function add(num1, num2) { ←  
  // code  
  return result;  
}  
  
let x = add(a, b); ←  
// code
```

function call

Image: Working of JavaScript Function with return statement

Reference: <https://cdn.programiz.com/cdn/farfuture/b4h4Zo5ZYxj-EyfQyao-J5TqbKEefFgqqusPGLWPFS0/mtime:1591786573/sites/tutorial2program/files/javascript-return-statement.png>

#### Example 4: Sum of Two Numbers

```
// program to add two numbers  
  
// declaring a function  
  
function add(a, b) {  
  
  return a + b;  
}  
  
// take input from the user  
  
let number1 = parseFloat(prompt("Enter first number: "));  
  
let number2 = parseFloat(prompt("Enter second number: "));  
  
// calling function  
  
let result = add(number1, number2);  
  
// display the result
```

```
  console.log("The sum is " + result);
```

## Output

Enter first number: 3.4

Enter second number: 4

The sum is 7.4

In the above program, the sum of the numbers is returned by the function using the return statement. And that value is stored in the result variable.

## Benefits of Using a Function

- Function makes the code reusable. You can declare it once and use it multiple times.
- Function makes the program easier as each small task is divided into a function.
- Function increases readability.

## Function Expressions

In Javascript, functions can also be defined as expressions. For example,

```
// program to find the square of a number
// function is declared inside the variable
let x = function (num) { return num * num };

console.log(x(4));
// can be used as variable value for other variables
let y = x(3);
console.log(y);
```

## Output

16

9

In the above program, variable x is used to store the function. Here the function is treated as an expression. And the function is called using the variable name.

The function above is called an anonymous function.

## Introduction to JavaScript Event

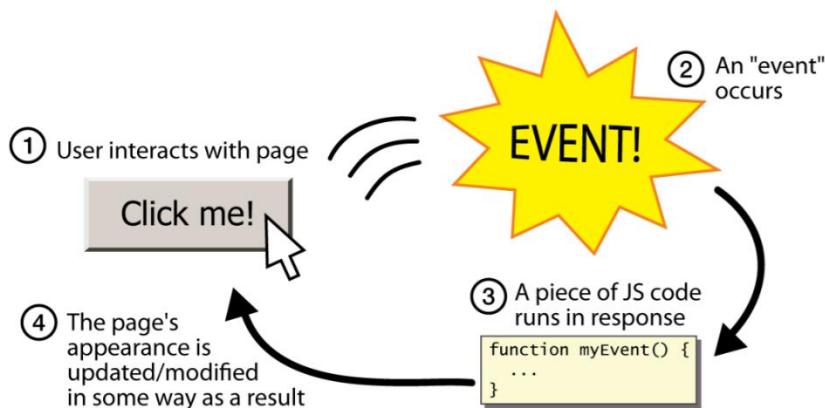
### What is an Event?

JavaScript's interaction with HTML is handled through events that occur when the user or the browser manipulates a page.

When the page loads, it is called an event. Many events can occur while a user is interacting with a Web page. For example, a user might click on a button, change some text, move the mouse pointer over a hyperlink or away from one, and, of course, cause a document to load, pressing any key, closing a window, resizing a window, etc.

Developers can use these events to execute JavaScript coded responses, which cause buttons to close windows, messages to be displayed to users, data to be validated, and virtually any other type of response imaginable.

Events are a part of the Document Object Model (DOM) Level 3 and every HTML element contains a set of events which can trigger JavaScript Code.



Event-driven programming is when parts of the programming are executed in an unpredictable sequence in response to specific events.

Events are objects in JavaScript with case-sensitive names, all of which are lower-case.

### What is Event handling?

Event Handling is the mechanism that controls the event and decides what should happen if an event occurs. This mechanism has the code which is known as event handler that is executed when an event occurs. Java Uses the Delegation Event Model to handle the events. This model defines the standard mechanism to generate and handle the events.

**Source** - The source is an object on which event occurs. Source is responsible for providing information of the occurred event to its handler. Java provides classes for source object.

**Listener** - It is also known as event handler. An event handler is a script that is implicitly executed in response to an event happening. From java implementation point of view the listener is also an object. Listener waits until it receives an event. Once the event is received, the listener processes the event and then returns.

## Functions of Event Handling

- Event Handling identifies where an event should be forwarded.
- It makes the forward event.
- It receives the forwarded event.
- It takes some kind of appropriate action in response, such as writing to a log, sending an error or recovery routine or sending a message.
- The event handler may ultimately forward the event to an event consumer.

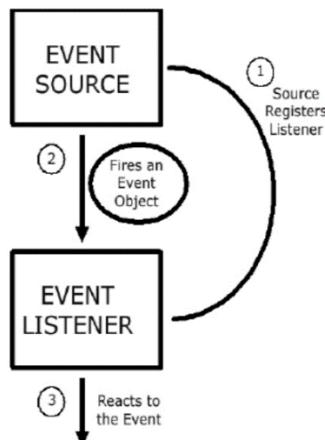


Figure 1- Delegation Event Model

JavaScript contains predetermined event handlers that deal with these events. Table shown below describes some commonly used JavaScript events.

Events	Description
onabort	Occurs when the loading of an image is aborted
onblur	Occurs when input focus is removed from a form element (e.g., when a user clicks the mouse button outside of a particular field)
onclick	Occurs when the user clicks on a link or form element
onchange	Occurs when a user changes the value of a form field

onerror	Occurs when an error takes place while a page or image is loading
onfocus	Occurs when a user gives input or focus to a form element
onload	Occurs when a page is loaded into the browser (i.e., opened)
onmouseOver	Occurs when the user moves the mouse pointer over a link, image or other visible element on a page
onmouseout	Occurs when the mouse pointer leaves a link, image or other visible element on a page
onreset	Occurs when a form's Reset button is clicked
onselect	Occurs when the user selects the text in a form field
onsubmit	Occurs when a form's Submit button is clicked
onunload	Occurs when a page is unloaded from the browser (i.e., closed)

Table 1- JavaScript user event examples

### Example – onload event

```
<!DOCTYPE html>
<!-- load.html
An example to illustrate the load event
A document for load.js -->
<html lang = "en">
<head>
<title> onLoad event handler </title>
<meta charset = "utf-8" />
<script type = "text/javascript" src = "load.js" ></script>
</head>
```

```
<body onload = "loadGreeting();">  
<p />  
</body>  
</html>
```

Load.js

```
// load.js  
// An example to illustrate the load event  
// The onload event handler  
function loadGreeting() {  
    alert("You are visiting the home page of \n" + "Pete's Pickled Peppers\n" + "WELCOME!!!");  
}
```

## Exceptional handling

Exception handling is one of the powerful JavaScript features to handle errors and maintain a regular JavaScript code/program flow.

An exception is an object with an explanation of what went amiss. Also, it discovers where the problem occurred. Errors occur due to mistakes made by developers, wrong input, or unforeseeable things.

A few reasons why exceptions occur are listed below:

- Dividing a number by zero: This results in infinity, thus throwing an exception.
- When a requested file does not exist in the system.
- When the user provides the wrong input.
- When the network drops during communication.

If a software engineer fails to plan for failure, then whatever project or code they are working on may not be successful (when an error does occur). That is where [exception handling](#) comes into play.

When JavaScript encounters an error and raises an exception. The JavaScript translator looks for an exception handling code rather than proceeding to the next statement. In a programming environment, exceptions can be handled, but errors cannot.

### JavaScript error types

Different errors may occur while executing a JavaScript code. There are three types of errors.

1. Syntax Errors: These are errors that cannot be interpreted by the computer. These errors stop the program from working.

In JavaScript, these errors are:

- Spelling errors (wrong spelling such as fiction instead of function).
  - The omission of important characters, such as not using a semicolon to end a statement.
  - Use of the wrong indentation.
2. Runtime Errors: These errors take place during execution. The errors get detected when your program runs. It crashes or raises an exception. Thus, exception handlers handle exception errors.

These errors are often caused by:

- The program not being able to find data because it does not exist.
- The data being an invalid type of data.

- 
3. Logical Errors: These types of errors do not throw an error or an exception at all. This is because they result from the code not doing what the developer intends it to. It's challenging to find logical errors. They can only be found through thorough testing.

## Error objects

When a runtime error occurs, it stops the code and raises an error object.

The error object has two properties:

- Name: It gives the error name.
- Message: It sets or returns the error message in the form of a string.

JavaScript uses six types of error objects. These error objects are the foundation of exception handling.

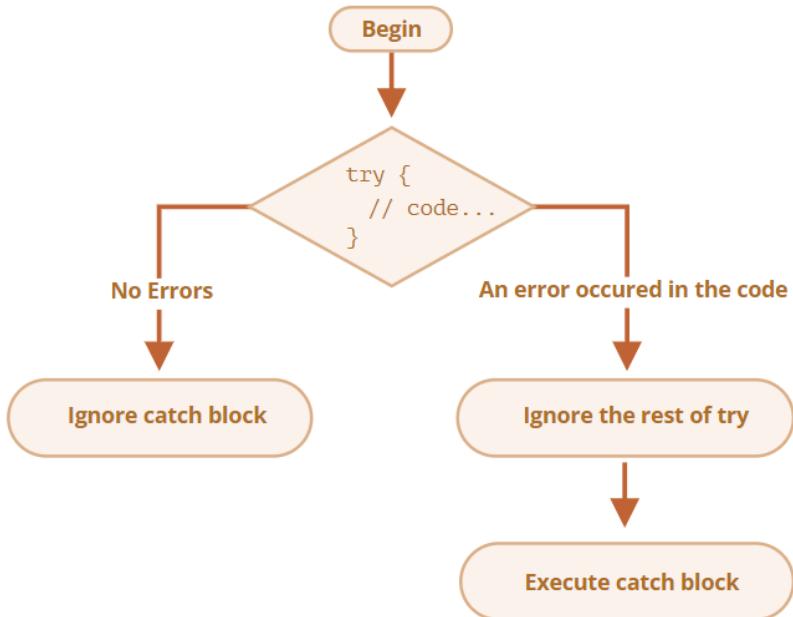
- EvalError: The EvalError function indicates the error that occurred in the eval() function. It's a global function that evaluates the JavaScript string. JavaScript does not throw this exception anymore.
- RangeError: RangeError exceptions occur when a numeric value is outside the specified range.
- ReferenceError: A ReferenceError exception occurs when undeclared variables are used. These exceptions commonly occur due to spelling errors on variables.
- Syntax Error: A Syntax Error exception occurs when JavaScript language rules get broken.
- TypeError: A TypeError exception occurs when a value is different from the one expected.
- URIError: A URIError exception is raised by encodeURI() and decodeURI() methods.

## How to handle exceptions in JavaScript

Now that we now understand what exceptions are. It's time to learn how to handle them to stop our code from crashing. JavaScript handles exceptions in try-catch-finally statements and throw statements.

### Key Terms

- A *try-catch-finally* statement is a code or program that handles exceptions.
- The *try* clause runs the code that generates exceptions.
- The *catch* clause catches exceptions that are thrown.
- A *finally* clause always gets executed.
- The *throw* statement generates exceptions.



Insert the JavaScript code inside the script tag to understand how each exception handling statement works.

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Try-Catch-Finally Statement</title>

</head>

<body>

<script type="text/JavaScript"></script>

<p>Click the button to see the output</p>

<button type="button" onclick="myFunction()">Click Here</button>

</body>

</html>
```

**Throw statements** - The throw statement is to raise your built-in exceptions.

Below is an example of a throw statement:

```
function myFunction() {  
  const x = 50;  
  const y = 0;  
  try {  
    if (y === 0) {  
      throw ("This is division by zero error");  
    } else {  
      const z = x / y;  
    }  
  } catch (error) {  
    alert("Error: " + error);  
  }  
}
```

**Try catch statements** - The try clause has the main code that may generate exceptions. If an exception is raised, the catch clause gets executed.

Here is an example of a try-catch statement:

```
function myFunction() {  
  const j = 70;  
  try {  
    alert ("The value of j is: " + j);  
  } catch (error) {  
    alert("Error: " + error.message);  
  }  
}
```

In the example above, we have made a typo error while calling the in-built alert() function. We have misspelled alert to produce an error deliberately. The catch clause catches the error and executes the code.

**Try catch finally statements** - The finally statement is the last block to be executed. It executes after try and catch clauses.

Here is an example of try-catch-finally statements:

```
function myFunction() {  
    const j = 70;  
    try {  
        alert("The value of j is : " + j);  
    } catch (error) {  
        alert("Error: " + error.message);  
    } finally {  
        alert("Finally: Finally gets executed")  
    }  
}
```

## The Browser Object Model (BOM)

A Web page is made dynamic by applying JavaScript processing to the HTML elements on that page. Up to this point you probably have considered HTML tags simply as markup codes providing structure to page content and supplying mechanisms through which styling is applied to that content. Importantly, though, HTML tags are also **software objects**. That is, all HTML tags have properties and methods that can be programmed. As is the case with all software objects, **properties** refer to structural, visual, or content characteristics of the element; methods refer to actions the object can perform. HTML tags, then, are programmable through JavaScript processing routines, or scripts, that set their properties and activate their methods in order to make Web pages **dynamic**.

The **browser object model (BOM)** is a hierarchy of browser objects that are used to manipulate methods and properties associated with the Web browser itself. Objects that make up the BOM include the window object, navigator object, screen object, history, location object, and the document object. The Document Object consists of objects that are used to manipulate methods and properties of the document or Web page loaded in the browser window. The document object represents the Web page currently loaded in the browser window. Each HTML element or tag that makes up the document is also considered an object. It is not necessary to explicitly create any of the objects that make up the browser object model. The objects are automatically created when a Web browser opens a Web page.

### The BOM Hierarchy

The top-level object in the BOM is the window object. The **window object** represents the browser window. All other browser objects are contained within the window object. The window object includes a number of properties and methods that can be used to control the Web browser. The window object along with its properties and methods are discussed in more detail in a later section.

The **document object** represents the Web page displayed in the browser. All elements on a Web page including HTML tags are contained within the document object. Since the document object is often considered the most important part of the BOM, it is represented by its own object model called the **Document Object Model or DOM**. The DOM will be discussed in more detail in later tutorials.

Other objects of the browser object model include the **navigator object**, the **screen object**, that contains information about the visitor's screen, the **history object**, that is part of the window object and contains the URLs that have been visited by the user, and the **location object** that contains information about the current URL. Within the window object are **document objects** representing elements within the Web pages. The general hierarchy of the BOM is shown in the illustration below.

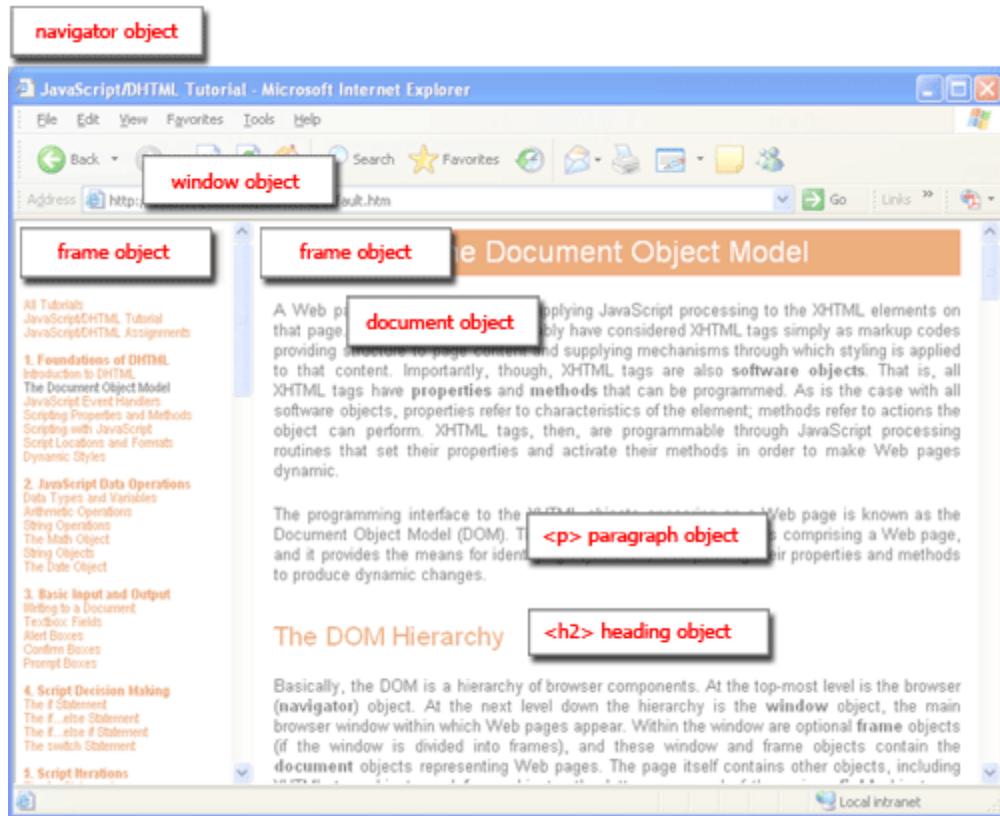


Figure 2 Components of the Browser Object Model (BOM)

## Identifying BOM Objects

In order to program BOM objects, they must be identified to the scripts that manipulate their properties and methods. The following table summarizes several of the references used within scripts to identify common BOM objects. These and other reference notations are explained and illustrated throughout these tutorials.

Reference	Object
window	The main browser window
window.navigator	Information about the browser itself
window.screen	The user's screen
window.history	URLs visited by a user

window.location	The current URL
window.document (document)	The document appearing in the main browser window
document.getElementById("id")	An HTML element appearing in a document and identified by its assigned <i>id</i> value.

As you can see, script references to BOM objects use standard dotted notation to trace through the BOM hierarchy to identify particular objects. In some cases, there are short-cut notations that do not require the complete hierarchical path to an object. For example, **window.document** can be shorten to **document**.

A good portion of client-side Web development is in working with the properties and methods associated with the browser itself, with its windows, and with the documents that occupy them. The largest part of client-side scripting, though, is in working with the properties and methods of HTML elements appearing on a Web page. In most cases, this involves detecting the **style settings** of HTML tags and changing these settings to change the appearance of, or to change the content enclosed by, these tags. In other cases, it involves calling up built-in methods to affect the behaviors of HTML tags. Being able to make proper script references to HTML elements is an important aspect of client-side scripting.

The most important object in the Browser Object Model is the **window** object. It helps in accessing information about the browser and its components. To access these features, it has various methods and properties.

Method	Description
window.alert()	Creates dialog box with message and an OK button
window.blur()	Removes focus from window
window.close()	Closes a browser window
window.confirm()	Creates dialog box with message, an OK button and a cancel button
window.getComputedStyle()	Get CSS styles applied to an element

window.moveTo(x,y)	Move a window's left and top edge to supplied coordinates
window.open()	Opens new browser window with URL specified as parameter
window.print()	Tells browser that user wants to print contents of current page
window.prompt()	Creates dialog box for retrieving user input
window.scrollBy()	Scrolls the document by the specified number of pixels
window.scrollTo()	Scrolls the document to the specified coordinates
window.setInterval()	Do something repeatedly at specified intervals
window.setTimeout()	Do something after a specified amount of time
window.stop()	Stop window from loading

The Window Object contains the following properties.

<b>Property</b>	<b>Description</b>
window.closed	Whether the window has been closed
window.length	Number of <iframe> elements in window
window.name	Gets or sets the name of the window
window.innerHeight	Height of window
window.innerWidth	Width of window
window.screenX	X-coordinate of pointer, relative to top left corner of screen

window.screenY	Y-coordinate of pointer, relative to top left corner of screen
window.location	Current URL of window object (or local file path)
window.history	Reference to history object for browser window or tab.
window.screen	Reference to screen object
window.pageXOffset	Distance document has been scrolled horizontally
window.pageYOffset	Distance document has been scrolled vertically

### **Example of alert() in JavaScript**

It displays alert dialog box. It has message and ok button.

```
<script type="text/javascript">
function msg(){
    alert("Hello Alert Box");
}
</script>
<input type="button" value="click" onclick="msg()"/>
```

### **Example of confirm() in JavaScript**

It displays the confirm dialog box. It has message with ok and cancel buttons.

```
<script type="text/javascript">
function msg(){
    var v= confirm("Are u sure?");
    if(v==true){
        alert("ok");
    }
    else{
```

```
    alert("cancel");
}
} </script>
<input type="button" value="delete record" onclick="msg()"/>
```

### Example of prompt() in JavaScript

It displays prompt dialog box for input. It has message and textfield.

```
<script type="text/javascript">
function msg(){
var v= prompt("Who are you?");
alert("I am "+v);
}
</script>
<input type="button" value="click" onclick="msg()"/>
```

### Example of open() in JavaScript

It displays the content in a new window.

```
<script type="text/javascript">
function msg(){
open("http://www.edunetfoundation.com");
}
</script>
<input type="button" value="javascript1" onclick="msg()"/>
```

### Example of setTimeout() in javascript

It performs its task after the given milliseconds.

```
<script type="text/javascript">
function msg(){
```

```
setTimeout(  
  function(){  
    alert("Display JavaScript after 2 seconds")  
  },2000);  
}  
</script>  

```

## JavaScript Navigator Object

The **JavaScript navigator object** is used for browser detection. It can be used to get browser information such as appName, appCodeName, userAgent etc.

The navigator object is the window property, so it can be accessed by:

window.navigator Or, navigator

### Property of JavaScript navigator object

There are many properties of navigator object that returns information of the browser.

No.	Property	Description
1	appName	returns the name
2	appVersion	returns the version
3	appCodeName	returns the code name
4	cookieEnabled	returns true if cookie is enabled otherwise false
5	userAgent	returns the user agent
6	language	returns the language. It is supported in Netscape and Firefox only.
7	userLanguage	returns the user language. It is supported in IE only.

8	plugins	returns the plugins. It is supported in Netscape and Firefox only.
9	systemLanguage	returns the system language. It is supported in IE only.
10	mimeTypes[]	returns the array of mime type. It is supported in Netscape and Firefox only.
11	platform	returns the platform e.g. Win32.
12	online	returns true if browser is online otherwise false.

### Methods of JavaScript navigator object

The methods of navigator object are given below.

No.	Method	Description
1	javaEnabled()	checks if java is enabled.
2	taintEnabled()	checks if taint is enabled. It is deprecated since JavaScript 1.2.

## Example of navigator object

Let's see the different usage of history object.

```
<script>

document.writeln("<br/>navigator.appCodeName: "+navigator.appCodeName);

document.writeln("<br/>navigator.appName: "+navigator.appName);

document.writeln("<br/>navigator.appVersion: "+navigator.appVersion);

document.writeln("<br/>navigator.cookieEnabled: "+navigator.cookieEnabled);

document.writeln("<br/>navigator.language: "+navigator.language);

document.writeln("<br/>navigator.userAgent: "+navigator.userAgent);

document.writeln("<br/>navigator.platform: "+navigator.platform);

document.writeln("<br/>navigator.onLine: "+navigator.onLine);

</script>
```

## JavaScript screen object

The Screen object provides the attributes of the screen on which the current window is being rendered.

To access the Screen object, you use the screen property of the window object: window.screen

The Screen object is typically used by the web analytic software like Google Analytics to collect information of the client device on which the web browsers are running.

### Property of JavaScript Screen Object

There are many properties of screen object that returns information of the browser.

No.	Property	Description
1	width	returns the width of the screen
2	height	returns the height of the screen
3	availWidth	returns the available width

4	availHeight	returns the available height
5	colorDepth	returns the color depth
6	pixelDepth	returns the pixel depth.

### Example of JavaScript Screen Object

```
<script>

document.writeln("<br/>screen.width: "+screen.width);

document.writeln("<br/>screen.height: "+screen.height);

document.writeln("<br/>screen.availWidth: "+screen.availWidth);

document.writeln("<br/>screen.availHeight: "+screen.availHeight);

document.writeln("<br/>screen.colorDepth: "+screen.colorDepth);

document.writeln("<br/>screen.pixelDepth: "+screen.pixelDepth);

</script>
```

### JavaScript setTimeout() method

The **setTimeout()** method in JavaScript is used to execute a function after waiting for the specified time interval. This method returns a numeric value that represents the ID value of the timer.

The **setTimeout()** method executes the function only once. This method can be written with or without the *window* prefix.

We can use the **clearTimeout()** method to stop the timeout or to prevent the execution of the function specified in the **setTimeout()** method. The value returned by the **setTimeout()** method can be used as the argument of the **clearTimeout()** method to cancel the timer.

The commonly used syntax of the **setTimeout()** method is given below.

```
window.setTimeout(function, milliseconds);
```

#### Parameter values

This method takes two parameter values function and milliseconds that are defined as follows.

**function:** It is the function containing the block of code that will be executed.

**milliseconds:** This parameter represents the time-interval after which the execution of the function takes place. The interval is in milliseconds. Its default value is 0. It defines how often the code will be executed. If it is not specified, the value 0 is used.

```
<html>
  <head>
    <title> setTimeout() method </title>
  </head>
  <body>
    <h1> Hello World :) :) </h1>
    <h3> This is an example of using the setTimeout() method </h3>
    <p> Click the following button before 2 seconds to see the effect. </p>
    <button onclick = "stop()"> Stop </button>
    <script>
      var a = setTimeout(fun1, 2000);

      function fun1() {
        var win1 = window.open();
        win1.document.write(" <h2> Learn Javascript </h2>");
        setTimeout(function(){win1.close()}, 2000);
      }

      function stop() {
        clearTimeout(a);
      }
    </script>
  </body>
</html>
```

## JavaScript setInterval() method

The **setInterval()** method in JavaScript is used to repeat a specified function at every given time-interval. It evaluates an expression or calls a function at given intervals. This method continues the calling of function until the window is closed or the **clearInterval()** method is called. This method returns a numeric value or a non-zero number that identifies the created timer.

Unlike the **setTimeout()** method, the **setInterval()** method invokes the function multiple times. This method can be written with or without the *window* prefix.

The commonly used syntax of **setInterval()** method is given below:

```
window.setInterval(function, milliseconds);
```

### Parameter values

This method takes two parameter values *function* and *milliseconds* that are defined as follows.

**function:** It is the function containing the block of code that will be executed.

**milliseconds:** This parameter represents the length of the time interval between each execution. The interval is in milliseconds. It defines how often the code will be executed. If its value is less than 10, the value 10 is used.

```
<html>
  <head>
    <title> setInterval() method </title>
  </head>
  <body>
    <h1> Hello World :) :) </h1>
    <h3> This is an example of using the setInterval() method </h3>
    <p> Here, the background color changes on every 200 milliseconds. </p>
    <button onclick = "stop()"> Stop </button>

    <script>
      var var1 = setInterval(color, 200);
    </script>
  </body>
</html>
```

```
function color() {  
    var var2 = document.body;  
  
    var2.style.backgroundColor = var2.style.backgroundColor == "lightblue" ? "lightgreen": "lightblue";  
}  
  
function stop() {  
    clearInterval(var1);  
}  
  
</script>  
  
</body>  
  
</html>
```

## JavaScript Date

The JavaScript date object can be used to get year, month and day. You can display a timer on the webpage by the help of JavaScript date object.

Syntax:

- `new Date();`
- `new Date(value);`
- `new Date(dateAsString);`
- `new Date(year, month[, day[, hour[, minute[, second[, millisecond]]]]]);`

Let's see the list of JavaScript date methods with their description.

Methods	Description
<code>getDate()</code>	It returns the integer value between 1 and 31 that represents the day for the specified date on the basis of local time.
<code>getDay()</code>	It returns the integer value between 0 and 6 that represents the day of the week on the basis of local time.

getFullYears()	It returns the integer value that represents the year on the basis of local time.
getHours()	It returns the integer value between 0 and 23 that represents the hours on the basis of local time.
getMilliseconds()	It returns the integer value between 0 and 999 that represents the milliseconds on the basis of local time.
getMinutes()	It returns the integer value between 0 and 59 that represents the minutes on the basis of local time.
getMonth()	It returns the integer value between 0 and 11 that represents the month on the basis of local time.
getSeconds()	It returns the integer value between 0 and 60 that represents the seconds on the basis of local time.
setDate()	It sets the day value for the specified date on the basis of local time.
setDay()	It sets the particular day of the week on the basis of local time.
setFullYears()	It sets the year value for the specified date on the basis of local time.
setHours()	It sets the hour value for the specified date on the basis of local time.
setMilliseconds()	It sets the millisecond value for the specified date on the basis of local time.
setMinutes()	It sets the minute value for the specified date on the basis of local time.
setMonth()	It sets the month value for the specified date on the basis of local time.

setSeconds()	It sets the second value for the specified date on the basis of local time.
toJSON()	It returns a string representing the Date object. It also serializes the Date object during JSON serialization.
toString()	It returns the date in the form of string.
toTimeString()	It returns the time portion of a Date object.
valueOf()	It returns the primitive value of a Date object.

### Examples:

```
Current Date and Time: <span id="txt"></span>

<script>

var today=new Date();

document.getElementById('txt').innerHTML=today;

</script>
```

```
Current Time: <span id="txt"></span>
```

```
<script>

window.onload=function(){getTime();}

function getTime(){

var today=new Date();

var h=today.getHours();

var m=today.getMinutes();

var s=today.getSeconds();
```

```
// add a zero in front of numbers<10
m=checkTime(m);
s=checkTime(s);
document.getElementById('txt').innerHTML=h+":"+m+":"+s;
setTimeout(function(){getTime()},1000);
}

//setInterval("getTime()",1000);//another way

function checkTime(i){
if (i<10){
  i="0" + i;
}
return i;
}

</script>
```

### Output:

Current Time: 15:18:10

### Example - Convert to JSON

```
var date1 = new Date();
date1.toJSON();
Output - "2022-02-23T15:49:08.596Z"
```

### What is Ajax?

Ajax stands for **A**synchronous **J**avascript **A**nd **X**ml. Ajax is just a means of loading data from the server and selectively updating parts of a web page without reloading the whole page.

Basically, what Ajax does is make use of the browser's built-in XMLHttpRequest (XHR) object to send and receive information to and from a web server asynchronously, in the background, without blocking the page or interfering with the user's experience.

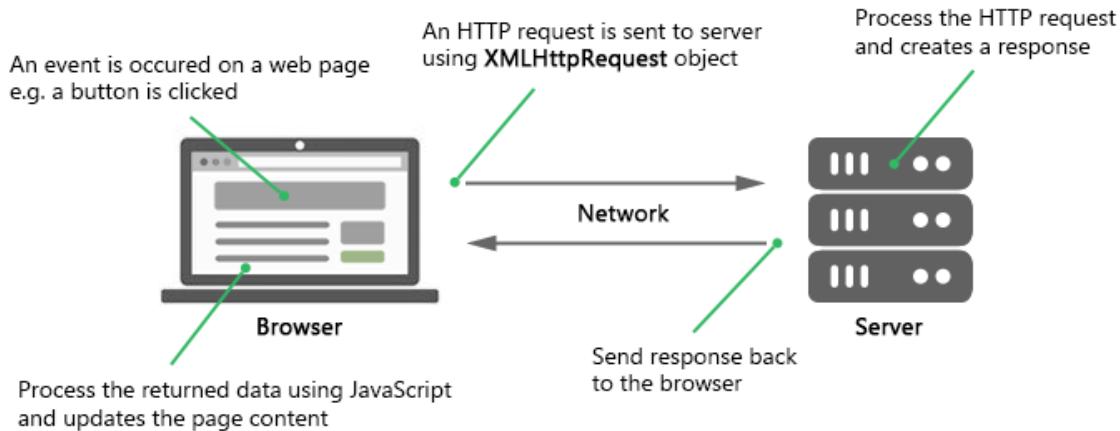
Ajax has become so popular that you hardly find an application that doesn't use Ajax to some extent. The example of some large-scale Ajax-driven online applications are: Gmail, Google Maps, Google Docs, YouTube, Facebook, Flickr, and so many other applications.

## Understanding How Ajax Works

To perform Ajax communication JavaScript uses a special object built into the browser—an XMLHttpRequest (XHR) object—to make HTTP requests to the server and receive data in response.

All modern browsers (Chrome, Firefox, IE7+, Safari, Opera) support the XMLHttpRequest object.

The following illustrations demonstrate how Ajax communication works:



Since Ajax requests are usually asynchronous, execution of the script continues as soon as the Ajax request is sent, i.e., the browser will not halt the script execution until the server response comes back.

In the following section we'll discuss each step involved in this process one by one:

### Sending Request and Retrieving the Response

Before you perform Ajax communication between client and server, the first thing you must do is to instantiate an XMLHttpRequest object, as shown below:

```
var request = new XMLHttpRequest();
```

Now, the next step in sending the request to the server is to instantiating the newly-created request object using the `open()` method of the `XMLHttpRequest` object.

The `open()` method typically accepts two parameters—the HTTP request method to use, such as "GET", "POST", etc., and the URL to send the request to, like this:

```
request.open("GET", "info.txt"); -Or- request.open("POST", "add-user.php");
```

And finally send the request to the server using the `send()` method of the `XMLHttpRequest` object.

```
request.send(); -Or- request.send(body);
```

The GET method is generally used to send small amount of data to the server. Whereas, the POST method is used to send large amount of data, such as form data.

In GET method, the data is sent as URL parameters. But, in POST method, the data is sent to the server as a part of the HTTP request body. Data sent through POST method will not visible in the URL.

In the following section we'll take a closer look at how Ajax requests actually works.

## Performing an Ajax GET Request

The GET request is typically used to get or retrieve some kind of information from the server that doesn't require any manipulation or change in database, for example, fetching search results based on a term, fetching user details based on their id or name, and so on.

The following example will show you how to make an Ajax GET request in JavaScript.

### Example

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<title>JavaScript Ajax GET Demo</title>

<script>

function displayFullName() {

    // Creating the XMLHttpRequest object
```

```
var request = new XMLHttpRequest();

// Instantiating the request object
request.open("GET", "greet.php?fname=John&lname=Clark");

// Defining event listener for readystatechange event
request.onreadystatechange = function() {
    // Check if the request is completed and was successful
    if(this.readyState === 4 && this.status === 200) {
        // Inserting the response from server into an HTML element
        document.getElementById("result").innerHTML = this.responseText;
    }
};

// Sending the request to the server
request.send();
}

</script>
</head>
<body>
<div id="result">
    <p>Content of the result DIV box will be replaced by the server response</p>
</div>
<button type="button" onclick="displayFullName()">Display Full Name</button>
</body>
```

```
</html>
```

When the request is asynchronous, the `send()` method returns immediately after sending the request. Therefore, you must check where the response currently stands in its lifecycle before processing it using the `readyState` property of the `XMLHttpRequest` object.

The `readyState` is an integer that specifies the status of an HTTP request. Also, the function assigned to the `onreadystatechange` event handler called every time the `readyState` property changes. The possible values of the `readyState` property are summarized below.

Value	State	Description
0	UNSENT	An <code>XMLHttpRequest</code> object has been created, but the <code>open()</code> method hasn't been called yet (i.e. request not initialized).
1	OPENED	The <code>open()</code> method has been called (i.e. server connection established).
2	HEADERS_RECEIVED	The <code>send()</code> method has been called (i.e. server has received the request).
3	LOADING	The server is processing the request.
4	DONE	The request has been processed and the response is ready.

The `status` property returns the numerical HTTP status code of the `XMLHttpRequest`'s response. Some of the common HTTP status codes are listed below:

- 200 — OK. The server successfully processed the request.
- 404 — Not Found. The server can't find the requested page.
- 503 — Service Unavailable. The server is temporarily unavailable.

Please check out the [HTTP status codes](#) reference for a complete list of response codes.

Here's the code from our "greet.php" file that simply creates the full name of a person by joining their first name and last name and outputs a greeting message.

### Example

```
<?php

if(isset($_GET["fname"]) && isset($_GET["lname"])) {

    $fname = htmlspecialchars($_GET["fname"]);

    $lname = htmlspecialchars($_GET["lname"]);

    // Creating full name by joining first and last name

    $fullname = $fname . " " . $lname;

    // Displaying a welcome message

    echo "Hello, $fullname! Welcome to our website./";

} else {

    echo "Hi there! Welcome to our website./";

}

?>
```

## Performing an Ajax POST Request

The POST method is mainly used to submit a form data to the web server.

The following example will show you how to submit form data to the server using Ajax.

### Example

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<title>JavaScript Ajax POST Demo</title>

<script>

function postComment() {
```

```
// Creating the XMLHttpRequest object
var request = new XMLHttpRequest();

// Instantiating the request object
request.open("POST", "confirmation.php");

// Defining event listener for readystatechange event
request.onreadystatechange = function() {
    // Check if the request is compete and was successful
    if(this.readyState === 4 && this.status === 200) {
        // Inserting the response from server into an HTML element
        document.getElementById("result").innerHTML = this.responseText;
    }
};

// Retrieving the form data
var myForm = document.getElementById("myForm");
var formData = new FormData(myForm);

// Sending the request to the server
request.send(formData);
}

</script>
</head>
<body>
```

```
<form id="myForm">

  <label>Name:</label>

  <div><input type="text" name="name"></div>

  <br>

  <label>Comment:</label>

  <div><textarea name="comment"></textarea></div>

  <p><button type="button" onclick="postComment()">Post Comment</button></p>

</form>

<div id="result">

  <p>Content of the result DIV box will be replaced by the server response</p>

</div>

</body>

</html>
```

If you are not using the `FormData` object to send form data, for example, if you're sending the form data to the server in the *query string* format, i.e. `request.send(key1=value1&key2=value2)` then you need to explicitly set the request header using `setRequestHeader()` method, like this:

```
request.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
```

The `setRequestHeader()` method, must be called after calling `open()`, but before calling `send()`.

Some common request headers are: `application/x-www-form-urlencoded`, `multipart/form-data`, `application/json`, `application/xml`, `text/plain`, `text/html`, and so on.

Here's the code of our "confirmation.php" file that simply outputs the values submitted by the user.

### Example

```
<?php

if($_SERVER["REQUEST_METHOD"] == "POST") {

  $name = htmlspecialchars(trim($_POST["name"]));
```

```
$comment = htmlspecialchars(trim($_POST["comment"]));  
  
// Check if form fields values are empty  
  
if(!empty($name) && !empty($comment)) {  
  
    echo "<p>Hi, <b>$name</b>. Your comment has been received successfully.<p>";  
  
    echo "<p>Here's the comment that you've entered: <b>$comment</b></p>";  
  
} else {  
  
    echo "<p>Please fill all the fields in the form!</p>";  
  
}  
  
} else {  
  
    echo "<p>Something went wrong. Please try again.</p>";  
  
}  
  
?>
```

For security reasons, browsers do not allow you to make cross-domain Ajax requests. This means you can only make Ajax requests to URLs from the same domain as the original page, for example, if your application is running on the domain "mysite.com", you cannot make Ajax request to "othersite.com" or any other domain. This is commonly known as *same origin policy*.

However, you can load images, style sheets, JS files, and other resources from any domain.

## JavaScript JSON

JSON stands for JavaScript Object Notation. JSON is extremely lightweight data-interchange format for data exchange between server and client which is quick and easy to parse and generate.

Like XML, JSON is also a text-based format that's easy to write and easy to understand for both humans and computers, but unlike XML, JSON data structures occupy less bandwidth than their XML versions. JSON is based on two basic structures:

- **Object:** This is defined as an unordered collection of key/value pairs (i.e. key:value). Each object begins with a left curly bracket { and ends with a right curly bracket }. Multiple key/value pairs are separated by a comma ,.
- **Array:** This is defined as an ordered list of values. An array begins with a left bracket [ and ends with a right bracket ]. Values are separated by a comma ,.

In JSON, property names or keys are always strings, while the value can be a string, number, true or false, null or even an object or an array. Strings must be enclosed in double quotes " and can contain escape characters such as \n, \t and \. A JSON object may look like this:

### Example

```
{  
  "book": {  
    "name": "Harry Potter and the Goblet of Fire",  
    "author": "J. K. Rowling",  
    "year": 2000,  
    "genre": "Fantasy Fiction",  
    "bestseller": true  
  }  
}
```

## JavaScript JSON Methods

Let's see the list of JavaScript **JSON** method with their description.

Methods	Description
<a href="#">JSON.parse()</a>	This method takes a JSON string and transforms it into a JavaScript object.
<a href="#">JSON.stringify()</a>	This method converts a JavaScript value (JSON object) to a JSON string representation.

Let's see an example to convert string in JSON format using parse() and stringify() method.

```

<script>

//JavaScript to illustrate JSON.parse() method.

var j = '{"Name":"Krishna","Email": "XYZ", "CN": "12345"}';

var data = JSON.parse(j);

document.write("Convert string in JSON format using parse() method<br>");

document.write(data.Email); //expected output: XYZ

//JavaScript to illustrate JSON.stringify() method.

var j = {Name:"Krishna",

Email: "XYZ", CN : 12345};

var data = JSON.stringify(j);

document.write("<br>Convert string in JSON format using stringify() method<br>");

document.write(data); //expected output: {"Name":"Krishna","Email":"XYZ","CN":12345}

</script>

```

### Output:

Convert string in JSON format using parse() method

XYZ

Convert string in JSON format using stringify() method

{"Name":"John","Email": "XYZ", "CN": 12345}

## **Learning Outcome**

### **Bootstrap**

## Overview of Bootstrap

Bootstrap is a free and open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites. It solves many problems which we had once, one of which is the cross-browser compatibility issue. Nowadays, the websites are perfect for all the browsers (IE, Firefox, and Chrome) and for all sizes of screens (Desktop, Tablets, Phablets, and Phones). All thanks to Bootstrap developers -Mark Otto and Jacob Thornton of Twitter, though it was later declared to be an open-source project.

### Overview of Bootstrap

- Framework for Front-End development
- Open source project by twitter
- Responsive
- Mobile First approach

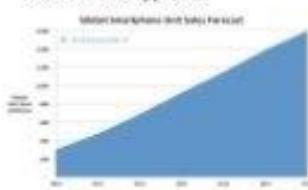


Image : Overview of Bootstrap

Reference: <https://image.slidesharecdn.com/bootstrap-150705164322-lva1-app6892/85/bootstrap-3-320.jpg?cb=1436114649>

## Advantages of Bootstrap

The first thing is that now days people use mobiles, laptops, tabs, personal computers. If we have to reach our website to all people, we should spread our business more and more. The website should be such that everyone can read and view the content of your website well. So now we will not make different designs for different devices because may be a new Mobile Screen Ratio may come tomorrow, or if a new laptop screen size is launched, then our website will not work well on it. So to avoid all this mess Bootstrap is there. In which you have to create a website just once, then whatever the screen viewer opens the website on any screen, the UI (User Interface) of the website will be shown well. And people create and sell templates from bootstrap which people use in WordPress, Joomla, Drupal etc.

## The Main Benefits of Bootstrap



Image : Benefits of Bootstrap  
Reference: [https://pbs.twimg.com/media/EonHh\\_wUwAEcsP2?format=png&name=large](https://pbs.twimg.com/media/EonHh_wUwAEcsP2?format=png&name=large)

### Saving time

This is the biggest advantage of Bootstrap that using it increases the development speed and gets the work done in a very short time. On the contrary, if you want to create a responsive design without bootstrap by yourself, then you have a lot of time in it.

### Easy to use

If you have basic knowledge of HTML and CSS, then you can easily use Bootstrap.

### Responsive Design

Through bootstrap you can easily create responsive design. If your website is responsive then it adjusts itself according to the screen size in any platform or device like desktop, laptop, mobile etc.

### Cross Browser Compatible

Bootstrap is designed in such a way that our web-page will look the same in almost all modern browsers like Firefox, Chrome, Internet Explorer, Opera etc.

### Open Source

The best thing about it is that you can use it for free.

### Customization

Bootstrap is easy to customize, customize means that if you do not want all the features of bootstrap, you can only use a few features. On Bootstrap's website give you a lot of options (Nav bar, Table, Form, Button, Model, Dropdown, Badges, etc.) Let you tick what you want, remove the tick from what you don't want and download and use it. Do it All the CSF classes are already created in Bootstrap; we just have to use those classes in our website and very easily and in a very

short time. We create a beautiful website in this way, Bootstrap reduces the time taken in our website development.

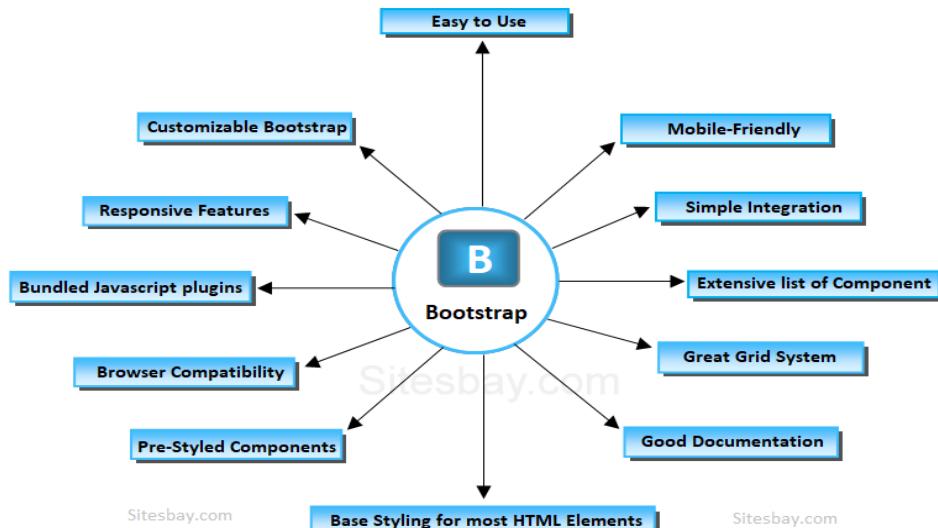


Image : Advantages of Bootstrap

Reference: <https://www.sitesbay.com/bootstrap/images/features-of-bootstrap.png>

## Bootstrap Container

In Bootstrap, container is used to set the content's margins dealing with the responsive behaviors of your layout. It contains the row elements and the row elements are the container of columns (known as grid system).

The container class is used to create boxed content.

Containers are used to pad the content inside of them, and there are two container classes available:

- The .container class provides a responsive fixed width container
- The .container-fluid class provides a full width container, spanning the entire width of the viewport

### Bootstrap simple container

Normal Size

container demo
container demo
container demo

### Bootstrap fluid container

Normal Size

container demo
container demo
container demo

Image : Bootstrap containers  
Reference: <https://www.jquery-az.com/wp-content/uploads/2016/11/12.1-Bootstrap-container.png>

## Fixed Container

Use the .container class to create a responsive, fixed-width container.

Note that its width (max-width) will change on different screen sizes:

	Extra small <576px	Small≥576px	Medium≥768px	Large≥992px	Extra large≥1200px
Max-width	100%	540px	720px	960px	1140px

## Fluid Container

Use the .container-fluid class to create a full width container, that will always span the entire width of the screen (width is always 100%)

### How to use Bootstrap 5 on a webpage:

There are two ways to include Bootstrap on the website.

- Include Bootstrap from the CDN link.
- Download Bootstrap from getbootstrap.com and use it.

### Bootstrap 5 CDN

If you don't want to download and host Bootstrap 5 yourself, you can include it from a CDN (Content Delivery Network).

jsDelivr provides CDN support for Bootstrap's CSS and JavaScript:

MaxCDN:

```
<!-- Latest compiled and minified CSS -->  
  
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"  
rel="stylesheet">  
  
<!-- Latest compiled JavaScript -->
```

## Downloading Bootstrap 5

If you want to download and host Bootstrap 5 yourself, go to <https://getbootstrap.com/>, and follow the instructions there.

## Bootstrap Components

Bootstrap provides a variety of customizable and reusable components which makes the development faster and easier. They are heavily based on the base modifier nomenclature i.e. the base class has many groups of shared properties together while the modifier class has a group of individual styles. For example, .btn is a base class and .btn-primary or .btn-success is a modifier class. The bootstrap components range from alerts, buttons, badges, cards to various other components.

<h2>Components of Bootstrap</h2>		
1.Jumbotron	9.List group	17.Carousel
2.Alerts	10.Card	18.Toast
3.Buttons	11.Dropdown	19.Tooltip
4.Button group	12.Nav	20.Popovers
5.Badge	13.Navbar	21.Collapse
6.Progress Bar	14.Forms	22.Modal
7.Spinner	15.Input group	23.Pagination
8. Scrollspy	16.Breadcrumb	24.Media object

Image : Bootstrap components  
Reference: <https://media.geeksforgeeks.org/wp-content/uploads/20210819203105/202108199-660x372.jpg>

---

## List of components:

### Jumbotron

It simply put extra attention to particular content or information by making it larger and more eye-catching.

### Alerts

It is a popup with a predefined message that appears after a particular action.

### Buttons

It is customized buttons that are used to perform an action in the form, dialogue box, etc. They are in multiple states, sizes and have predefined styles.

### Button group

It is a group of buttons aligned in a single line and they can be arranged both vertically as well as horizontally.

### Badge

It is a labeling component that is used to add additional information.

### Progress Bar

It is used to show the progress of a particular operation with a custom progress bar. They have text labels, stacked bars, and animated backgrounds.

### Spinner

The spinner displays the loading state of websites or projects. They are built with HTML, CSS and don't require any JavaScript.

### Scrollspy

It keeps updating the navigation bar to the currently active link based on the scroll position in the viewport.

## List group

It is used to display an unordered series of content in a proper way.

## Card

It provides a customizable, extensible, and flexible content container.

## Dropdown

It is used to drop the menu in the format of a list of links, they are contextual and toggleable overlays.

## Navs

It is used to create a basic and simple navigation menu with a .nav base class.

## Navbar

The navigation bar is the headers at the top of a website or webpage.

## Forms

Forms are used to take multiple inputs at once from the user. Bootstrap has two layouts available stacked and inline.

## Input groups

They have extended form controls by adding a button, button group or text on either side of inputs.

## Toast

It displays a message for a small amount of time, a few seconds. They are alert messages designed to imitate push notifications popular in desktop and mobile systems.

## Tooltip

It provides small information about the element/link when the mouse hovers over the element.

## Popovers

It displays extra information about the element/link when clicked on it.

## Collapse

It is a JavaScript plugin that is used to show or hide the content.

## Modal

It is a small popup window positioned over the actual window.

## Pagination

It is used to easily navigate between different pages, a large block of connected links is used for making them accessible.

## Media Object

The Media object is used for repetitive and complex components like tweets or blogs. The images or videos are placed/aligned to the left or the right of the content.

## Advance Components of Bootstrap

Bootstrap is already pre-packaged with a huge selection of useful tools, extensions and components, and is more than enough to kick-start most web design or web application projects. But there will be times when the bundled basic components are not quite enough for what you need. That is where this post comes in. We have assembled 45 extensions, plugins, addons and components that will allow you to extend Bootstrap even further.

## Accessibility

This plugin adds accessibility mark-up to the default components of Bootstrap. Components include: Alert, Tooltip, Popover, Modal Dialog, Dropdown Menu, Tab Panel, Collapse and Carousel.

## Breadcrumbs

Bootstrap-Breadcrumb: A Bootstrap JavaScript plugin that allows you to programmatically manipulate breadcrumb navigation.

## Calendar

Bootstrap-Calendar: A Full view calendar based on Bootstrap.

March 2013
[<< Prev](#)
[Today](#)
[Next >>](#)
[Year](#)
[Month](#)
[Week](#)
[Day](#)

To see example with events navigate to [march 2013](#)

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
24	25	26	27	28	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
<span style="color: yellow;">●</span>	<span style="color: yellow;">●</span>	<span style="color: yellow;">●</span> <span style="color: blue;">●</span>	<span style="color: yellow;">●</span> <span style="color: blue;">●</span> <span style="color: green;">●</span> <span style="color: green;">●</span>	<span style="color: blue;">●</span> <span style="color: green;">●</span> <span style="color: green;">●</span> <span style="color: green;">●</span>	<span style="color: purple;">●</span>	<span style="color: purple;">●</span>
17	18	19	20	21	22	23
<span style="color: purple;">●</span>	<span style="color: purple;">●</span>	<span style="color: grey;">●</span>				
24	25	26	27	28	29	30

First day of week language-dependant ▼

Select Language (default: en-US) ▼

Open events in modal window

**Events**

This list is populated with events dynamically

● This is warning class event with very long title to check how it fits to event in day view

● This is information class

● Event that ends on timeline

● This is special event

● This is success event

● Short day event

● This is simple event

● Event 3

● This is inverse event

Image : Advance components of Bootstrap  
 Reference: <https://www.drupal.org/files/project-images/Twitter%20Bootstrap%20jQuery%20Calendar.png>

## Carousel

Bootstrap Modal Carousel – A collection of plugins for displaying a carousel in fullscreen modal window.

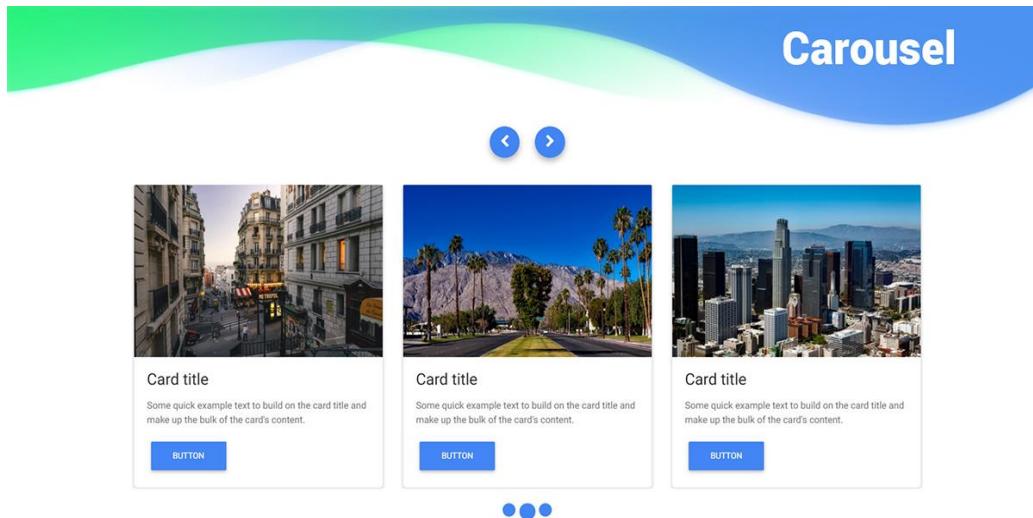


Image : Advance components of Bootstrap  
 Reference: <https://mdcdn.b-cdn.net/wp-content/uploads/2017/12/carousel.jpg>

## Checkbox

prettyCheckable – A jQuery plugin for replacing the default checkboxes and radio inputs.

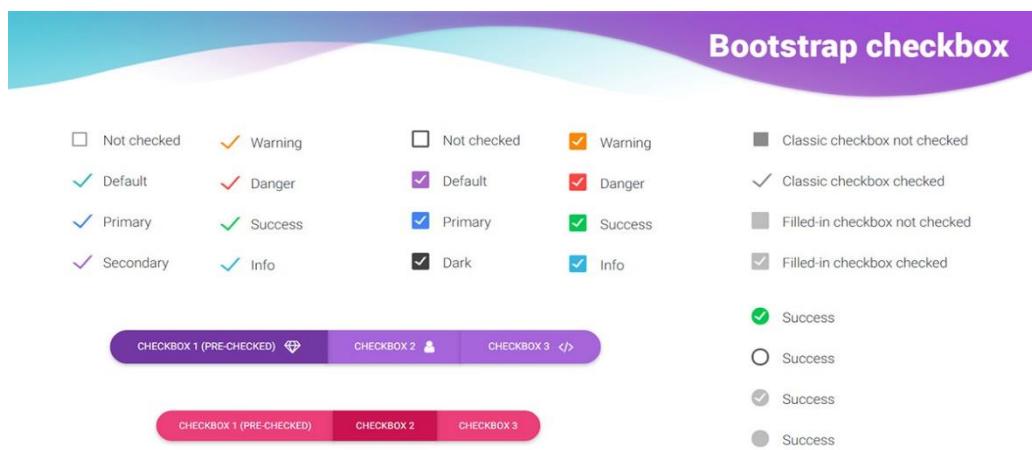


Image : Advance components of Bootstrap  
 Reference: <https://mdcdn.b-cdn.net/wp-content/uploads/2017/07/bootstrap-checkbox.jpg>

## Color Picker

Simple Color Picker – A very simple and lightweight (200 lines of JavaScript and 100 lines of CSS) jQuery color picker for Bootstrap.

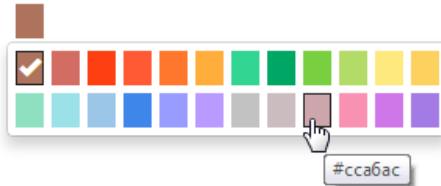


Image : Advance components of Bootstrap

Reference: <https://freefrontend.com/assets/img/jquery-color-picker-plugins/jquery-simplecolorpicker.png>

## Combobox

Bootstrap Combobox – A combobox plugin that integrates well with Bootstrap.

# Bootstrap Combobox



Image : Advance components of Bootstrap

Reference: <https://jquery-plugins.net/image/plugin/bootstrap-combobox.png>

## Contact Form

Bootstrap-Contact – A simple PHP contact form using Bootstrap and the jQuery validation plugin.

jQuery Gridform – A jQuery plugin for creating complex table-based forms with Bootstrap.

## jQuery Gridly

Gridly is a jQuery plugin to enable dragging and dropping as well as resizing on a grids. In the bricks.

### Example



Image : Advance components of Bootstrap  
Reference: <https://www.htmlion.com/img/jquery-plugins/Query-gridly.jpg>

## Datepicker

Datepicker – A plugin for adding a datepicker field to any element.

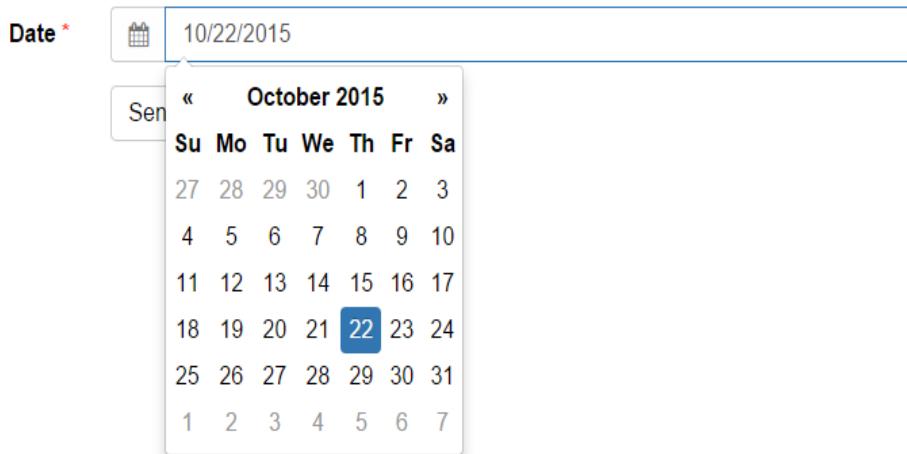


Image : Advance components of Bootstrap  
Reference: [https://formden.com/static/assets/img/posts/date-picker/example\\_date.png](https://formden.com/static/assets/img/posts/date-picker/example_date.png)

Date Range Picker – This date range picker component creates a drop-down from which you can select a range of dates.

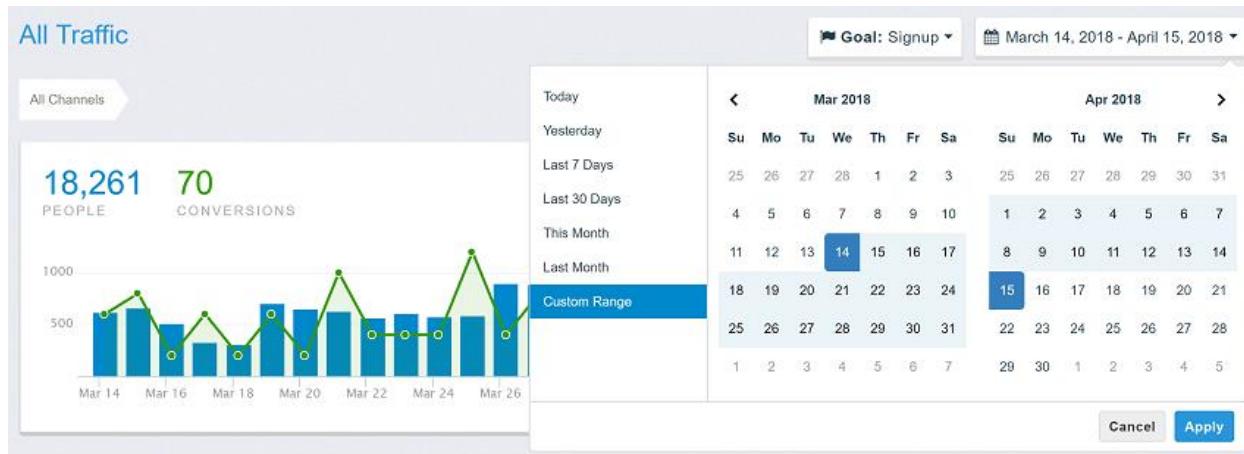


Image : Advance components of Bootstrap

Reference: <https://camo.githubusercontent.com/d7265660ea14a97c52ab3f6ce8684294b1df30892fcdb596f2c961ca8d1bfce2/68747470733a2f2f692e696d6775722e636f6d2f5554526c6161722e706e67>

ClockPicker – A unique clock-style timepicker for Bootstrap.

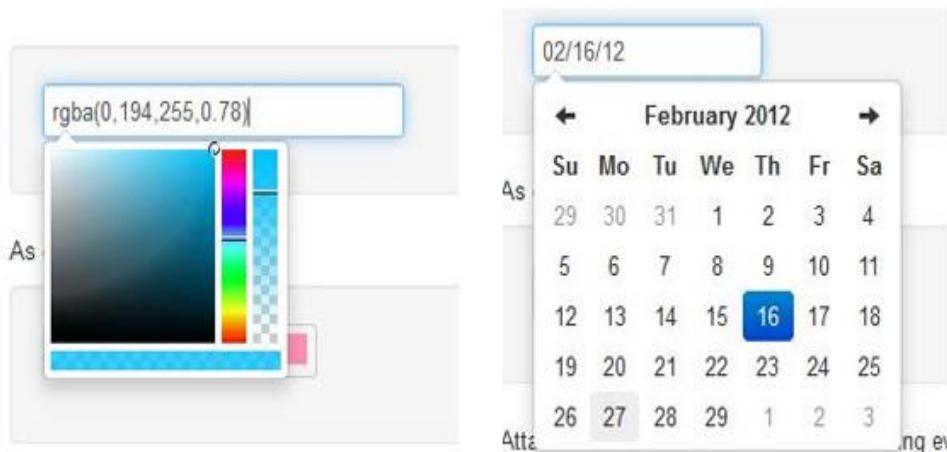


Image : Advance components of Bootstrap

Reference: <http://www.webappers.com/img/2012/07/bootstrap-datepicker-calendar.jpg>

Dialog Boxes & Alerts

Bootbox.js – A small JS library that allows you to create programmatic dialog boxes using Bootstrap's modals.

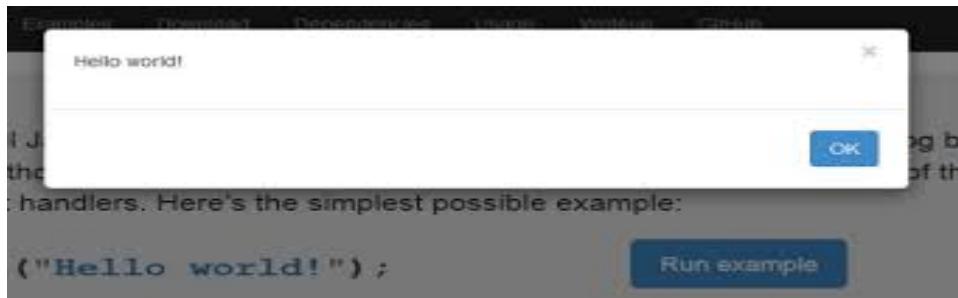


Image : Advance components of Bootstrap

Reference: <https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcRk-zNcmg5V4Klh1eaWwKwdTfjAMBEETQ1RA&usqp=CAU>

Bootstrap Prompts – An plugin to replace the alert(), prompt(), confirm() notifications on the browser when using Twitter Bootstrap with modals.

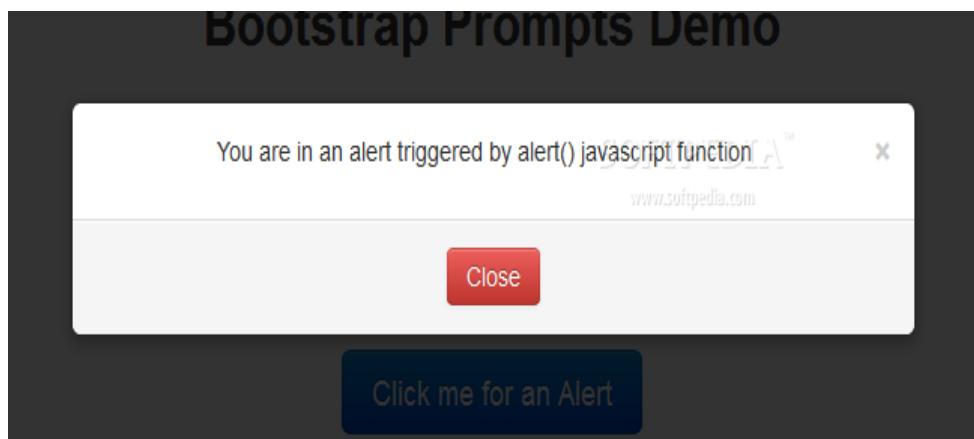


Image : Advance components of Bootstrap

Reference: [https://scripts-cdn.softpedia.com/screenshots/bootstrap-prompts\\_1.png](https://scripts-cdn.softpedia.com/screenshots/bootstrap-prompts_1.png)

Bootstrap Confirmation – A plugin that replaces popovers with confirmation dialogs.

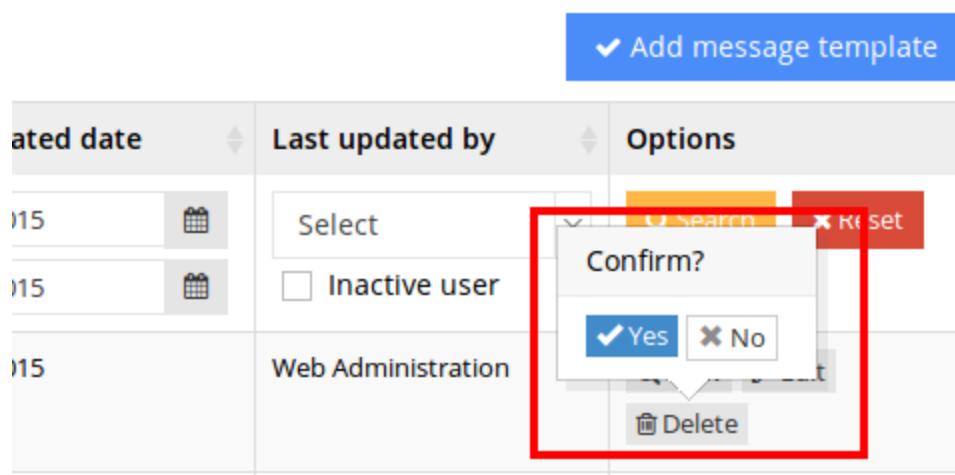


Image : Advance components of Bootstrap

Reference: <https://i.stack.imgur.com/cut2D.png>

## File Upload

jQuery File Upload – A file upload widget which features multiple file selection, drag & drop, progress bars, validation and preview images.

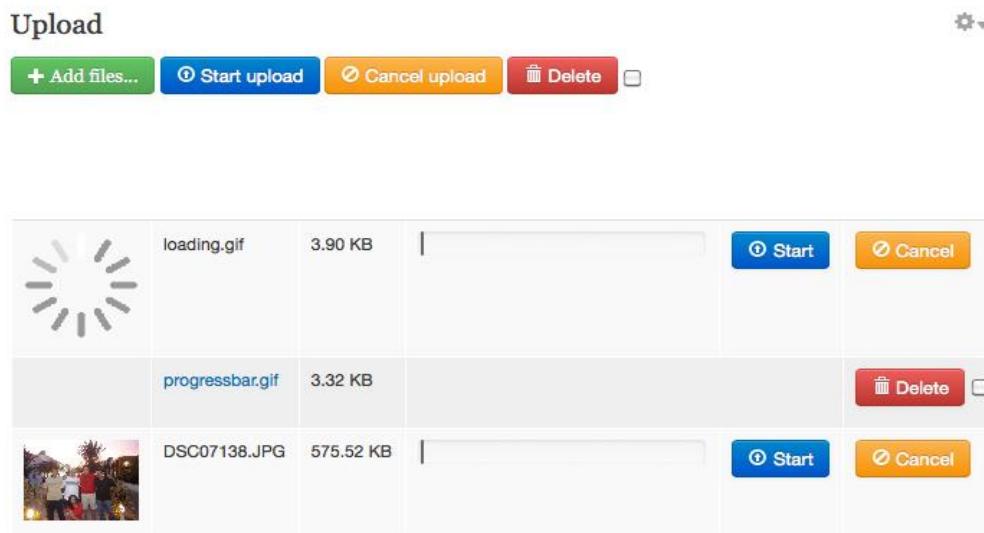


Image : Advance components of Bootstrap  
Reference: [https://www.drupal.org/files/project-images/jquery\\_file\\_upload.jpg](https://www.drupal.org/files/project-images/jquery_file_upload.jpg)

## Form Validation

BootstrapValidator – A jQuery plugin for validating forms within Bootstrap.

**Email**  
3456@123 !  
Bruh, that email address is invalid.

**Phone**  
abc !  
Only accept number.

**Password** **Confirm Password**  
... ✓ ... ✓

Image : Advance components of Bootstrap  
Reference: <https://i0.wp.com/www.cssscript.com/wp-content/uploads/2021/02/Bootstrap-Form-Validation-Library-Without-jQuery-Native-Validator.png?fit=602%2C448&ssl=1>

jqBootstrapValidation – Another jQuery validation framework for Bootstrap forms.

## jqBootstrapValidation

Email address

- This is required

Email address

- Not a valid email address

Legal  I agree to the [terms and conditions](#)

- You must agree to the terms and conditions

Image : Advance components of Bootstrap

Reference: <https://jquery-plugins.net/image/plugin/jqbootstrapvalidation-jquery-validation-plugin-for-bootstrap.png>

Validator – A simple and user-friendly form validator plugin for Bootstrap.

### Classic validation

Enter your name

Enter your

The field is

Enter your password

Enter your password

The field is required

Image : Advance components of Bootstrap

Reference: <https://i0.wp.com/www.cssscript.com/wp-content/uploads/2017/03/Just-validate.png?fit=462%2C341&ssl=1>

## HTML Tables

Tablecloth.js – Building off Bootstrap, this is a jQuery plugin that helps you easily style HTML tables.

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

Image : Advance components of Bootstrap

Reference: <http://formoid.com/articles/data/upload/2017/03/1inverse-bootstrap-table.jpg>

Navigable Table – A Bootstrap plugin for smooth navigation across table inputs.

**Bootstrap Navigable Table**  
Made by team Hoodie with love. Leave a tip or fork this.

Name	E-Mail	Birthday
Raphael Saadiq	male ▾	raphael@example.com
Dawn Robinson	female ▾	dawn@example.com
Ali Shaheed Muhammad	male ▾	ali@example.com
Joe Doe	female ▾	mm/dd/yyyy

**Supported keyboard shortcuts**

- jump alt + arrow keys
- move alt + shift + up/down
- insert alt + enter (+ shift)
- duplicate alt + d (+ shift)
- delete alt + shift + backspace

Image : Advance components of Bootstrap

Reference: <https://designposts.net/wp-content/uploads/2014/09/Bootstrap-Navigable-Table.jpg>

## Image Gallery

Image Gallery – This plugin shows images and videos in the modal dialog of the Bootstrap. It features swipe, mouse & keyboard navigation, transition effects, Fullscreen support and on-demand content loading.



Image : Advance components of Bootstrap  
 Reference: <https://mdbcdn.b-cdn.net/wp-content/uploads/2017/09/gallery-fb.jpg>

## In-Place Editing

X-editable – A library that allows you to create editable elements on your Bootstrap page.

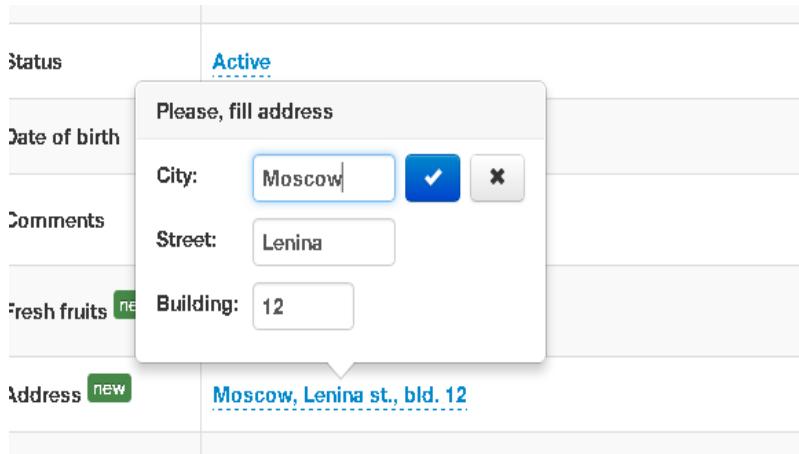


Image : Advance components of Bootstrap  
 Reference: <http://www.webappers.com/img/2012/12/editable-fields.png>

## Layout Grid

jQDrawBootstrapGrid – A simple jQuery plugin that draws grid columns to a Bootstrap enabled layout.



Image : Advance components of Bootstrap  
Reference: <https://www.webwash.net/wp-content/uploads/2018/01/d8-bootstrap-layouts-feature.png>

## Magnify

Magnify – A JS plugin for adding a magnifying glass to images on mouseover.



Image : Advance components of Bootstrap  
Reference:

<https://camo.githubusercontent.com/9dca757e32e3a73c530bae322463bda634071a20d75e4e4e4307181f922cfb57/68747470733a2f2f7261772e6769746875622e636f6d2f6d617263617562652f626f6f7473747261702d6d61676e6966792f6d61737465722f6578616d706c652f73637265656e73686f742e706e67>

## Modal Windows

Bootstrap Modal – This plugin extends Bootstrap's native modals to provide additional functionality (responsive, stackable, Ajax...).

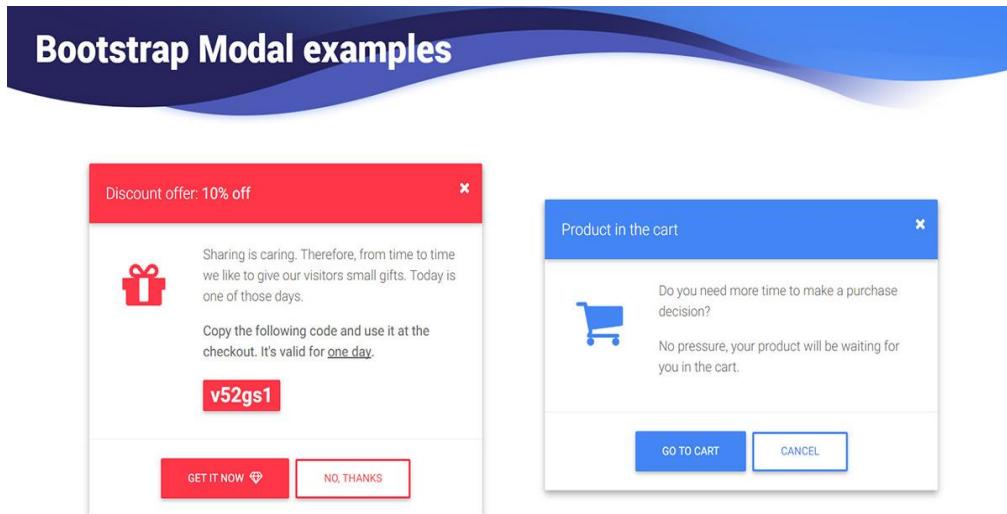


Image : Advance components of Bootstrap  
Reference: <https://mdbootstrap.com/docs/4.0/components/modal/>

Bootstrap Scroll Modal – A modification of the Bootstrap Modal plugin that allows for unlimited modal height with full page scrolling.

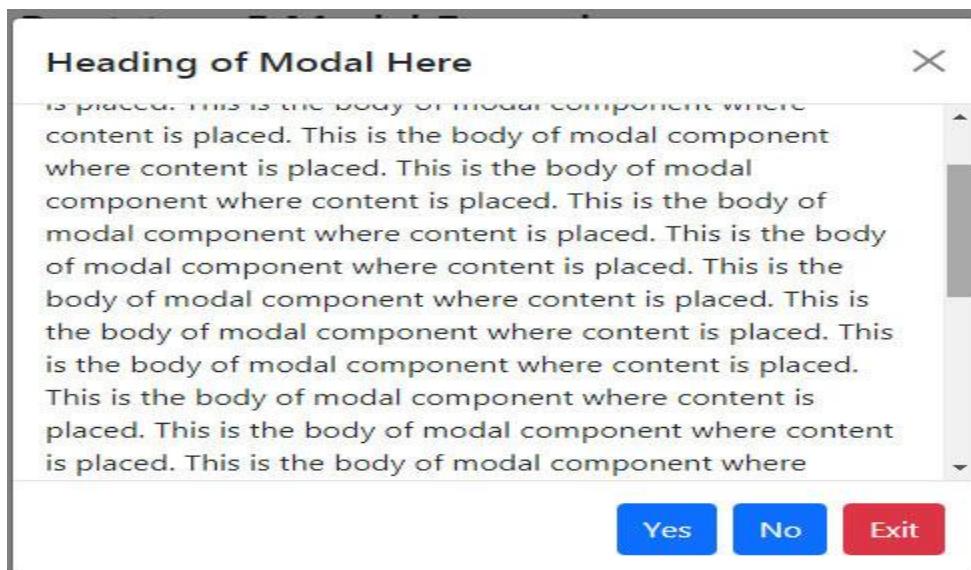


Image : Advance components of Bootstrap  
Reference: <https://www.jquery-az.com/wp-content/uploads/2021/06/1-2-modal-sccrollable.jpg>

## Navigation

Contextmenu – A context menu plugin for Bootstrap.



Image : Advance components of Bootstrap  
Reference: <https://www.codehim.com/wp-content/uploads/2019/08/context-menu-bootstrap.jpg>

## Pagination

bootpag – A jQuery plugin helps you create dynamic pagination with Bootstrap.

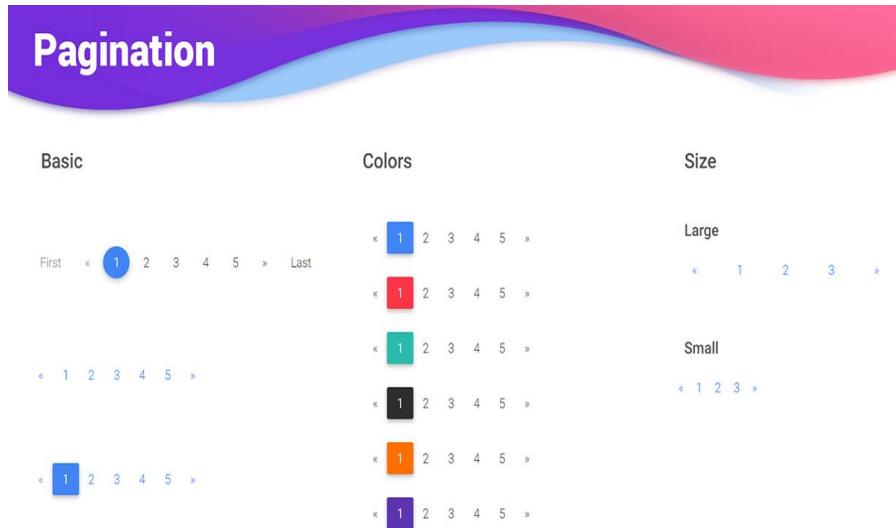


Image : Advance components of Bootstrap  
Reference: <https://mdbcdn.b-cdn.net/wp-content/uploads/2016/08/pagination.jpg>

## Progress Bars

Bootstrap Progressbar – A multi-color progress bar component for Bootstrap.

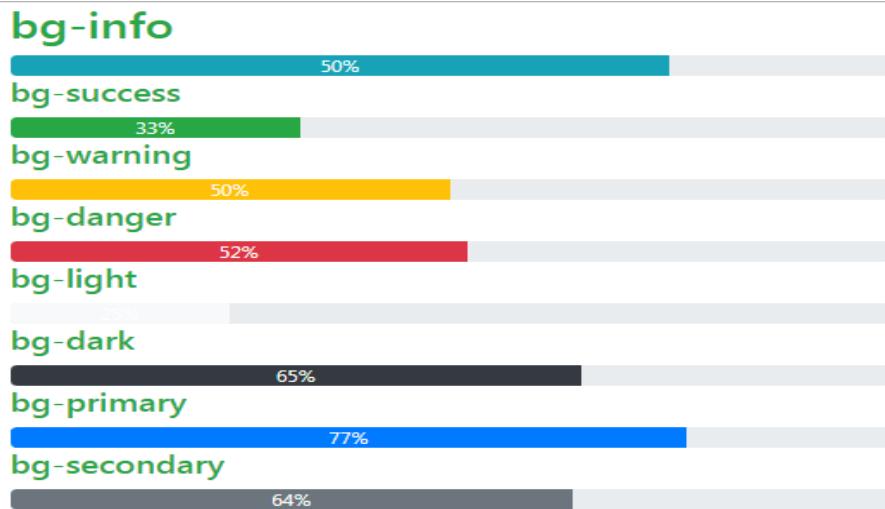


Image : Advance components of Bootstrap  
 Reference: <https://www.jquery-az.com/wp-content/uploads/2018/01/19-2-Bootstrap-4p-progress-colors.png>

## Ratings

Bootstrap Star Rating – A jQuery star rating plugin for Bootstrap that supports fractional star fill and RTL input support.

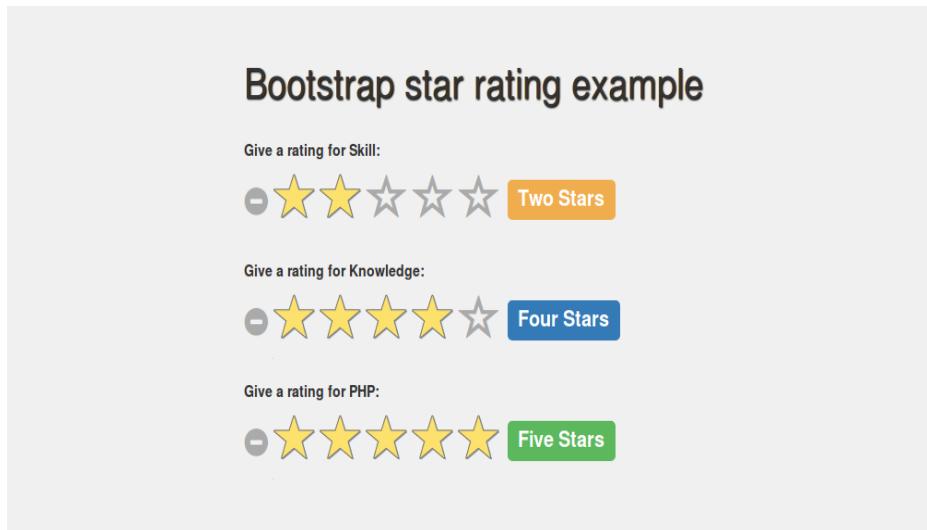


Image : Advance components of Bootstrap  
 Reference: <https://www.itsolutionstuff.com/upload/bootstrap-rating.png>

## Social Buttons

Social Buttons for Bootstrap – A pure CSS social sign-in button library.

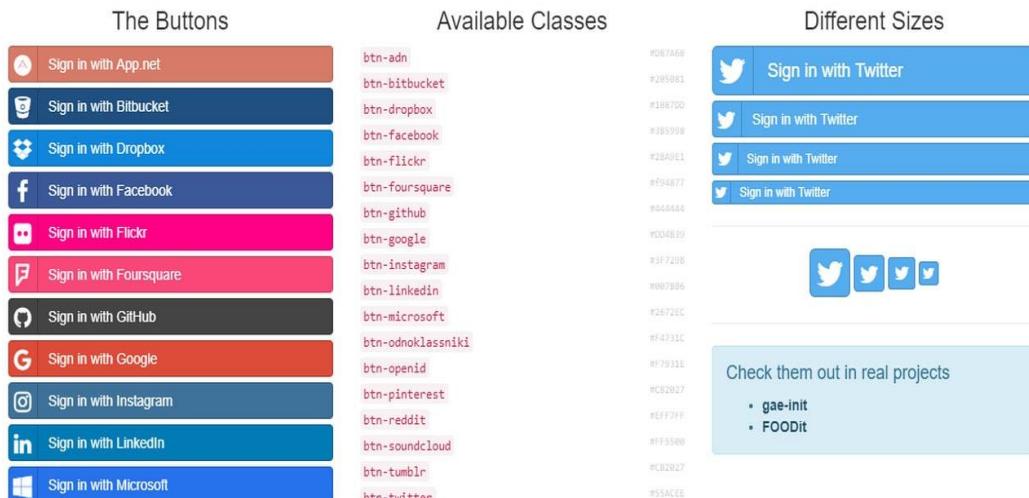


Image : Advance components of Bootstrap  
Reference: <https://desigmodo.com/wp-content/uploads/2019/08/6-Social-Buttons-for-Bootstrap.jpg>

## Tabs

Tabcordion.js – A simple jQuery plugin that transforms a set of Bootstrap tabs into a Bootstrap accordion.

## Bootstrap Tabs!

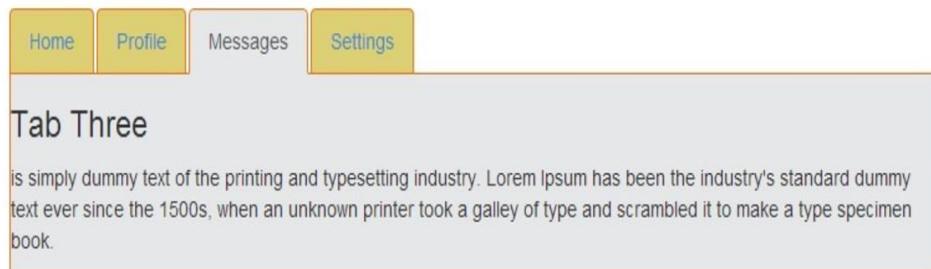


Image : Advance components of Bootstrap  
Reference: [https://images.saymedia-content.com/.image/t\\_share/MTc0Mjk3NDI1Njk4MzY2OTcy/apply-custom-styles-to-bootstrap-tabs-step-by-step.jpg](https://images.saymedia-content.com/.image/t_share/MTc0Mjk3NDI1Njk4MzY2OTcy/apply-custom-styles-to-bootstrap-tabs-step-by-step.jpg)

## Bootstrap 5 Utilities

### Background

With bootstrap, it's easy to add some background-color CSS rule in an element to convey a specific connotation using its predefined contextual background color classes which follow its built-in theme colors. These are composed of a subset of color palettes for generating color schemes.

Note: Background color utility classes do not set color or text color. You can use the contextual text color utility classes via `.text-*` color utilities if you want to achieve this style according to your constraint (more on this below).

The following are the supported background color utility classes that you can use when adding `background-color`:

Class	Description
<code>.bg-primary</code>	Apply background-color: <code>#0D6EFD</code> on an element.
<code>.bg-secondary</code>	Apply background-color: <code>#6C757D</code> on an element.
<code>.bg-success</code>	Apply background-color: <code>#198754</code> on an element.
<code>.bg-danger</code>	Apply background-color: <code>#DC3545</code> on an element.
<code>.bg-warning</code>	Apply background-color: <code>#FFC107</code> on an element.
<code>.bg-info</code>	Apply background-color: <code>#0DCAFO</code> on an element.
<code>.bg-light</code>	Apply background-color: <code>#F8F9FA</code> on an element.
<code>.bg-dark</code>	Apply background-color: <code>#212529</code> on an element.
<code>.bg-body</code>	Apply Bootstrap's default body background color on an element.
<code>.bg-white</code>	Apply background-color: <code>#FFFFFF</code> on an element.
<code>.bg-transparent</code>	Apply background-color: <code>transparent</code> on an element.

Image : Bootstrap 5 Utilities

Reference: <https://designmodo.com/wp-content/uploads/2021/04/1.png>

## Borders

Another CSS style that is regularly used in any layout design is border. The border properties allow you to define the style, width, and color of an element's border. With bootstrap, you can quickly style the border and border-radius of an element by using the predefined border utility classes.

The following are the supported border utility classes that you can use when adding a border to an element:

Class	Description
.border	Add a border on all sides of an element.
.border-top	Add a border at the top side of an element.
.border-right	Add a border on the right side of an element.
.border-left	Add a border on the left side of an element.
.border-0	Remove all borders from every side of an element.
.border-top-0	Remove the top side border of an element.
.border-right-0	Remove the right side border of an element.
.border-bottom-0	Remove the bottom side border of an element.
.border-left-0	Remove the left side border of an element.

Image : Bootstrap 5 Utilities

Reference: <https://designmodo.com/wp-content/uploads/2021/04/4.png>

The following are the border-color utilities built on bootstrap theme colors:

Class	Description
.border-primary	Apply border-color: #0D6EFD to an element's border.
.border-secondary	Apply border-color: #6C757D to an element's border.
.border-success	Apply border-color: #198754 to an element's border.
.border-danger	Apply border-color: #DC3545 to an element's border.
.border-warning	Apply border-color: #FFC107 to an element's border.
.border-info	Apply border-color: #0DCAFO to an element's border.
.border-light	Apply border-color: #F8F9FA to an element's border.
.border-dark	Apply border-color: #212529 to an element's border.
.border-white	Apply border-color: #FFFFFF to an element's border.

Image : Bootstrap 5 Utilities

Reference: <https://designmodo.com/wp-content/uploads/2021/04/5.png>

Additionally, you can also define the border thickness of an element using the border width utility classes:

Class	Description
.border-1	Apply border-width: <b>1px</b> to an element's border.
.border-2	Apply border-width: <b>2px</b> to an element's border.
.border-3	Apply border-width: <b>3px</b> to an element's border.
.border-4	Apply border-width: <b>4px</b> to an element's border.
.border-5	Apply border-width: <b>5px</b> to an element's border.

Image : Bootstrap 5 Utilities

Reference: <https://designmodo.com/wp-content/uploads/2021/04/6.png>

You can also use the border-radius utility classes to quickly add a radius or rounded corner(s) to an element:

Class	Description
.rounded	Apply rounded radius on all corners of an element.
.rounded-top	Apply rounded radius on top-left and top-right corners of an element.
.rounded-end	Apply rounded radius on top-right and bottom-right corners of an element.
.rounded-bottom	Apply rounded radius on bottom-left and bottom-right corners of an element.
.rounded-start	Apply rounded radius on top-left and bottom-left corners of an element.
.rounded-circle	Apply rounded radius on an element identical to a circle shape.
.rounded-pill	Apply rounded radius on an element identical to a pill shape.
.rounded-0	Remove rounded radius from all corners of an element.
.rounded-1	Apply border-radius: <b>.2rem</b> to an element.
.rounded-2	Apply border-radius: <b>.25rem</b> to an element.
.rounded-3	Apply border-radius: <b>.3rem</b> to an element.

Image : Bootstrap 5 Utilities

Reference: <https://designmodo.com/wp-content/uploads/2021/04/7.png>

## Color

You can also apply the same contextual colors that we used for the background and border color to every text element through bootstrap text color utility classes. These are frequently used for conveying meaning for a particular action or situation on your website or app.

The use of these utility classes is very straightforward. To do this, you can simply utilize the following classes and apply these classes directly to any of your text elements:

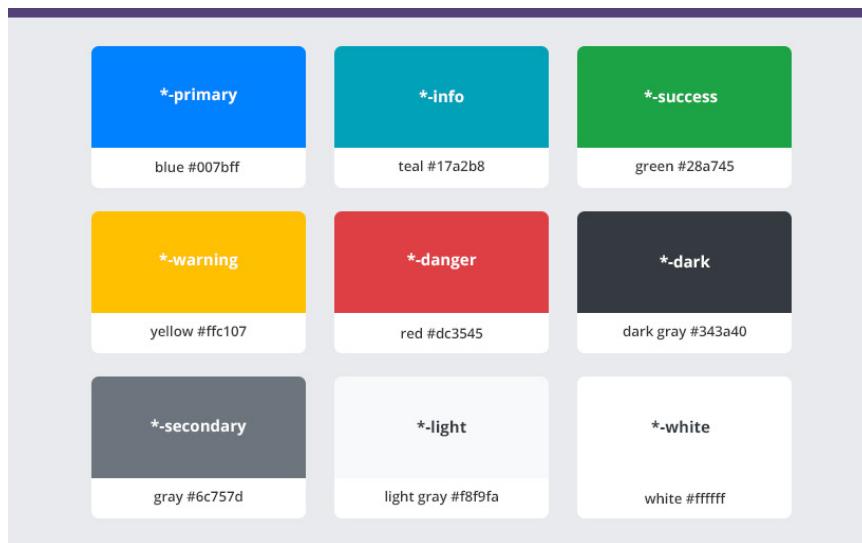


Image : Bootstrap 5 Utilities

Reference: <https://www.bitdegree.org/learn/storage/media/images/7383b588-563f-4117-82bd-4c866b5490bd.jpg>

## Display

Another helpful set of bootstrap utility classes that lets you easily and responsively toggle display value of a specific element in a specific breakpoint or viewport are the display utility classes.

These classes consist of a subset of classes that mostly use breakpoint or viewport abbreviation in them to control the responsive display by showing and hiding elements by screen resolution or device viewport width.

The following are the main supported display utility classes that you can use when dealing with a component's responsive toggle:

Class	Description
.d-none	Hide an element on all screen resolution or viewport size.
.d-inline	Enable an element to create an inline-level block container.
.d-inline-block	Enable an element to create an inline-level block container spanning through the content.
.d-block	Enable an element to create a block-level container.
.d-grid	Enable an element to create a grid-like container.
.d-table	Enable an element to act like a table.
.d-table-row	Enable an element to act like a row of cells in a table.
.d-table-cell	Enable an element to act like a data cell in a table.
.d-flex	Enable an element to act like a block-level flex container.
.d-inline-flex	Enable an element to act like an inline-level flex container.

Image : Bootstrap 5 Utilities

Reference: <https://desigmodo.com/wp-content/uploads/2021/04/10.png>

Additionally, display utility classes can be applied to all breakpoints, from xs to xxl. You can use these utility classes to show or hide elements on your preferred screen resolution or viewport width using the format .d-{breakpoint}-{value} for sm, md, lg, xl, and xxl. For instance, you can use .d-md-none, .d-sm-block, .d-md-flex and the list go on.

Lastly, you can change the display value of an element when printing via print display utility classes. These display in print utility classes support for the same display values as the responsive .d-\* utilities. You can do this simply by adding the print- after .d-\*.

The following are the list of all the display in print utility classes.

- .d-print-none
- .d-print-inline
- .d-print-inline-block
- .d-print-block
- .d-print-grid
- .d-print-table
- .d-print-table-row
- .d-print-table-cell
- .d-print-flex
- .d-print-inline-flex

## Flex

Flexbox offers a better way to organize elements in a web page in a predictable manner. While it sometimes performs like a float, it offers a lot more than that such as reordering elements and avoiding known issues of float.

In version 4 of Bootstrap, flexbox support has finally arrived. Using display properties via display utilities, it's easy to create a flexbox container and transform direct children elements into flex items.

Bootstrap comes with the following main flexbox utility classes:

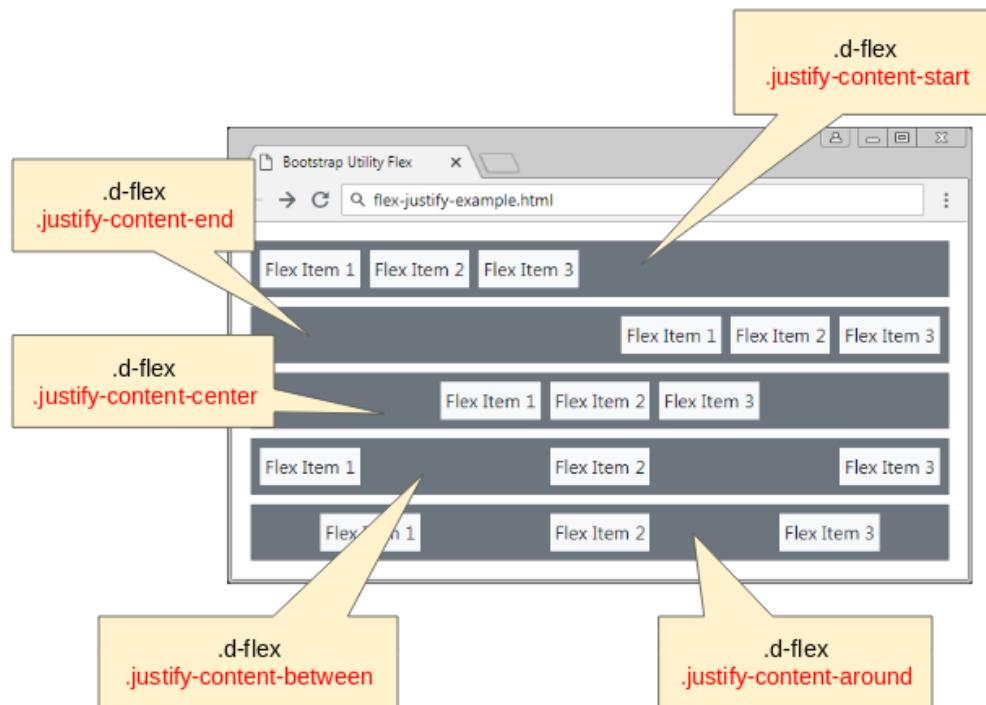


Image : Bootstrap 5 Utilities

Reference: <https://s1.07planning.com/en/12023/images/22980304.png>

Additionally, responsive variations exist on most of these flexbox utility classes. As an example for `.d-flex`, you can use classes on specific breakpoints or viewports such as `.d-sm-flex` or `.d-xl-flex`.

## Interactions

Bootstrap 5 also provides CSS property controls that allow users to interact with content. This determines whether the user can select text or not and if a specific pointer event is active in a text element. This doesn't have any effect on content loaded as part of a browser's user interface except in textboxes.

The following are the supported interactions utility classes:

Class	Description
.user-select-all	Enable users to select a text element upon click.
.user-select-auto	Default select behavior on text element.
.user-select-none	Prevent users from selecting a text element including its sub-elements.
.pe-none	Prevents interactions with a pointer (mouse, stylus, touch).
.pe-auto	Default element's pointer behavior.

Image : Bootstrap 5 Utilities

Reference: <https://desigmodo.com/wp-content/uploads/2021/04/14.png>

## Overflow

With bootstrap, it's also easy to set your preferred behavior for an element's overflow using the overflow utility classes. For instance, when an element's content is too big to fit in its container context, you can specify whether to clip content in both directions or add a scrollbar on it.

Bootstrap provides the following overflow utility classes that you can use on your elements right out of the box:

Class	Description
.overflow-auto	If the content is clipped, a scroll-bar will be added to view the rest of the content.
.overflow-hidden	Content is clipped. The rest of the content will be invisible with no scroll-bar provided.
.overflow-visible	Content is not clipped and renders outside the element's box.
.overflow-scroll	Content is clipped. Display scroll-bar whether or not any content is actually clipped to see the rest of the content.

Image : Bootstrap 5 Utilities

Reference: <https://desigmodo.com/wp-content/uploads/2021/04/16.png>

## Position

Another useful set of bootstrap utility classes are the position utilities. These classes allow you to define the type of positioning method and final location you want an element to behave in a web page.

Bootstrap offers the following position and edge positioning utility classes on the fly:

Class	Description
.position-static	The element is positioned static by default according to the normal flow of the web page.
.position-relative	The element is positioned relative according to the normal flow of the web page which can be affected relatively to itself through the values of top, right, bottom, and left.
.position-absolute	The element is positioned relative to the closest position ancestor with no space generated for the element in the web page. Its position can be affected by the values of top, right, bottom, and left.
.position-fixed	The element is positioned on the same place even if the page is scrolled and is relative to its initial ancestor elements or block. Its final position is determined by the values of top, right, bottom, and left.
.position-sticky	The element is positioned based on the position of the web page scroll. Its position can be affected by the values of top, right, bottom, and left.
.top-*	Use for setting up the vertical top position.
.start-*	Use for setting up the horizontal left position (in LTR).
.bottom-*	Use for setting up the vertical bottom position.
.end-*	Use for setting up the horizontal right position (in LTR).
.[top/start/bottom/end]-0	Set 0 edge position.
.[top/start/bottom/end]-50	Set 50% edge position.
.[top/start/bottom/end]-100	Set 100% edge position.
.translate-middle	Applies the transformations <code>translateX(-50%)</code> and <code>translateY(-50%)</code> to the element which, in combination with the edge positioning utilities, allows you to absolute center an element.

Image : Bootstrap 5 Utilities

Reference: <https://designmodo.com/wp-content/uploads/2021/04/18.png>

## Box Shadow

With the box-shadow CSS property, you can cast shadow effects around an element's frame which is determined by X and Y offsets. Bootstrap 5 also has its own out-of-the-box utility classes to quickly add box shadows to your elements.

The following are the supported box-shadow utility classes:

Class	Description
.shadow-none	Removes box-shadow in an element.
.shadow	Add a basic box-shadow in an element.
.shadow-sm	Add a small amount of box-shadow effect in an element.
.shadow-lg	Add a large amount of box-shadow effect in an element.

Image : Bootstrap 5 Utilities

Reference: <https://desigmodo.com/wp-content/uploads/2021/04/20.png>

## Sizing

One of the important factors in web design is the responsive sizes of each element that can span or shrink in size across different screen resolutions or viewport widths.

Aside from the responsive columns and flexbox classes, bootstrap also offers sizing utility classes that can make an element as wide or tall depending on the layout you are trying to achieve.

This became possible through the sizing utility classes provided by bootstrap out of the box.

Class	Description
.w-25	Set the width of an element to 25% of its parent container.
.w-50	Set the width of an element to 50% of its parent container.
.w-75	Set the width of an element to 75% of its parent container.
.w-100	Set the width of an element to 100% of its parent container.
.w-auto	Set the width of an element to auto.
.mw-100	Set the max-width of an element to 100%.
.h-25	Set the height of an element to 25% of its parent container.
.h-50	Set the height of an element to 50% of its parent container.
.h-75	Set the height of an element to 75% of its parent container.
.h-100	Set the height of an element to 100% of its parent container.
.h-auto	Set the height of an element to auto.
.mh-100	Set the max-height of an element to 100%.
.vw-100	Set the width of an element to 100% of the width of the screen resolution or viewport.
.min-vw-100	Set the min-width of an element to 100% of the width of the screen resolution or viewport.
.vh-100	Set the height of an element to 100% of the height of the screen resolution or viewport.
.min-vh-100	Set the min-height of an element to 100% of the height of the screen resolution or viewport.

Image : Bootstrap 5 Utilities

Reference: <https://desigmodo.com/wp-content/uploads/2021/04/22.png>

## Spacing

Bootstrap offers a variety of shorthand responsive margin, padding, and gap utility classes to modify an element position or appearance. These spacing utility classes can be used on any supported breakpoints, from xs to xxl and don't have any breakpoint abbreviation in them. The standard measurement for each margin, padding and gap classes are ranging from .25rem to 3rem.

These classes are formatted {property}{sides}-{size} for xs and {property}{sides}-{breakpoint}-{size} for sm, md, lg, xl, and xxl. Available sizes starts from 0 up to 5 and an extra auto keyword to set the margin to auto.

The following are supported spacing utility classes:

Class	Description
.mt-*	Set a top margin on an element.
.mb-*	Set a bottom margin on an element.
.ml-*	Set a left margin on an element.
.mr-*	Set a right margin on an element.
.mx-*	Set a left and right margin on an element.
.my-*	Set a top and bottom margin on an element.
.pt-*	Set a top padding on an element.
.pb-*	Set a bottom padding on an element.
.pl-*	Set a left padding on an element.
.pr-*	Set a right padding on an element.
.px-*	Set a left and right padding on an element.
.py-*	Set a top and bottom padding on an element.
.gap-*	Set a gap on the parent grid container

Image : Bootstrap 5 Utilities

Reference: <https://desgnmodo.com/wp-content/uploads/2021/04/24.png>

## Text

Bootstrap 5 also added a few new extra utility classes for common text or link control. With these classes, you can easily realign text to components, wrap text, modify the font size or weight, transform case and more. Just like the previous set of utility classes that I introduced and demonstrated above, responsive classes are also available on some of these text utility classes which you can use with viewport width breakpoints as the grid system.

The text alignment classes used for aligning text:

- .text-start - It places the text to the left on all types of the viewport.
- .text-end - It places the text to the right on all types of the viewport.
- .text-center - It places the text to the center on all types of the viewport
- .text-sm-start - It places the text to the left on small sizes viewports.
- .text-md-end - It places the text to the right on medium size viewport.

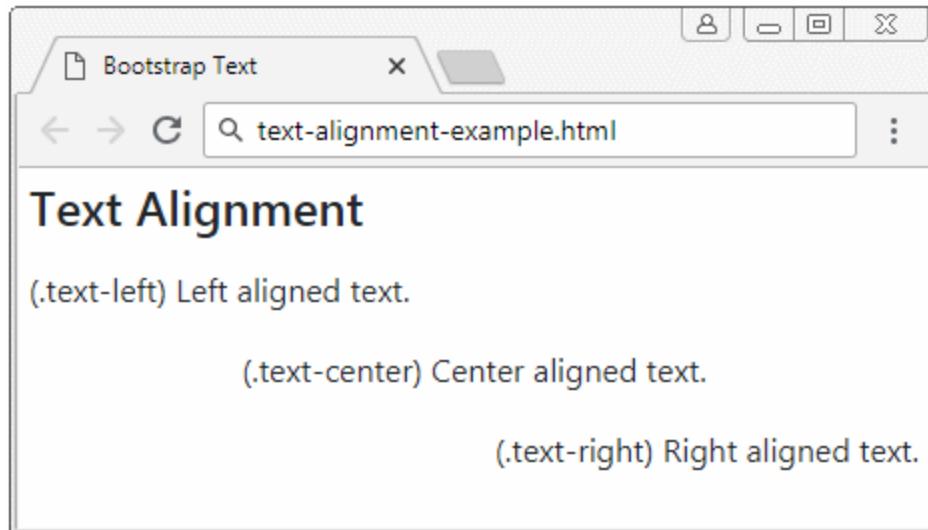


Image : Bootstrap 5 Utilities  
Reference: <https://s1.o7planning.com/en/12071/images/24027841.gif>

## Introduction to jQuery

jQuery is an open source JavaScript library that simplifies the interactions between an HTML/CSS document, or more precisely the Document Object Model (DOM), and JavaScript.

Elaborating the terms, jQuery simplifies HTML document traversing and manipulation, browser event handling, DOM animations, Ajax interactions, and cross-browser JavaScript development.

jQuery is widely famous with its philosophy of “Write less, do more.” This philosophy can be further elaborated as three concepts:

Finding some elements (via CSS selectors) and doing something with them (via jQuery methods) i.e. locate a set of elements in the DOM, and then do something with that set of elements.

Chaining multiple jQuery methods on a set of elements

Using the jQuery wrapper and implicit iteration

### What is jQuery?

jQuery is a lightweight, "write less, do more", JavaScript library. The purpose of jQuery is to make it much easier to use JavaScript on your website.

jQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish, and wraps them into methods that you can call with a single line of code. jQuery also simplifies a lot of the complicated things from JavaScript, like AJAX calls and DOM manipulation.

The jQuery library contains the following features:

- HTML/DOM manipulation
- CSS manipulation
- HTML event methods
- Effects and animations
- AJAX
- Utilities

**Basic syntax for any jQuery function is:**

```
$(selector).action()
```

- A \$ sign is to define/access jQuery
- A (selector) is to “query (or find)” HTML elements in html page
- A jQuery action() is the action to be performed on the selected element(s)

## Why jQuery?

Some of the key points which support the answer for why to use jQuery:

- It is incredibly popular, which is to say it has a large community of users and a healthy amount of contributors who participate as developers and evangelists.
- It normalizes the differences between web browsers so that you don't have to.
- It is intentionally a lightweight footprint with a simple yet clever plugin architecture.
- Its repository of plugins is vast and has seen steady growth since jQuery's release.
- Its API is fully documented, including inline code examples, which in the world of JavaScript libraries is a luxury. Heck, any documentation at all was a luxury for years.
- It is friendly, which is to say it provides helpful ways to avoid conflicts with other JavaScript libraries.

### Advantages:

- Wide range of plug-ins. jQuery allows developers to create plug-ins on top of the JavaScript library.
- Large development community
- It has a good and comprehensive documentation
- It is a lot more easy to use compared to standard javascript and other javascript libraries.
- JQuery lets users develop Ajax templates with ease, Ajax enables a sleeker interface where actions can be performed on pages without requiring the entire page to be reloaded.
- Being Light weight and a powerful chaining capabilities makes jQuery more strong.

### Disadvantages:

- While JQuery has an impressive library in terms of quantity, depending on how much customization you require on your website, the functionality may be limited thus using raw javascript may be inevitable in some cases.
- The JQuery javascript file is required to run JQuery commands, while the size of this file is relatively small (25-100KB depending on the server), it is still a strain on the client computer and maybe your web server as well if you intend to host the JQuery script on your own web server.

## Adding jQuery to Your Web Pages

There are several ways to start using jQuery on your web site. You can:

- Download the jQuery library from [jQuery.com](http://jQuery.com)
- Include jQuery from a CDN, like Google

## Downloading jQuery

There are two versions of jQuery available for downloading:

- Production version - this is for your live website because it has been minified and compressed
- Development version - this is for testing and development (uncompressed and readable code)

Both versions can be downloaded from [jQuery.com](http://jQuery.com).

The jQuery library is a single JavaScript file, and you reference it with the HTML <script> tag (notice that the <script> tag should be inside the <head> section):

```
<head>
<script src="jquery-3.5.1.min.js"></script>
</head>
```

## jQuery CDN

If you don't want to download and host jQuery yourself, you can include it from a CDN (Content Delivery Network).

Google is an example of someone who host jQuery:

```
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
</head>
```

## jQuery Syntax

The jQuery syntax is tailor-made for selecting HTML elements and performing some action on the element(s).

Basic syntax is:

```
$(selector).action()
```

- A \$ sign to define/access jQuery
- A (selector) to "query (or find)" HTML elements
- A jQuery action() to be performed on the element(s)

Examples:

1. `$(this).hide()` - hides the current element.
2. `$("p").hide()` - hides all `<p>` elements.
3. `$(".test").hide()` - hides all elements with `class="test"`.
4. `$("#test").hide()` - hides the element with `id="test"`.

## The Document Ready Event

You might have noticed that all jQuery methods in our examples, are inside a document ready event:

```
$(document).ready(function(){  
    // jQuery methods go here...  
});
```

This is to prevent any jQuery code from running before the document is finished loading (is ready).

It is good practice to wait for the document to be fully loaded and ready before working with it. This also allows you to have your JavaScript code before the body of your document, in the head section.

```
$("p")
```

Here are some examples of actions that can fail if methods are run before the document is fully loaded:

- Trying to hide an element that is not created yet
- Trying to get the size of an image that is not loaded yet

```
$(function(){  
  
    // jQuery methods go here...  
  
});
```

Tip: The jQuery team has also created an even shorter method for the document ready event:

## **jQuery Selectors**

jQuery selectors allow you to select and manipulate HTML element(s).

jQuery selectors are used to "find" (or select) HTML elements based on their name, id, classes, types, attributes, values of attributes and much more. It's based on the existing CSS Selectors, and in addition, it has some own custom selectors.

All selectors in jQuery start with the dollar sign and parentheses: `$( )`.

### The element Selector

The jQuery element selector selects elements based on the element name.

You can select all `<p>` elements on a page like this:

#### Example

When a user clicks on a button, all `<p>` elements will be hidden:

```
<!DOCTYPE html>  
  
<html>
```

```
<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">

</script>

<script>

$(document).ready(function(){

  $("button").click(function(){

    $("p").hide();

  });

});

</script>

</head>

<body>

<h2>This is a heading</h2>

<p>This is a paragraph.</p>

<button>Click me to hide paragraphs</button>

</body>

</html>
```

## The #id Selector

The jQuery #id selector uses the id attribute of an HTML tag to find the specific element.

An id should be unique within a page, so you should use the #id selector when you want to find a single, unique element.

To find an element with a specific id, write a hash character, followed by the id of the HTML element:

```
$("#test")
```

### Example

When a user clicks on a button, the element with id="test" will be hidden:

```
<!DOCTYPE html>

<html>
<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
</script>
<script>

$(document).ready(function(){

  $("button").click(function(){

    $("#test").hide();

  });

});

</script>
```

```
</head>

<body>
<h2>This is a heading</h2>
<p>This is a paragraph.</p>
<p id="test">This is another paragraph.</p>
<button>Click me</button>

</body>
</html>
```

## The .class Selector

The jQuery .class selector finds elements with a specific class. To find elements with a specific class, write a period character, followed by the name of the class:

```
$(".test")
```

### Example

When a user clicks on a button, the elements with class="test" will be hidden:

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
```

```
$( ".test" ).hide();  
});  
});  
</script>  
</head>  
<body>  
<h2 class="test">This is a heading</h2>  
<p class="test">This is a paragraph.</p>  
<p>This is another paragraph.</p>  
<button>Click me</button>  
</body>  
</html>
```

## More Examples of jQuery Selectors

Syntax	Description
<code>\$( "*")</code>	Selects all elements
<code>\$(this)</code>	Selects the current HTML element
<code>\$( "p.intro" )</code>	Selects all <code>&lt;p&gt;</code> elements with <code>class="intro"</code>
<code>\$( "p:first" )</code>	Selects the first <code>&lt;p&gt;</code> element
<code>\$( "ul li:first" )</code>	Selects the first <code>&lt;li&gt;</code> element of the first <code>&lt;ul&gt;</code>

\$(“ul li:first-child”)	Selects the first <code>&lt;li&gt;</code> element of every <code>&lt;ul&gt;</code>
\$(“[href]”)	Selects all elements with an <code>href</code> attribute
\$(“a[target=_blank]”)	Selects all <code>&lt;a&gt;</code> elements with a <code>target</code> attribute value equal to <code>_blank</code>
\$(“a[target!=_blank]”)	Selects all <code>&lt;a&gt;</code> elements with a <code>target</code> attribute value NOT equal to <code>_blank</code>
\$(“:button”)	Selects all <code>&lt;button&gt;</code> elements and <code>&lt;input&gt;</code> elements of <code>type="button"</code>
\$(“tr:even”)	Selects all even <code>&lt;tr&gt;</code> elements
\$(“tr:odd”)	Selects all odd <code>&lt;tr&gt;</code> elements

## jQuery Attributes

Some of the most basic components we can manipulate when it comes to DOM elements are the properties and attributes assigned to those elements.

Most of these attributes are available through JavaScript as DOM node properties. Some of the more common properties are –

- `className`
- `tagName`
- `id`
- `href`
- `title`
- `rel`
- `src`

Consider the following HTML markup for an image element –

```
<img id = "imageid" src = "image.gif" alt = "Image" class = "myclass"
```

In this element's markup, the tag name is img, and the markup for id, src, alt, class, and title represents the element's attributes, each of which consists of a name and a value.

jQuery gives us the means to easily manipulate an element's attributes and gives us access to the element so that we can also change its properties.

## Get Attribute Value

The attr() method can be used to either fetch the value of an attribute from the first element in the matched set or set attribute values onto all matched elements.

### Example

Following is a simple example which fetches title attribute of <em> tag and set <div id = "divid"> value with the same value –

```
<html>
  <head>
    <title>The jQuery Example</title>
    <script type = "text/javascript"
      src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">
    </script>

    <script type = "text/javascript" language = "javascript">
      $(document).ready(function() {
        var title = $("em").attr("title");
        $("#divid").text(title);
      });
    </script>
  </head>
```

```
<body>
  <div>
    <em title = "Bold and Brave">This is first paragraph.</em>
    <p id = "myid">This is second paragraph.</p>
    <div id = "divid"></div>
  </div>
</body>
</html>
```

This will produce following result –

*This is first paragraph.*

This is second paragraph.

**Bold and Brave**

## Set Attribute Value

The attr(name, value) method can be used to set the named attribute onto all elements in the wrapped set using the passed value.

### Example

Following is a simple example which set src attribute of an image tag to a correct location –

```
<html>
  <head>
    <title>The jQuery Example</title>
```

```
<script type = "text/javascript"
src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">
</script>

<script type = "text/javascript" language = "javascript">
$(document).ready(function() {
    $("#myimg").attr("src", "/jquery/images/jquery.jpg");
});
</script>

</head>

<body>
<div>
<img id = "myimg" src = "/images/jquery.jpg" alt = "Sample image" />
</div>
</body>
</html>
```

This will produce following result –



## Applying Styles

The `addClass( classes )` method can be used to apply defined style sheets onto all the matched elements. You can specify multiple classes separated by space.

### Example

Following is a simple example which sets class attribute of a para `<p>` tag –

```
<html>
  <head>
    <title>The jQuery Example</title>
    <script type = "text/javascript"
      src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">
    </script>

    <script type = "text/javascript" language = "javascript">
      $(document).ready(function() {
        $("em").addClass("selected");
        $("#myid").addClass("highlight");
      });
    </script>

    <style>
      .selected { color:red; }
      .highlight { background:yellow; }
    </style>
  </head>
```

```
<body>  
  <em title = "Bold and Brave">This is first paragraph.</em>  
  <p id = "myid">This is second paragraph.</p>  
</body>  
</html>
```

This will produce following result –

*This is first paragraph.*

This is second paragraph.

## Attribute Methods

Following table lists down few useful methods which you can use to manipulate attributes and properties –

S.No	Methods	Description
1	attr(properties)	Set a key/value object as properties to all matched elements.
2	attr(key, fn)	Set a single property to a computed value, on all matched elements.
3	removeAttr(name)	Remove an attribute from each of the matched elements.
4	hasClass(class)	Returns true if the specified class is present on at least one of the set of matched elements.

5	removeClass(class)	Removes all or the specified class(es) from the set of matched elements.
6	toggleClass(class)	Adds the specified class if it is not present, removes the specified class if it is present.
7	html()	Get the html contents (innerHTML) of the first matched element.
8	html(val)	Set the html contents of every matched element.
9	text()	Get the combined text contents of all matched elements.
10	text(val)	Set the text contents of all matched elements.
11	val()	Get the input value of the first matched element.
12	val(val)	Set the value attribute of every matched element if it is called on <input> but if it is called on <select> with the passed <option> value then passed option would be selected, if it is called on check box or radio box then all the matching check box and radiobox would be checked.

## jQuery - DOM Traversing

jQuery is a very powerful tool which provides a variety of DOM traversal methods to help us select elements in a document randomly as well as in sequential method. Most of the DOM Traversal Methods do not modify the jQuery object and they are used to filter out elements from a document based on given conditions.

### Find Elements by Index

Following is a simple example which adds the color to second list item.

```
<html>
```

```
<head>

<title>The JQuery Example</title>

<script type = "text/javascript"

src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">

</script>

<script type = "text/javascript" language = "javascript">

$(document).ready(function() {

    $("li").eq(2).addClass("selected");

});

</script>

<style>

.selected { color:red; }

</style>

</head>

<body>

<div>

<ul>

    <li>list item 1</li>

    <li>list item 2</li>

    <li>list item 3</li>

    <li>list item 4</li>

    <li>list item 5</li>


```

```
<li>list item 6</li>
</ul>
</div>
</body>
</html>
```

This will produce following result –

- list item 1
- list item 2
- **list item 3**
- list item 4
- list item 5
- list item 6

## Filtering out Elements

The filter( selector ) method can be used to filter out all elements from the set of matched elements that do not match the specified selector(s). The selector can be written using any selector syntax.

### Example

Following is a simple example which applies color to the lists associated with middle class –

```
<html>
<head>
<title>The JQuery Example</title>
<script type = "text/javascript"
src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">
</script>

<script type = "text/javascript" language = "javascript">
```

```
$(document).ready(function() {  
    $("li").filter(".middle").addClass("selected");  
});  
</script>  
  
<style>  
    .selected { color:red; }  
</style>  
</head>  
  
<body>  
    <div>  
        <ul>  
            <li class = "top">list item 1</li>  
            <li class = "top">list item 2</li>  
            <li class = "middle">list item 3</li>  
            <li class = "middle">list item 4</li>  
            <li class = "bottom">list item 5</li>  
            <li class = "bottom">list item 6</li>  
        </ul>  
    </div>  
</body>  
</html>
```

This will produce following result –

- list item 1
- list item 2
- list item 3
- list item 4
- list item 5
- list item 6

## Locating Descendant Elements

The `find( selector )` method can be used to locate all the descendant elements of a particular type of elements. The selector can be written using any selector syntax.

### Example

Following is an example which selects all the `<span>` elements available inside different `<p>` elements –

```
<html>
  <head>
    <title>The JQuery Example</title>
    <script type = "text/javascript"
      src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">
    </script>

    <script type = "text/javascript" language = "javascript">
      $(document).ready(function() {
        $("p").find("span").addClass("selected");
      });
    </script>
```

```
<style>
  .selected { color:red; }

</style>

</head>

<body>
  <p>This is 1st paragraph and <span>THIS IS RED</span></p>
  <p>This is 2nd paragraph and <span>THIS IS ALSO RED</span></p>
</body>

</html>
```

This will produce following result –

This is 1st paragraph and **THIS IS RED**

This is 2nd paragraph and **THIS IS ALSO RED**

## jQuery DOM Filter Methods

Following table lists down useful methods which you can use to filter out various elements from a list of DOM elements –

S.No	Methods	Description
1	eq(index)	Reduce the set of matched elements to a single element.
2	filter(selector)	Removes all elements from the set of matched elements that do not match the specified selector(s).

3	filter(fn)	Removes all elements from the set of matched elements that do not match the specified function.
4	is(selector)	Checks the current selection against an expression and returns true, if at least one element of the selection fits the given selector.
5	map(callback)	Translate a set of elements in the jQuery object into another set of values in a jQuery array (which may, or may not contain elements).
6	not(selector)	Removes elements matching the specified selector from the set of matched elements.
7	slice(start,[end])	Selects a subset of the matched elements.

## jQuery DOM Traversing Methods

Following table lists down other useful methods which you can use to locate various elements in a DOM –

S.No	Methods	Description
1	add(selector)	Adds more elements, matched by the given selector, to the set of matched elements.
2	andSelf()	Add the previous selection to the current selection.
3	children([selector])	Get a set of elements containing all of the unique immediate children of each of the matched set of elements.

<b>4</b>	closest(selector)	Get a set of elements containing the closest parent element that matches the specified selector, the starting element included.
<b>5</b>	contents()	Find all the child nodes inside the matched elements (including text nodes), or the content document, if the element is an iframe.
<b>6</b>	end()	Revert the most recent 'destructive' operation, changing the set of matched elements to its previous state.
<b>7</b>	find(selector)	Searches for descendant elements that match the specified selectors.
<b>8</b>	next([selector])	Get a set of elements containing the unique next siblings of each of the given set of elements.
<b>9</b>	nextAll(selector)	Find all sibling elements after the current element.
<b>10</b>	offsetParent()	Returns a jQuery collection with the positioned parent of the first matched element.
<b>11</b>	parent([selector])	Get the direct parent of an element. If called on a set of elements, parent returns a set of their unique direct parent elements.
<b>12</b>	parents([selector])	Get a set of elements containing the unique ancestors of the matched set of elements (except for the root element).
<b>13</b>	prev([selector])	Get a set of elements containing the unique previous siblings of each of the matched set of elements.
<b>14</b>	prevAll([selector])	Find all sibling elements in front of the current element.

15	siblings([selector])	Get a set of elements containing all of the unique siblings of each of the matched set of elements.
----	----------------------	---

## jQuery - CSS Selectors Methods

The jQuery library supports nearly all of the selectors included in Cascading Style Sheet (CSS) specifications 1 through 3, as outlined on the World Wide Web Consortium's site.

Using JQuery library developers can enhance their websites without worrying about browsers and their versions as long as the browsers have JavaScript enabled. Most of the JQuery CSS Methods do not modify the content of the jQuery object and they are used to apply CSS properties on DOM elements.

### Apply CSS Properties

This is very simple to apply any CSS property using JQuery method `css( PropertyName, PropertyValue )`.

Here is the syntax for the method –

```
selector.css( PropertyName, PropertyValue );
```

Here you can pass `PropertyName` as a javascript string and based on its value, `PropertyValue` could be string or integer.

### Apply Multiple CSS Properties

You can apply multiple CSS properties using a single JQuery method `CSS( {key1:val1, key2:val2....} )`. You can apply as many properties as you like in a single call.

Here is the syntax for the method –

```
selector.css( {key1:val1, key2:val2....keyN:valN} )
```

Here you can pass `key` as property and `val` as its value as described above.

### Example

Following is an example which adds font color as well as background color to the second list item.

```
<html>
  <head>
```

```
<title>The jQuery Example</title>
<script type = "text/javascript"
       src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">
</script>

<script type = "text/javascript" language = "javascript">
$(document).ready(function() {
    $("li").eq(2).css({ "color":"red", "background-color":"green" });
});
</script>
</head>

<body>
<div>
<ul>
    <li>list item 1</li>
    <li>list item 2</li>
    <li>list item 3</li>
    <li>list item 4</li>
    <li>list item 5</li>
    <li>list item 6</li>
</ul>
</div>
</body>
</html>
```

This will produce following result-

- list item 1
- list item 2
- **list item 3**
- list item 4
- list item 5
- list item 6

## Setting Element Width & Height

The width( val ) and height( val ) method can be used to set the width and height respectively of any element.

Example- Following is a simple example which sets the width of first division element where as rest of the elements have width set by style sheet

```
<html>
  <head>
    <title>The jQuery Example</title>
    <script type = "text/javascript"
      src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">
    </script>

    <script type = "text/javascript" language = "javascript">
      $(document).ready(function() {
        $("div:first").width(100);
        $("div:first").css("background-color", "blue");
      });
    </script>

    <style>
      div {
        width:70px; height:50px; float:left;
        margin:5px; background:red; cursor:pointer;
      }
    </style>
  </head>

  <body>
    <div></div>
    <div>d</div>
    <div>d</div>
    <div>d</div>
    <div>d</div>
  </body>
</html>
```

## jQuery CSS Methods

Following table lists down all the methods which you can use to play with CSS properties –

S.No	Methods	Description
1	css(name)	Return a style property on the first matched element.

2	css(name, value)	Set a single style property to a value on all matched elements.
3	css(properties)	Set a key/value object as style properties to all matched elements.
4	height(val)	Set the CSS height of every matched element.
5	height()	Get the current computed, pixel, height of the first matched element.
6	innerHeight()	Gets the inner height (excludes the border and includes the padding) for the first matched element.
7	innerWidth()	Gets the inner width (excludes the border and includes the padding) for the first matched element.
8	offset()	Get the current offset of the first matched element, in pixels, relative to the document.
9	offsetParent()	Returns a jQuery collection with the positioned parent of the first matched element.
10	outerHeight([margin])	Gets the outer height (includes the border and padding by default) for the first matched element.
11	outerWidth([margin])	Get the outer width (includes the border and padding by default) for the first matched element.
12	position()	Gets the top and left position of an element relative to its offset parent.
13	scrollLeft([val])	When a value is passed in, the scroll left offset is set to that value on all matched elements.
14	scrollLeft()	Gets the scroll left offset of the first matched element.

<b>15</b>	scrollTop(val)	When a value is passed in, the scroll top offset is set to that value on all matched elements.
<b>16</b>	scrollTop()	Gets the scroll top offset of the first matched element.
<b>17</b>	width(val)	Set the CSS width of every matched element.
<b>18</b>	width()	Get the current computed, pixel, width of the first matched element.

## jQuery Effects

jQuery provides a trivially simple interface for doing various kind of amazing effects. jQuery methods allow us to quickly apply commonly used effects with a minimum configuration. This tutorial covers all the important jQuery methods to create visual effects.

### jQuery hide() and show()

With jQuery, you can hide and show HTML elements with the hide() and show() methods:

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("#hide").click(function(){
    $("p").hide();
  });
  $("#show").click(function(){
    $("p").show();
  });
});
</script>
</head>
<body>

<p>If you click on the "Hide" button, I will disappear.</p>
```

```
<button id="hide">Hide</button>
<button id="show">Show</button>

</body>
</html>
```

This will produce following result-

If you click on the "Hide" button, I will disappear.

## jQuery toggle()

You can also toggle between hiding and showing an element with the toggle() method.

Shown elements are hidden and hidden elements are shown:

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("p").toggle();
  });
});
</script>
</head>
<body>

<button>Toggle between hiding and showing the paragraphs</button>

<p>This is a paragraph with little content.</p>
<p>This is another small paragraph.</p>

</body>
</html>
```

## jQuery Fade

With jQuery you can fade elements in and out of visibility.

### jQuery Fading Methods

With jQuery you can fade an element in and out of visibility. jQuery has the following fade methods:

- fadeIn()
- fadeOut()
- fadeToggle()
- fadeTo()

#### **jQuery fadeIn() Method**

The jQuery fadeIn() method is used to fade in a hidden element.

Syntax:

```
$(selector).fadeIn(speed,callback);
```

The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds. The optional callback parameter is a function to be executed after the fading completes.

#### **jQuery fadeOut() Method**

The jQuery fadeOut() method is used to fade out a visible element.

Syntax:

```
$(selector).fadeOut(speed,callback);
```

The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds. The optional callback parameter is a function to be executed after the fading completes.

#### **jQuery fadeToggle() Method**

The jQuery fadeToggle() method toggles between the fadeIn() and fadeOut() methods.

If the elements are faded out, fadeToggle() will fade them in. If the elements are faded in, fadeToggle() will fade them out.

Syntax:

```
$(selector).fadeToggle(speed,callback);
```

The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds. The optional callback parameter is a function to be executed after the fading completes.

### **jQuery fadeTo() Method**

The jQuery fadeTo() method allows fading to a given opacity (value between 0 and 1).

Syntax:

```
$(selector).fadeTo(speed,opacity,callback);
```

The required speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds. The required opacity parameter in the fadeTo() method specifies fading to a given opacity (value between 0 and 1).

The optional callback parameter is a function to be executed after the function completes.

## jQuery slide

**jQuery Sliding Methods-** With jQuery you can create a sliding effect on elements. jQuery has the following slide methods:

- slideDown()
- slideUp()
- slideToggle()
- jQuery slideDown() Method

The jQuery slideDown() method is used to slide down an element.

Syntax:

```
$(selector).slideDown(speed,callback);
```

The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds. The optional callback parameter is a function to be executed after the sliding completes.

### **jQuery slideUp() Method**

The jQuery slideUp() method is used to slide up an element.

Syntax:

```
$(selector).slideUp(speed,callback);
```

The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds. The optional callback parameter is a function to be executed after the sliding completes.

### jQuery slideToggle() Method

The jQuery slideToggle() method toggles between the slideDown() and slideUp() methods. If the elements have been slid down, slideToggle() will slide them up.

If the elements have been slid up, slideToggle() will slide them down.

```
$(selector).slideToggle(speed,callback);
```

The optional speed parameter can take the following values: "slow", "fast", milliseconds. The optional callback parameter is a function to be executed after the sliding completes.

## jQuery Animations

The jQuery animate() method performs custom animation using element's style properties. The animate() method changes existing style properties to the specified properties with motion.

Specify a selector to get the reference of an element to which you want to add animation effect and then call animate() method with JSON object for style properties, speed of animation and other options.

```
$('selector expression').animate({ stylePropertyName : 'value'}, duration, easing, callback);
```

```
$('selector expression').animate({ propertyName : 'value'}, { options });
```

### Parameters

Parameters	Details
<b>properties</b>	An object of CSS properties and values that the animation will move toward
<b>duration</b>	(default: 400) A string or number determining how long the animation will run

<b>easing</b>	(default: swing) A string indicating which easing function to use for the Transition
<b>complete</b>	A function to call once the animation is complete, called once per matched element.
<b>start</b>	specifies a function to be executed when the animation begins.
<b>step</b>	specifies a function to be executed for each step in the animation.
<b>queue</b>	a Boolean value specifying whether or not to place the animation in the effects queue.
<b>progress</b>	specifies a function to be executed after each step in the animation.
<b>done</b>	specifies a function to be executed when the animation ends.
<b>fail</b>	specifies a function to be executed if the animation fails to complete.
<b>specialEasing</b>	a map of one or more CSS properties from the styles parameter, and their corresponding easing functions.
<b>always</b>	specifies a function to be executed if the animation stops without completing.

## Animation with callback

Sometimes we need to change words position from one place to another or reduce size of the words and change the color of words automatically to improve the attraction of our website or web apps. JQuery helps a lot with this concept using fadeIn(), hide(), slideDown() but its functionality are limited and it only done the specific task which assign to it.

Jquery fix this problem by providing an amazing and flexible method called. `animate ()`. This method allows to set custom animations which is used css properties that give permission to fly over borders. for example, if we give css style property as `width:200;` and current position of the DOM element is 50, `animate` method reduces current position value from given css value and animate that element to 150.But we don't need to bother about this part because animation engine will handle it.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>
<script>
$("#btn1").click(function(){
  $("#box").animate({width: "200px"});
});
</script>
<button id="btn1">Animate Width</button>
<div id="box" style="background:#98bf21; height:100px; width:100px; margin:6px;"></div>
```

# DBMS

## Basic Concepts of DBMS

In this section, we will read about:

- Purpose of database systems
- Data abstraction
- Database Users
- Data Independence (Logical & Physical)
- Instance & Schemes
- Three layered Architecture of DBMS
- Different Levels of Abstraction.
- Data Modeling
- E-R Modeling
- Logical Model: Object & Record based – Object oriented model - Entity relationshipmodels
- Entity sets & relationships sets
- Concept of attributes and relationships
- Introduction to mapping constraints.
- Basic Concepts of ER Model in DBMS
- Introduction to DBMS
- Structure of DBMS
- Relational Models
- Introduction to Hierarchical Model and Network Model
- Introduction to RDBMS and Relational Models
- Introduction to relational algebra and relational calculus
- Understanding database technologies
- Relational Data Structure
- Keys and Relational Data Manipulation
- Relational Algebra
- Relational Algebraic Operations
- Set Operations
- Fundamental Operations
- Relational Calculus
- Data Definition Language
- Operators: select, project, join, rename etc.

## Purpose of database systems

### What is Database?

A **database** is an organized collection of data, so that it can be easily accessed and managed.

You can organize data into tables, rows, columns, and index it to make it easier to find relevant information.

**Database handlers** create a database in such a way that only one set of software program provides access of data to all the users.

The **main purpose** of the database is to operate a large amount of information by storing, retrieving, and managing data.

There are many **dynamic websites** on the World Wide Web nowadays which are handled through databases. For example, a model that checks the availability of rooms in a hotel. It is an example of a dynamic website that uses a database.

There are many **databases available** like MySQL, Sybase, Oracle, MongoDB, Informix, PostgreSQL, SQL Server, etc.

Modern databases are managed by the database management system (DBMS).

**SQL** or Structured Query Language is used to operate on the data stored in a database. SQL depends on relational algebra and tuple relational calculus.

A cylindrical structure is used to display the image of a database.



Image1: Database

Reference: <https://static.javatpoint.com/sqlpages/images/database.png>

## What is DBMS?

**Database Management System (DBMS)** refers to the technology solution used to optimize and manage the storage and retrieval of data from databases. DBMS offers a systematic approach to manage databases via an interface for users as well as workloads accessing the databases via apps. The management responsibilities for DBMS encompass information within the databases, the processes applied to databases (such as access and modification), and the database's logic structure. DBMS also facilitates additional administrative operations such as change management, disaster recovery, compliance, and performance monitoring, among others.

In order to facilitate these functions, DBMS has the following key components:

**Software.** DBMS is primarily a software system that can be considered as a management console or an interface to interact with and manage databases. The interfacing also spreads across real-world physical systems that contribute data to the backend databases. The OS, networking software, and the hardware infrastructure is involved in creating, accessing, managing, and processing the databases.

**Data.** DBMS contains operational data, access to database records and metadata as a resource to perform the necessary functionality. The data may include files with such as index files, administrative information, and data dictionaries used to represent data flows, ownership, structure, and relationships to other records or objects.

**Procedures.** While not a part of the DBMS software, procedures can be considered as instructions on using DBMS. The documented guidelines assist users in designing, modifying, managing, and processing databases.

**Database languages.** These are components of the DBMS used to access, modify, store, and retrieve data items from databases; specify database schema; control user access; and perform other associated database management operations. Types of DBMS languages include Data Definition Language (DDL), Data Manipulation Language (DML), Database Access Language (DAL) and Data Control Language (DCL).

**Query processor.** As a fundamental component of the DBMS, the query processor acts as an intermediary between users and the DBMS data engine in order to communicate query requests. When users enter an instruction in SQL language, the command is executed from the high-level language instruction to a low-level language that the underlying machine can understand and process to perform the appropriate DBMS functionality. In addition to instruction parsing and translation, the query processor also optimizes queries to ensure fast processing and accurate results.

**Runtime database manager.** A centralized management component of DBMS that handles functionality associated with runtime data, which is commonly used for context-based database access. This component checks for user authorization to request the query; processes the approved queries; devises an optimal strategy for query execution; supports concurrency so that multiple users can simultaneously work on same databases; and ensures integrity of data recorded into the databases.

**Database manager.** Unlike the runtime database manager that handles queries and data at runtime, the database manager performs DBMS functionality associated with the data within databases. Database manager allows a set of commands to perform different DBMS operations that include creating, deleting, backup, restoring, cloning, and other database maintenance tasks. The database manager may also be used to update the database with patches from vendors.

**Database engine.** This is the core software component within the DBMS solution that performs the core functions associated with data storage and retrieval. A database engine is also accessible via APIs that allow users or apps to create, read, write, and delete records in databases.

**Reporting.** The report generator extracts useful information from DBMS files and displays it in structured format based on defined specifications. This information may be used for further analysis, decision making, or business intelligence.

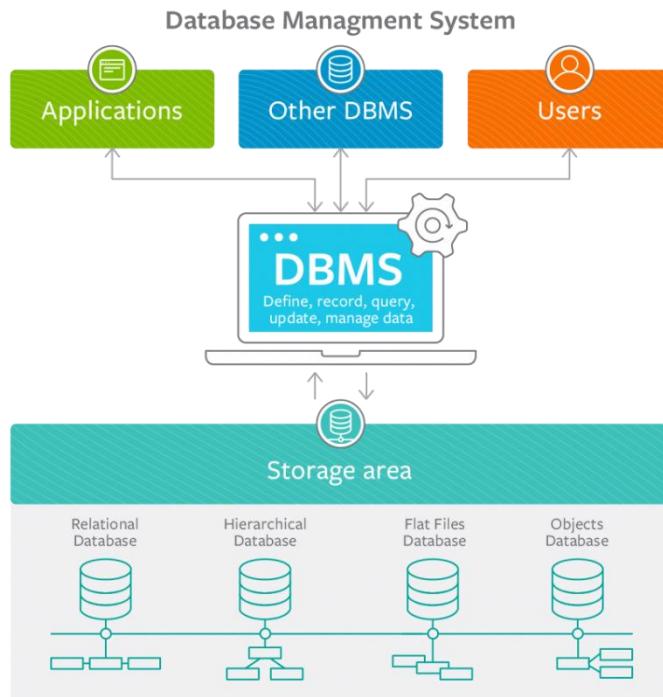


Image2: DBMS

Reference: <https://blogs.bmc.com/wp-content/uploads/2018/08/dbms-database-management-systems-810x898.png>

## What are the types of DBMS?

Depending upon the usage requirements, there are following types of databases available in the market:

- Centralized database.
- Distributed database.
- Personal database.
- End-user database.
- Commercial database.
- NoSQL database.
- Operational database.
- Relational database.
- Cloud database.
- Object-oriented database.
- Graph database.

### Centralized Database

The information(data) is stored at a centralized location and the users from different locations can access this data. This type of database contains application procedures that help the users to access the data even from a remote location.

Various kinds of authentication procedures are applied for the verification and validation of end users, likewise, a registration number is provided by the application procedures which keeps a track and record of data usage. The local area office handles this thing.

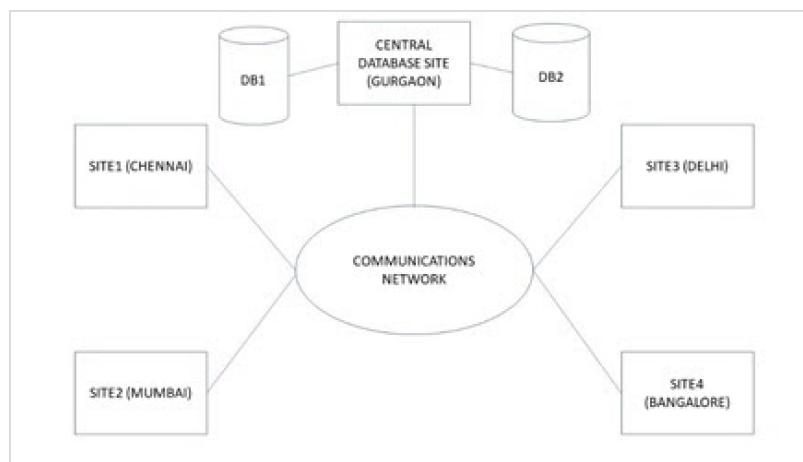


Image3: Centralized Database

Reference: <https://www.tutorialspoint.com/assets/questions/images/109302-1532341956.jpg>

## Distributed Database

Just opposite of the centralized database concept, the distributed database has contributions from the common database as well as the information captured by local computers also. The data is not at one place and is distributed at various sites of an organization. These sites are connected to each other with the help of communication links which helps them to access the distributed data easily.

You can imagine a distributed database as a one in which various portions of a database are stored in multiple different locations(physical) along with the application procedures which are replicated and distributed among various points in a network.

There are two kinds of distributed database, viz. homogenous and heterogeneous. The databases which have same underlying hardware and run over same operating systems and application procedures are known as homogeneous DDB, for eg. All physical locations in a DDB. Whereas, the operating systems, underlying hardware as well as application procedures can be different at various sites of a DDB which is known as heterogeneous DDB.

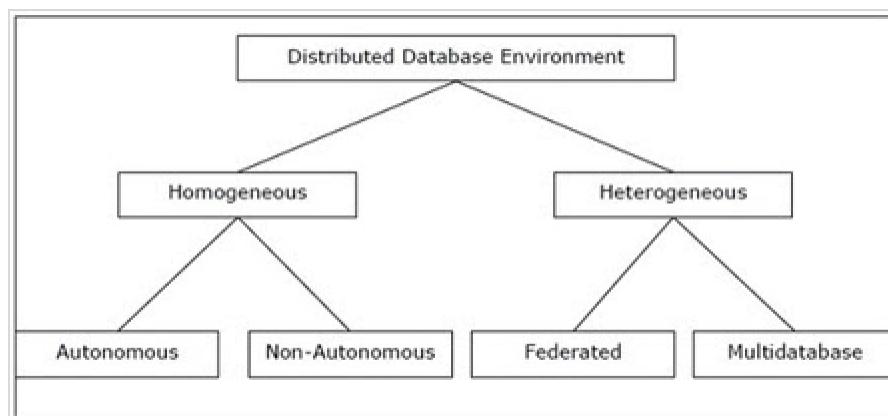


Image4: Distributed Database

Reference: <https://www.tutorialspoint.com/assets/questions/images/104606-1532341968.jpg>

## Personal Database

Data is collected and stored on personal computers which is small and easily manageable. The data is generally used by the same department of an organization and is accessed by a small group of people.

## End User Database

The end user is usually not concerned about the transaction or operations done at various levels and is only aware of the product which may be a software or an application. Therefore, this is a

shared database which is specifically designed for the end user, just like different levels' managers. Summary of whole information is collected in this database.

### Commercial Database

These are the paid versions of the huge databases designed uniquely for the users who want to access the information for help. These databases are subject specific, and one cannot afford to maintain such a huge information. Access to such databases is provided through commercial links.

### NoSQL Database

These are used for large sets of distributed data. There are some big data performance issues which are effectively handled by relational databases, such kind of issues are easily managed by NoSQL databases. There are very efficient in analyzing large size unstructured data that may be stored at multiple virtual servers of the cloud.

### Operational Database

Information related to operations of an enterprise is stored inside this database. Functional lines like marketing, employee relations, customer service etc. require such kind of databases.

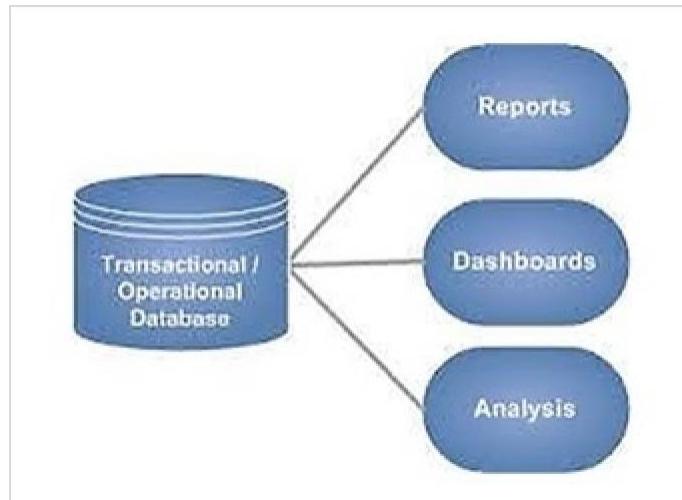


Image5: Operational Database

Reference: <https://www.tutorialspoint.com/assets/questions/images/104532-1532341980.jpg>

### Relational Databases

These databases are categorized by a set of tables where data gets fit into a pre-defined category. The table consists of rows and columns where the column has an entry for data for a specific

category and rows contains instance for that data defined according to the category. The Structured Query Language (SQL) is the standard user and application program interface for a relational database.

There are various simple operations that can be applied over the table which makes these databases easier to extend, join two databases with a common relation and modify all existing applications.

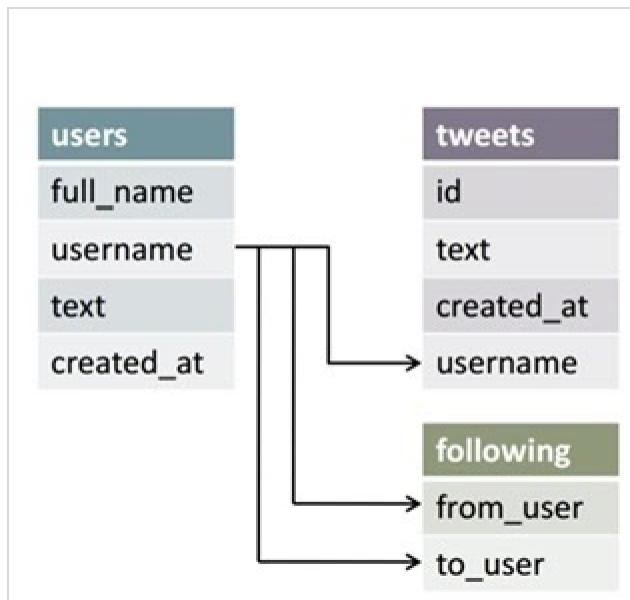


Image6: Relational Database

Reference: <https://www.tutorialspoint.com/assets/questions/images/112548-1532342000.jpg>

## Cloud Databases

Now a day, data has been specifically getting stored over clouds also known as a virtual environment, either in a hybrid cloud, public or private cloud. A cloud database is a database that has been optimized or built for such a virtualized environment. There are various benefits of a cloud database, some of which are the ability to pay for storage capacity and bandwidth on a per-user basis, and they provide scalability on demand, along with high availability.

A cloud database also gives enterprises the opportunity to support business applications in a software-as-a-service deployment.

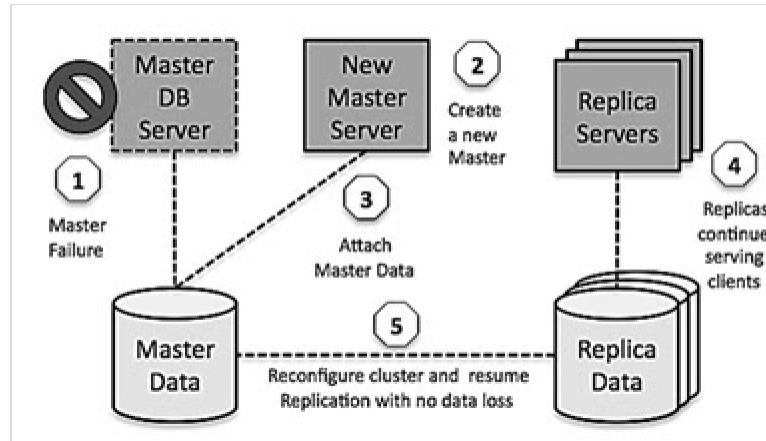


Image7: Cloud Database

Reference: <https://www.tutorialspoint.com/assets/questions/images/112415-1532342015.jpg>

## Object-Oriented Databases

An object-oriented database is a collection of object-oriented programming and relational database. There are various items which are created using object-oriented programming languages like C++, Java which can be stored in relational databases, but object-oriented databases are well-suited for those items.

An object-oriented database is organized around objects rather than actions, and data rather than logic. For example, a multimedia record in a relational database can be a definable data object, as opposed to an alphanumeric value.

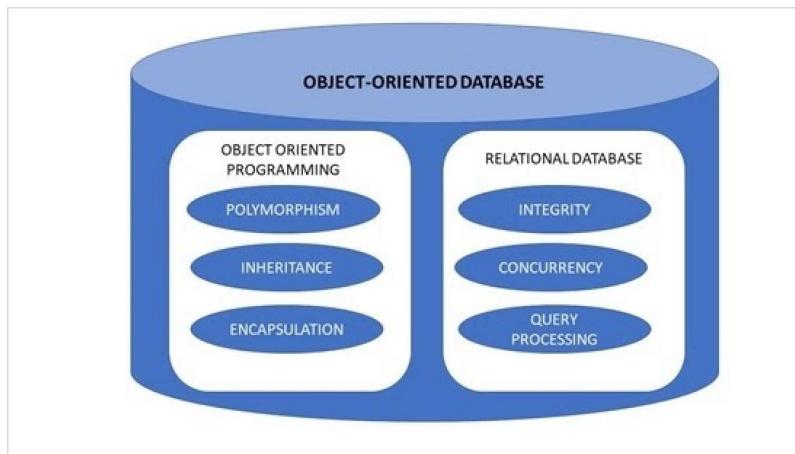


Image8: Object OrientedDatabase

Reference: <https://www.tutorialspoint.com/assets/questions/images/113587-1532342027.jpg>

## Graph Databases

The graph is a collection of nodes and edges where each node is used to represent an entity and each edge describes the relationship between entities. A graph-oriented database, or graph database, is a type of NoSQL database that uses graph theory to store, map and query relationships.

Graph databases are basically used for analyzing interconnections. For example, companies might use a graph database to mine data about customers from social media.

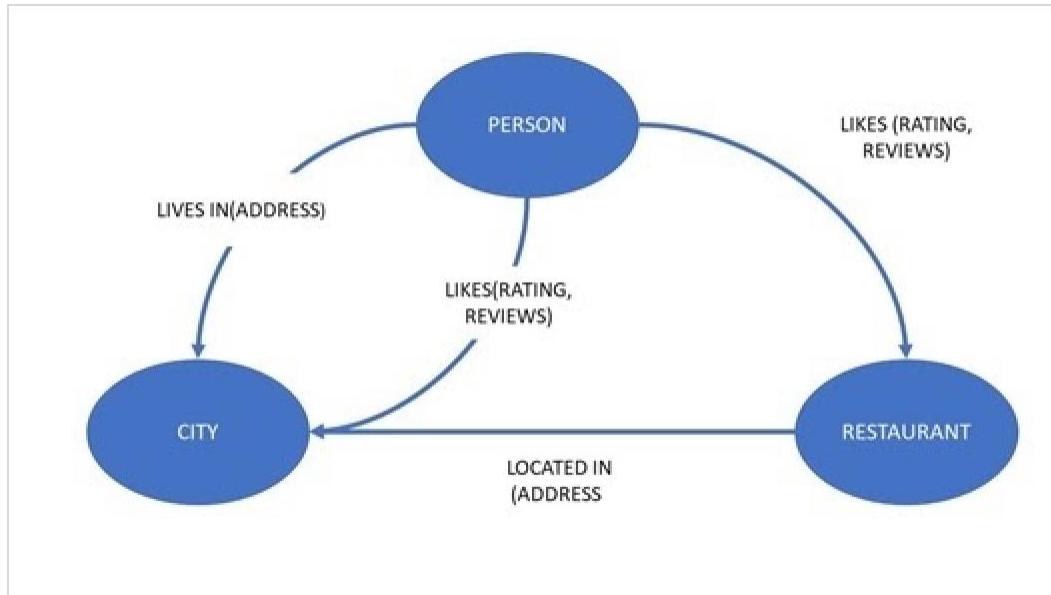


Image9: Graph Database

Reference: <https://www.tutorialspoint.com/assets/questions/images/114315-1532342037.jpg>

## DBMS vs. File System

There are following differences between DBMS and File system:

DBMS	File System
DBMS is a collection of data. In DBMS, the user is not required to write the procedures.	File system is a collection of data. In this system, the user has to write the procedures for managing the database.
DBMS gives an abstract view of data that hides the details.	File system provides the detail of the data representation and storage of data.
DBMS provides a crash recovery mechanism, i.e., DBMS protects the user from the system failure.	File system doesn't have a crash mechanism, i.e., if the system crashes while entering some data, then the content of the file will be lost.
DBMS provides a good protection mechanism.	It is very difficult to protect a file under the file system.
DBMS contains a wide variety of	File system can't efficiently store and retrieve the data.

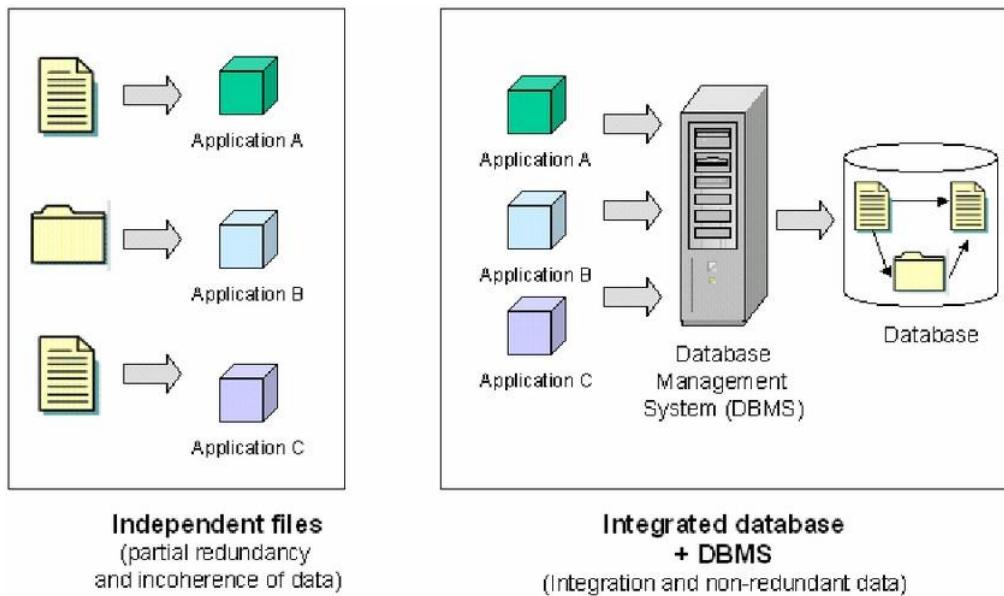


Image10: DBMS vs File System

Reference:

[https://www.researchgate.net/profile/Yvan\\_Bedard/publication/11041878/figure/fig1/AS:277384914849800@1443145\\_125825/independent-files- vs-integrated-database-DBMS.png](https://www.researchgate.net/profile/Yvan_Bedard/publication/11041878/figure/fig1/AS:277384914849800@1443145_125825/independent-files- vs-integrated-database-DBMS.png)

## Data Abstraction

It is one of the main and important characteristics of database approach. Data abstraction is the concept of hiding the details like data definition, data organization and storage of data from the end users and showing them only the essential things as per their requirement.

For example, an end user may be naïve user, application programmer, or an expert in DBMS. Their requirement in viewing data is different for different user. The users may not be interested in everything about a database. Since most of the end users of any database may not have enough idea about the implementation, the database developers have hidden many of the unwanted (for end users) information through several levels of data abstraction.

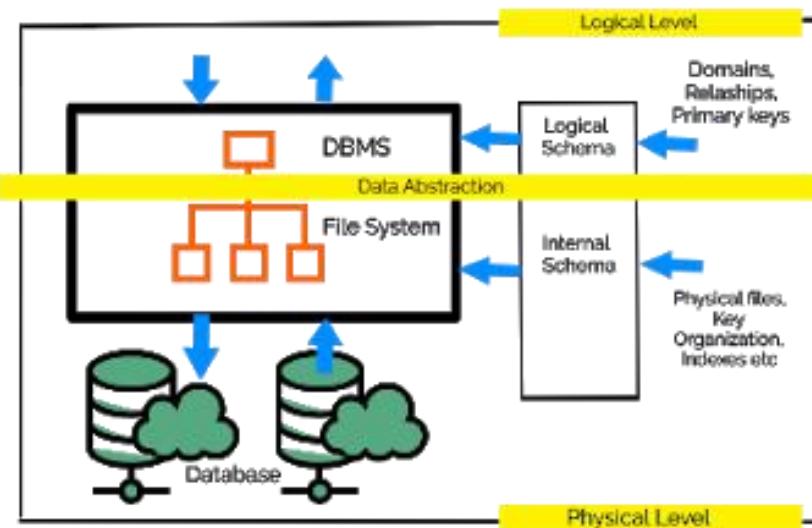


Image11: Data Abstraction

Reference:

[https://lh3.googleusercontent.com/proxy/SPbTFbsIwUqzpABk6nb\\_6eIt5QB3howul7kzRsiYJQ29r3JOoJMYGRtrHkZsRrmhe9osOTmHv77fBwl1knff7cExBkZHBSI7bi1c10wW5vv5NYA](https://lh3.googleusercontent.com/proxy/SPbTFbsIwUqzpABk6nb_6eIt5QB3howul7kzRsiYJQ29r3JOoJMYGRtrHkZsRrmhe9osOTmHv77fBwl1knff7cExBkZHBSI7bi1c10wW5vv5NYA)

## Database Users

**Database Users** are the one who interacts with the system. There can be four **different types of users** according to the way they interact with the system and for all the different users, different kind of user interfaces are designed as well. The four **types of users** are:

1. Naive users
2. Application Programmers
3. Sophisticated Users
4. DBA

Let us have a look over all the four users and their interfaces:

### Naive Users

Naive users are also termed as unsophisticated users and they interact with the system by calling anyone application program that has been written previously.

**For e.g. –** To transfer the program from one account to another, there is a need for an application program called transfer.

---

The user interface that is required for the naïve users is a forms interface, in which the user can fill the required fields.

Naive users can also easily read the reports that are generated from the database.

### **Application programmers**

Application programmers are the one who is responsible to write the application programs. They develop user interfaces through different tools. To construct the forms and the reports such that there is no need to write the program, there is a tool named Rapid Application Development (RAD).

Some special type of programming languages is also available such that includes vital control structures such as for loops, while loops and many others with the data manipulation language's statements. These special programming languages are termed as fourth-generation languages and they include the special features to provide the ability for the generation of the forms and to display the data on the screen.

In today's world, a large variety of commercial database systems includes these fourth generation languages.

### **Sophisticated users**

Sophisticated users aren't interested in writing programs and they interact with the system without writing any programs. Contrary, they use database query languages to interact with the system.

Sophisticated Users submit their queries to a query processor. Query Processor provides the facility to break the DML statements into instructions that can be understood by the storage manager.

Analysts are one among the sophisticated users. They use the tools to perform their task such as:

1. **Online analytical processing (OLAP)** - It helps the analysts to view them the summaries of the data in different ways.
2. **Data Mining Tools** – It helps the analysts find a certain kind of pattern in the given data.

### **DBA [Database Administrators]**

- In any organization where many people use the same resources, there is

a need for a chief administrator to oversee and manage these resources.

- In a database environment, the primary resource is the database itself, and the secondary resource is the DBMS and related software.
- Administering these resources is the responsibility of the database administrator (DBA).
- The DBA is responsible for authorizing access to the database, coordinating and monitoring its use, and acquiring software and hardware resources as needed.
- The DBA is accountable for problems such as security breaches and poor system response time. In large organizations, the DBA is assisted by a staff that carries out these functions.

### **Data Independence (Logical & Physical)**

- Data independence can be explained using the three-schema architecture.
- Data independence refers to the characteristic of being able to modify the schema at one level of the database system without altering the schema at the next higher level.

### **Types of Data Independence**

In DBMS there are two types of data independence

- Physical data independence
- Logical data independence.

### **Levels of Database**

Before we learn Data Independence, a refresher on Database Levels is important. The database has 3 levels as shown in the diagram below

1. Physical/Internal
2. Conceptual
3. External

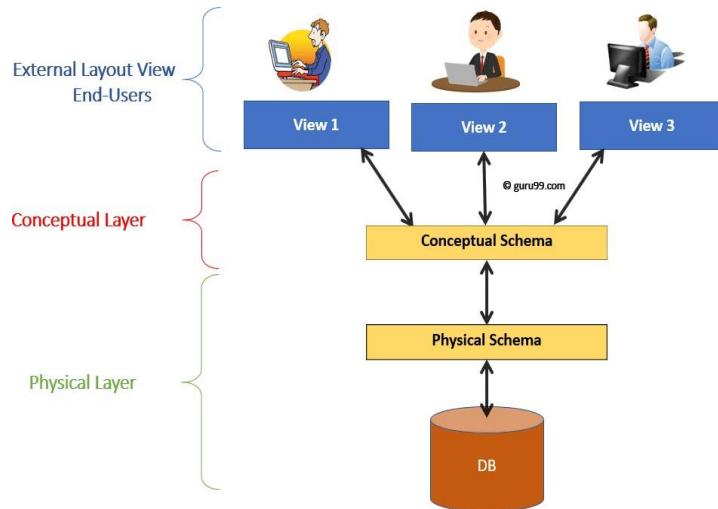


Image12: Levels of DBMS Architecture Diagram

Reference: [https://www.guru99.com/images/1/042919\\_0417\\_DataIndepend1.png](https://www.guru99.com/images/1/042919_0417_DataIndepend1.png)

Consider an Example of a University Database. At the different levels this is how the implementation will look like:

Type of Schema	Implementation
External Schema info(cid:int,cname:string)	<b>View 1:</b> Course <b>View 2:</b> studeninfo(id:int, name:string)
Conceptual Schema	Students(id: int, name: string, login: string, age:integer) Courses(id: int, cname:string, credits:integer) Enrolled(id: int, grade:string)
Physical Schema	<ul style="list-style-type: none"> <li>Relations stored as unordered files.</li> <li>Index on the first column of Students.</li> </ul>

## Physical Data Independence

Physical data independence helps you to separate conceptual levels from the internal/physical levels. It allows you to provide a logical description of the database without the need to specify physical structures. Compared to Logical Independence, it is easy to achieve physical data independence.

With Physical independence, you can easily change the physical storage structures or devices with an effect on the conceptual schema. Any change done would be absorbed by the mapping between the conceptual and internal levels. Physical data independence is achieved by the presence of the internal level of the database and then the transformation from the conceptual level of the database to the internal level.

Examples of changes under Physical Data Independence

Due to Physical independence, any of the below changes will not affect the conceptual layer.

- Using a new storage device like Hard Drive or Magnetic Tapes
- Modifying the file organization technique in the Database
- Switching to different data structures.
- Changing the access method.
- Modifying indexes.
- Changes to compression techniques or hashing algorithms.
- Change of Location of Database from say C drive to D Drive

## Logical Data Independence

Logical Data Independence is the ability to change the conceptual scheme without changing

1. External views
2. External API or programs

Any change made will be absorbed by the mapping between external and conceptual levels.

When compared to Physical Data independence, it is challenging to achieve logical data independence.

## Examples of changes under Logical Data Independence

Due to Logical independence, any of the below change will not affect the external layer.

1. Add/Modify/Delete a new attribute, entity or relationship is possible without a rewrite of existing application programs
2. Merging two records into one
3. Breaking an existing record into two or more records

## Difference between Physical and Logical Data Independence

Logical Data Independence	Physical Data Independence
Logical Data Independence is mainly concerned with the structure or changing the data definition.	Mainly concerned with the storage of the data.
It is difficult as the retrieving of data is mainly dependent on the logical structure of data.	It is easy to retrieve.

Compared to Logic Physical independence it is difficult to achieve logical data independence.	Compared to Logical Independence it is easy to achieve physical data independence.
You need to make changes in the Application program if new fields are added or deleted from the database.	A change in the physical level usually does not need change at the Application program level.
Modification at the logical levels is significant whenever the logical structures of the database are changed.	Modifications made at the internal levels may or may not be needed to improve the performance of the structure.
Concerned with conceptual schema	Concerned with internal schema
Example: Add/Modify/Delete a new attribute techniques, hashing	Example: change in compression algorithms, storage devices, etc

## Importance of Data Independence

- Helps you to improve the quality of the data
- Database system maintenance becomes affordable
- Enforcement of standards and improvement in database security
- You don't need to alter data structure in application programs
- Permit developers to focus on the general structure of the Database rather than worrying about the internal implementation
- It allows you to improve state which is undamaged or undivided
- Database incongruity is vastly reduced.
- Easily making modifications in the physical level is needed to improve the performance of the system.

## Instance & Schemes

- The data which is stored in the database at a particular moment of time is called an instance of the database.
- The overall design of a database is called schema.
- A database schema is the skeleton structure of the database. It represents the logical view of the entire database.
- A schema contains schema objects like table, foreign key, primary key, views, columns, data types, stored procedure, etc.
- A database schema can be represented by using the visual diagram. That diagram shows the database objects and relationship with each other.
- A database schema is designed by the database designers to help programmers whose software will interact with the database. The process of database creation is called data modeling.

A schema diagram can display only some aspects of a schema like the name of record type, data type, and constraints. Other aspects can't be specified through the schema diagram. For example, the given figure neither shows the data type of each data item nor the relationship among various files.

In the database, actual data changes quite frequently. For example, in the given figure, the database changes whenever we add a new grade or add a student. The data at a particular moment of time is called the instance of the database.

STUDENT			
Name	Student_number	Class	Major
COURSE			
Course_name	Course_number	Credit_hours	Department
PREREQUISITE			
Course_number	Prerequisite_number		
SECTION			
Section_identifier	Course_number	Semester	Year
GRADE_REPORT			
Student_number	Section_identifier	Grade	

Image13: Data Model Schema & Instance

Reference: <https://static.javatpoint.com/dbms/images/dbms-data-model-schema-and-instance.png>

## Differences Between Schema and Instance

1. A schema is the design representation of a database whereas instance is the snapshot of a database at a particular moment.
2. Instance changes very frequently, whenever data is removed or added in the database. As against, the changes in schema occurs rarely.
3. For example, schema and instance can be easily perceived by analogy to a program. At the time of writing a program in a programming language, the variables of that program is declared at first, this is analogous to the schema definition. Additionally, each variable in a program must have some values associated at a particular time; this is similar to an instance.

## Sub-Schema

A subschema lets the user have access to different areas of applications in which the user designed. The areas that are included in an application are set, types, record types, data items, and data aggregates. Schemas may have many different subschemas' that are all very different. There are several different reasons for subschema's. One reason for a subschema is to let a user or programmer see different views without having to see all the data contained in the database. Another reason would be to improve security so that someone who is not authorized can change or add to the data.

Data definition language describes the database in which there are possibly many programs that have been writing in different program languages. The description is in the form of names and characteristics of data items, data aggregates, records, areas, and sets included in the database, and the relationships that exist and that have to be maintained between occurrences of elements within the database.

A data item is represented by a value in a database, which is an occurrence of the smallest unit of the named data.

A Data aggregate is the occurrence of named collections of data items that are inside the record. There are two kinds, the first is a vector that is a one-dimensional sequence of data items. These all have identical characteristics. The other is a repeating group and is a collection of data that occurs several times within a record occurrence.

Record is an occurrence of a named collection of zero, one, or more items. Keys are a way to uniquely identify a record.

Set is an occurrence of a named collection of records. For every record in the schema there must be one occurrence.

Area is a named collection of records but it does not have to be linked between each record.

Database contains all the records within the schema; however each database must contain a separate schema.

### Three Layered of Architecture in DBMS

- The DBMS design depends upon its architecture. The basic client/server architecture is used to deal with a large number of PCs, web servers, database servers and other components that are connected with networks.
- The client/server architecture consists of many PCs and a workstation which are connected via the network.
- DBMS architecture depends upon how users are connected to the database to get their request done.

### Types of DBMS Architecture

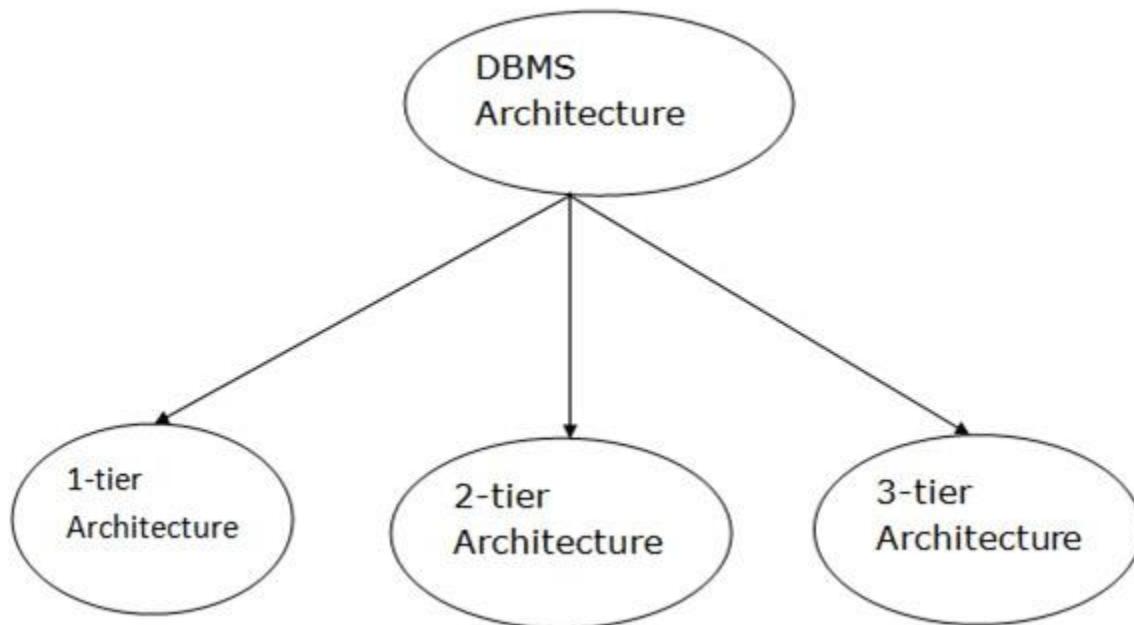


Image14: DBMS Architecture

Reference: <https://static.javatpoint.com/dbms/images/dbms-architecture.png>

Database architecture can be seen as a single tier or multi-tier. But logically, database architecture is of two types like: **2-tier architecture** and **3-tier architecture**.

### 1-Tier Architecture

- In this architecture, the database is directly available to the user. It means the user can directly sit on the DBMS and uses it.
- Any changes done here will directly be done on the database itself. It doesn't provide a handy tool for end users.
- The 1-Tier architecture is used for development of the local application, where programmers can directly communicate with the database for the quick response.

### 2-Tier Architecture

- The 2-Tier architecture is same as basic client-server. In the two-tier architecture, applications on the client end can directly communicate with the database at the server side. For this interaction, API's like: **ODBC**, **JDBC** are used.
- The user interfaces and application programs are run on the client-side.
- The server side is responsible to provide the functionalities like: query processing and transaction management.
- To communicate with the DBMS, client-side application establishes a connection with the server side.

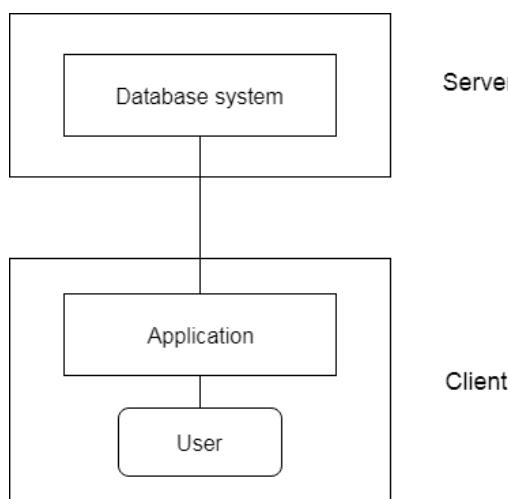


Image15: 2-Tier Architecture

Reference: <https://static.javatpoint.com/dbms/images/dbms-2-tier-architecture.png>

### 3-Tier Architecture

- The 3-Tier architecture contains another layer between the client and server. In this architecture, client can't directly communicate with the server.
- The application on the client-end interacts with an application server which further communicates with the database system.
- End user has no idea about the existence of the database beyond the application server. The database also has no idea about any other user beyond the application.
- The 3-Tier architecture is used in case of large web application.

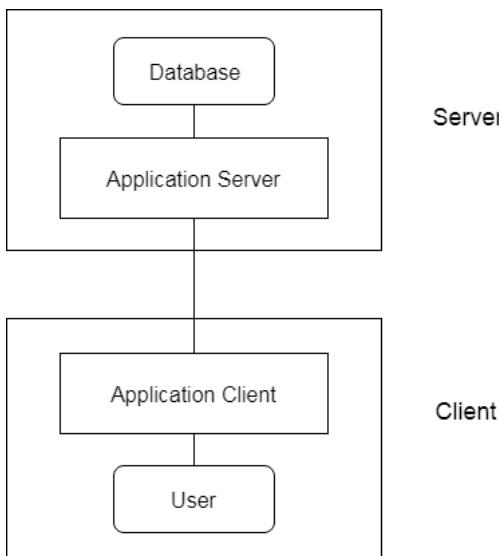


Image16: 3-Tier Architecture

Reference: <https://static.javatpoint.com/dbms/images/dbms-3-tier-architecture.png>

## Different Levels of Abstraction

- The three-schema architecture is also called ANSI/SPARC architecture or different-level of Abstraction.
- This framework is used to describe the structure of a specific database system.
- The three-schema architecture is also used to separate the user applications and physical database.
- The three-schema architecture contains three-levels. It breaks the database down into three different categories.

The level of Abstraction is as follows

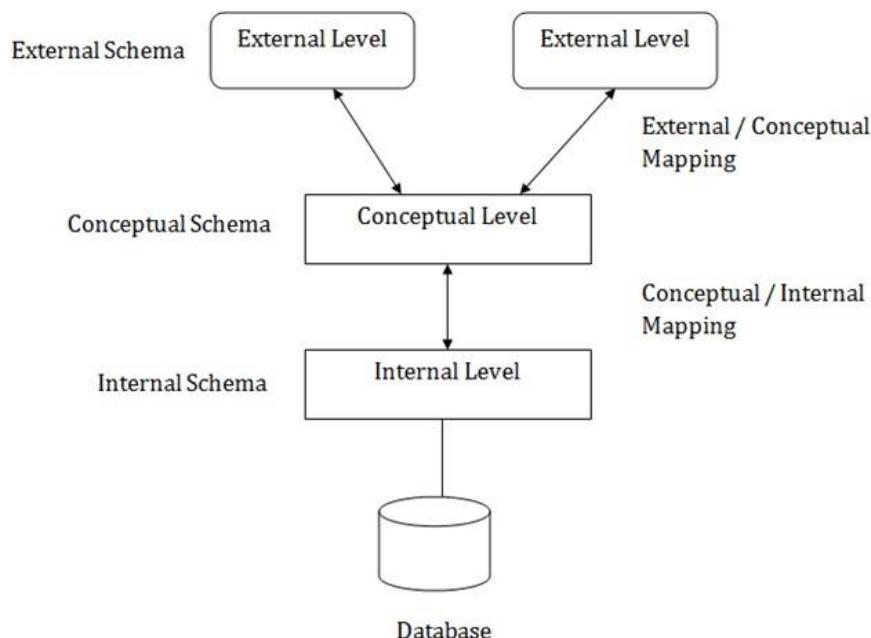


Image17: Level of Abstraction

Reference: <https://static.javatpoint.com/dbms/images/dbms-three-schema-architecture.png>

In the above diagram:

- It shows the DBMS architecture.
- Mapping is used to transform the request and response between various database levels of architecture.
- Mapping is not good for small DBMS because it takes more time.
- In External / Conceptual mapping, it is necessary to transform the

request from external level to conceptual schema.

- In Conceptual / Internal mapping, DBMS transforms the request from the conceptual to internal level.

### **Physical level or Internal Level**

- The internal level has an internal schema which describes the physical storage structure of the database.
- The internal schema is also known as a physical schema.
- It uses the physical data model. It is used to define that how the data will be stored in a block.
- The physical level is used to describe complex low-level data structures in detail.

### **Logical Level or Conceptual Level**

- The conceptual schema describes the design of a database at the conceptual level. Conceptual level is also known as logical level.
- The conceptual schema describes the structure of the whole database.
- The conceptual level describes what data are to be stored in the database and also describes what relationship exists among those data.
- In the conceptual level, internal details such as an implementation of the data structure are hidden.
- Programmers and database administrators work at this level.

### **View Level or External Level**

- At the external level, a database contains several schemas that are sometimes called subschemas. The subschema is used to describe the different view of the database.
- An external schema is also known as view schema.
- Each view schema describes the database part that a particular user group is interested in and hides the remaining database from that user group.
- The view schema describes the end user interaction with database systems.

## Data Modelling

Data Model is the modeling of the data description, data semantics, and consistency constraints of the data. It provides the conceptual tools for describing the design of a database at each level of data abstraction. Therefore, there are following four data models used for understanding the structure of the database:

### Relational Data Model

This type of model designs the data in the form of rows and columns within a table. Thus, a relational model uses tables for representing data and in-between relationships. Tables are also called relations. This model was initially described by Edgar F. Codd, in 1969. The relational data model is the widely used model which is primarily used by commercial data processing applications.

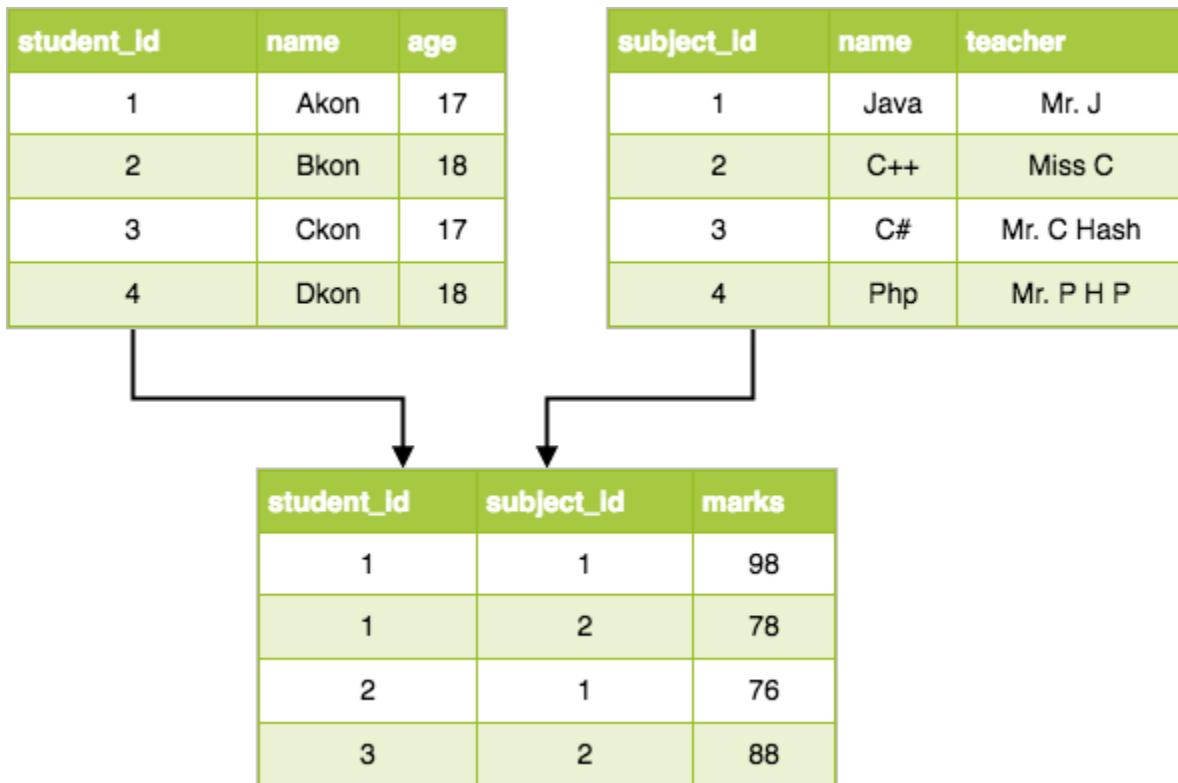


Image18: Relational Data Models

Reference: <https://www.studytonight.com/dbms/images/relational-dbms-model.png>

## Entity-Relationship Data Model

An ER model is the logical representation of data as objects and relationships among them. These objects are known as entities, and relationship is an association among these entities. This model was designed by Peter Chen and published in 1976 papers. It was widely used in database designing. A set of attributes describe the entities. For example, student\_name, student\_id describes the 'student' entity. A set of the same type of entities is known as an 'Entity set', and the set of the same type of relationships is known as 'relationship set'.

## Object-based Data Model

An extension of the ER model with notions of functions, encapsulation, and object identity, as well. This model supports a rich type system that includes structured and collection types. Thus, in 1980s, various database systems following the object-oriented approach were developed. Here, the objects are nothing but the data carrying its properties.

## Semistructured Data Model

This type of data model is different from the other three data models (explained above). The semistructured data model allows the data specifications at places where the individual data items of the same type may have different attributes sets. The Extensible Markup Language, also known as XML, is widely used for representing the semistructured data. Although XML was initially designed for including the markup information to the text document, it gains importance because of its application in the exchange of data.

## Hierarchical Model

This database model organises data into a tree-like-structure, with a single root, to which all the other data is linked. The hierarchy starts from the Root data, and expands like a tree, adding childnodes to the parent nodes.

In this model, a child node will only have a single parent node.

This model efficiently describes many real-world relationships like indexes of a book, recipes etc.

In hierarchical models, data is organised into a tree-like structure with one one-to-many relationship between two different types of data, for example, one department can have many courses, many professors and of-course many students.

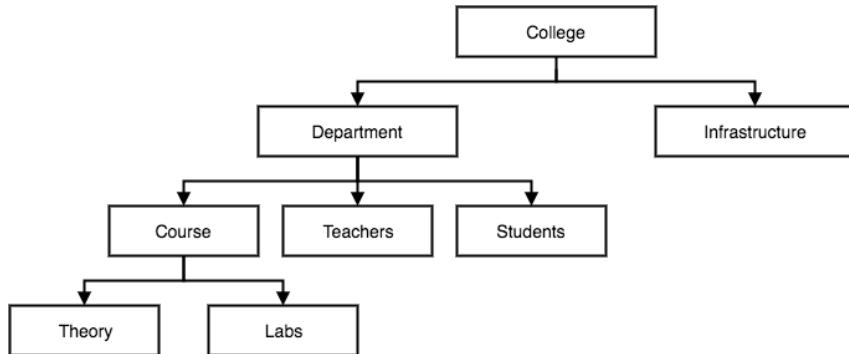


Image19: Hierarchical Model

Reference: <https://www.studytonight.com/dbms/images/hierarchical-dbms-model.png>

## Network Model

This is an extension of the Hierarchical model. In this model data is organised more like a graph, and are allowed to have more than one parent node.

In this database model data is more related as more relationships are established in this database model. Also, as the data is more related, hence accessing the data is also easier and faster. This database model was used to map many-to-many data relationships.

This was the most widely used database model, before the Relational Model was introduced.

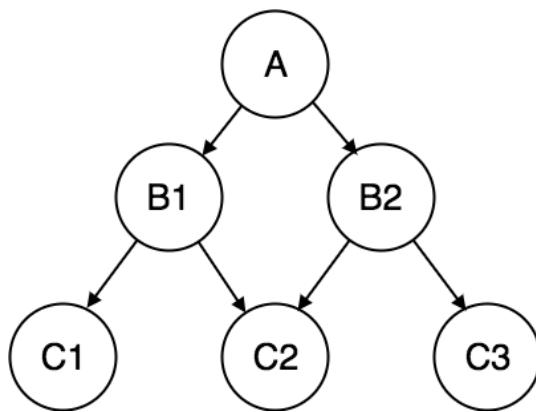


Image20: Network Model

Reference: <https://www.studytonight.com/dbms/images/network-dbms-model.png>

## Flat Data Model

Flat data model is the first and foremost introduced model and in this all the data used is kept in the same plane. Since it was used earlier this model was not so scientific.

Roll No	Name	Course
5482	Mark	Web Designing
5486	Steve	Java
5496	Smith	Oracle

Image21: Flat Data Model

Reference: <https://whatisdbms.com/wp-content/uploads/2016/06/Flat-Data-Model-in-DBMS.jpg>

## Record base Data Model

Record base model is used to specify the overall structure of the database and in this there are many record types. Each record type has fixed no. of fields having the fixed length.

## ER modeling

- ER model stands for an Entity-Relationship model. It is a high-level data model. This model is used to define the data elements and relationship for a specified system.
- It develops a conceptual design for the database. It also develops a very simple and easy to design view of data.
- In ER modeling, the database structure is portrayed as a diagram called an entity-relationship diagram.

**For example**, suppose we design a school database. In this database, the student will be an entity with attributes like address, name, id, age, etc. The address can be another entity with attributes like city, street name, pin code, etc and there will be a relationship between them.

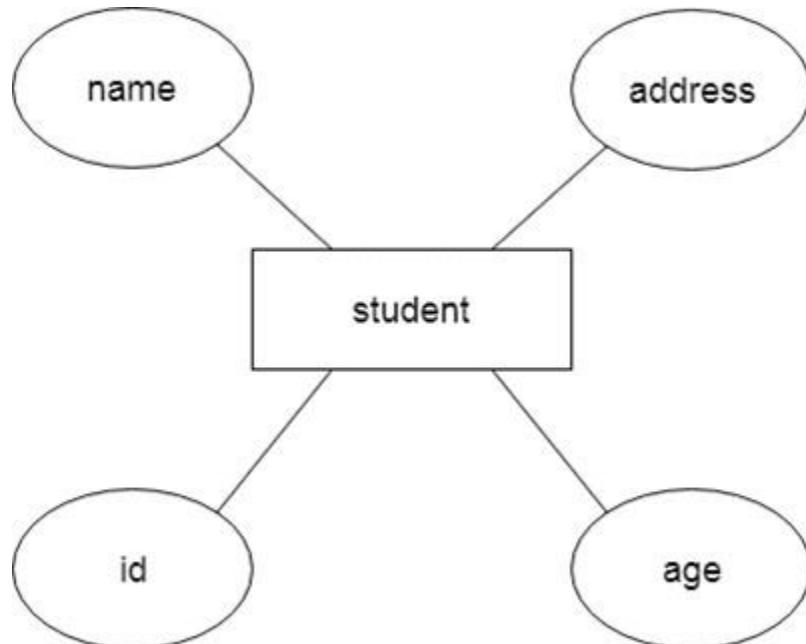


Image22: ER Model

Reference: <https://static.javatpoint.com/dbms/images/dbms-er-model-concept.png>

## Component of ER Diagram

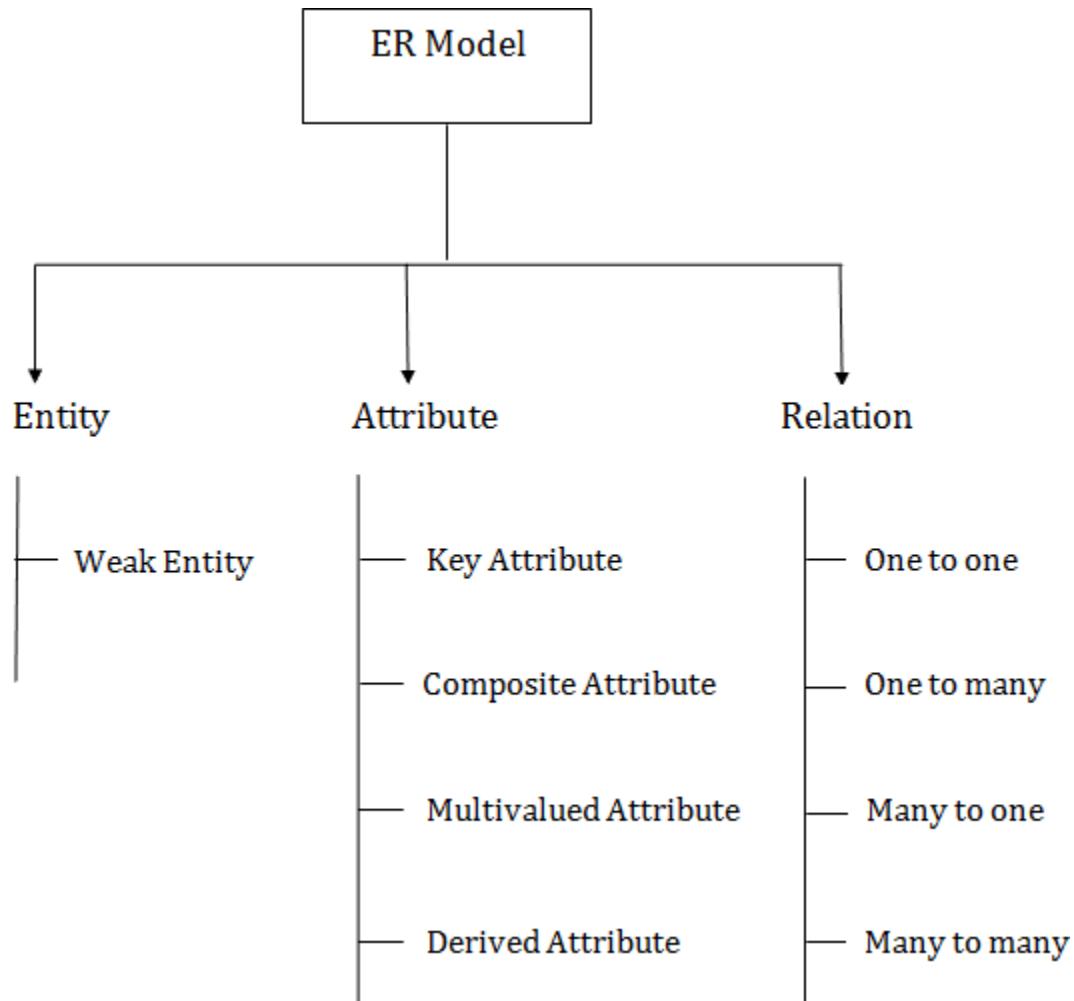


Image23: ER Model Concept Diagram

Reference: <https://static.javatpoint.com/dbms/images/dbms-er-model-concept-diagram.png>

## Logical Data Model

The logical data model is the one used most in designing BI applications. It builds upon the requirements provided by the business group. It includes a further level of detail, supporting both the business system-related and data requirements.

The business rules are appropriated into the logical data model, where they form relationships between the various data objects and entities. As opposed to a conceptual data model, which may have very general terms, the logical data model is the first step in designing and building out the architecture of the applications.

Like the conceptual data model, the logical data model is independent of specific database and data storage structures. It uses indexes and foreign keys to represent data relationships, but these are defined in a generic database context independent of any specific DBMS product.

The characteristics of the logical data model include:

- Features independent of specific database and data storage structures.
- Specific entities and attributes to be implemented.
- Identification of the business rules and relationships between those entities and attributes.
- Definitions of the primary keys, foreign keys, alternate keys, and inversion entities.

The logical model is used as a bridge from the application designer's view to the database design and the developer's specifications. This model should be used to validate whether the resulting applications that are built fulfill business and data requirements.

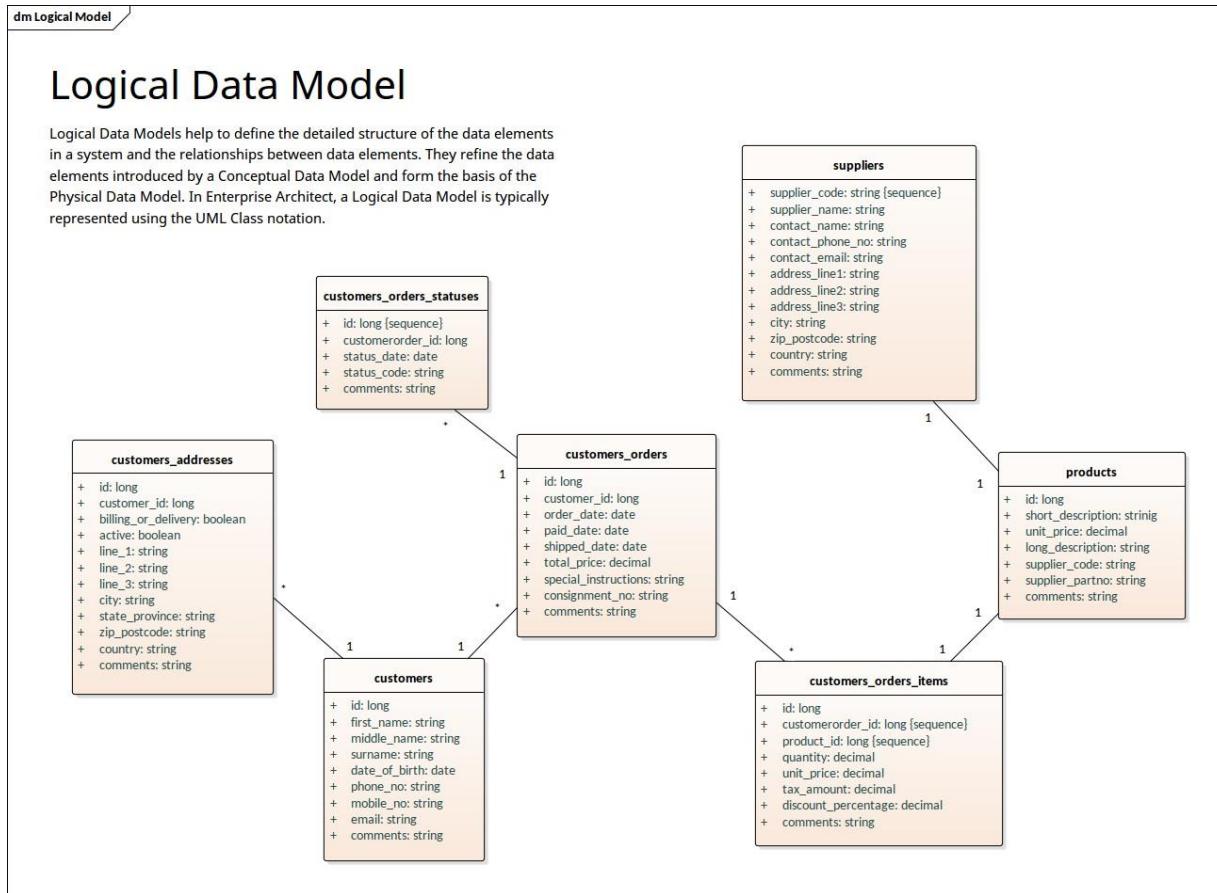


Image24: Logical Data Model Diagram

Reference: <https://sparxsystems.com/resources/gallery/diagrams/images/logical-data-model-uml-notation.png>

## Entity Sets

An entity may be any object, class, person or place. In the ER diagram, an entity can be represented as rectangles.

Consider an organization as an example- manager, product, employee, department etc. can be taken as an entity.

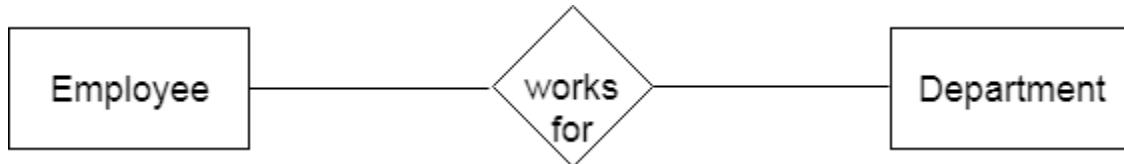


Image25: ER Model for Entity

Reference: <https://static.javatpoint.com/dbms/images/dbms-er-model-concept2.png>

## Definition of Strong Entity

The **Strong Entity** is the one whose existence does not depend on the existence of any other entity in a schema. It is denoted by a **single rectangle**. A strong entity always has the **primary key** in the set of attributes that describes the strong entity. It indicates that each entity in a strong entity set can be uniquely identified.

Set of similar types of strong entities together forms the **Strong Entity Set**. A strong entity holds the relationship with the weak entity via an **Identifying Relationship**, which is denoted by double diamond in the ER diagram. On the other hand, the relationship between two strong entities is denoted by a single diamond and it is simply called a **relationship**.

Let us understand this concept with the help of an example; a customer borrows a loan. Here we have two entities first a customer entity, and second a loan entity.

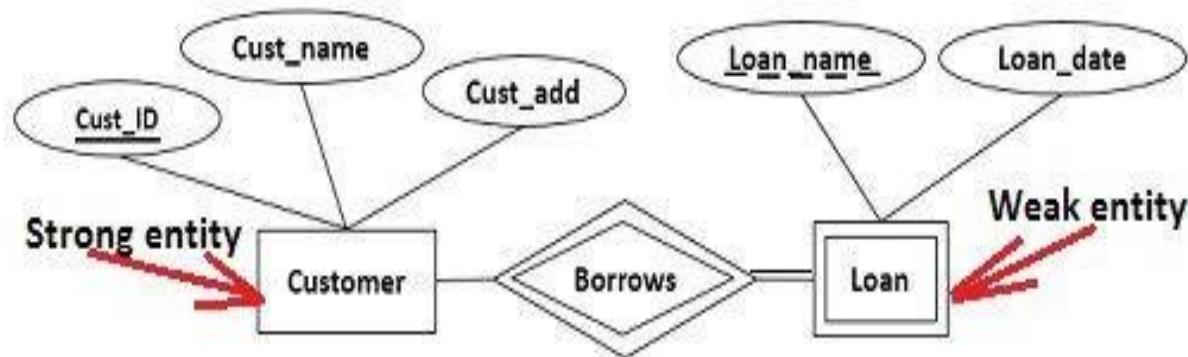


Image26: Strong &amp; Weak Entity Diagram

Reference: <https://techdifferences.com/wp-content/uploads/2016/12/Strong-entity-and-weak-entity.jpg>

Observing the ER-diagram above, for each loan, there should be at least one borrower otherwise that loan would not be listed in the Loan entity set. But even if a customer does not borrow any loan it would be listed in the Customer entity set. So, we can conclude that a customer entity does not depend on a loan entity.

<b>Cust_ID</b>	<b>Cust_name</b>	<b>Cust_add</b>
101	John	Xyzy
103	Ruby	Pqr
109	John	Uvfw

### Customer Entity Set

Image27: Customer EntitySet Diagram

Reference: [https://techdifferences.com/wp-content/uploads/2016/12/Strong\\_Entity-set.jpg](https://techdifferences.com/wp-content/uploads/2016/12/Strong_Entity-set.jpg)

The second thing you can observe is that the Customer entity has a primary key Cust\_ID which uniquely identifies each entity in Customer Entity set. This makes the Customer entity a strong entity on which a loan entity depends.

### Definition of Weak Entity

A **Weak entity** is the one that depends on its owner entity i.e. a strong entity for its existence. A weak entity is denoted by the **double rectangle**. Weak entities **do not** have the **primary key** instead it has a **partial key** that uniquely discriminates against the weak entities. The **primary key of a weak entity** is a composite key formed from the **primary key of the strong entity** and **partial key of the weak entity**.

The collection of similar weak entities is called **Weak Entity Set**. The relationship between a weak entity and a strong entity is always denoted with an **Identifying Relationship** i.e., **double diamond**.

For further illustration let us discuss the above example, this time from a weak entity's point of view. We have Loan as our weak entity, and as I said above for each loan there must be at least one borrower. You can observe in the loan entity set, no customer has borrowed a car loan and hence, it has totally vanished from the loan entity set. For the presence of a car loan in the loan entity set, it must have been borrowed by a customer. In this way, the weak Loan entity is dependent on the strong Customer entity.

Loan_name	Loan_date	Amount
Home	20/11/2015	20000
Education	5/10/2015	10000
Home	20/11/2015	20000

### Loan Entity Set

Image28: Loan EntitySet Diagram

Reference: [https://techdifferences.com/wp-content/uploads/2016/12/Weak\\_Entity\\_set.jpg](https://techdifferences.com/wp-content/uploads/2016/12/Weak_Entity_set.jpg)

The second thing, we know is a weak entity does not have a primary key. So here `Loan_name`, the partial key of the weak entity and `Cust_ID` primary key of the customer entity makes the primary key of the loan entity.

In the Loan entity set, we have two exactly same entities i.e. a **Home loan on date 20/11/2015 with amount 20000**. Now how to identify who had borrowed them this can be done with the help of the primary key of the weak entity (`Loan_name` + `Cust_ID`). So, it will be determined that one home loan is borrowed by Customer 101 Jhon and other by Customer 103 Ruby. This is how the composite primary key of weak entities identifies each entity in the weak entity set.

## Difference between Strong and Weak Entity

S.NO	STRONG ENTITY	WEAK ENTITY
1.	Strong entity always has a primary key.	While a weak entity has a partial discriminator key.
2.	Strong entity is not dependent on any other entity.	Weak entities depend on strong entities.
3.	Strong entity is represented by a single rectangle.	Weak entity is represented by a double rectangle.
4.	Two strong entity's relationships are represented by a single diamond.	While the relation between one strong and one weak entity is represented by a double diamond.
5.	Strong entities have either total participation or not.	While a weak entity always has total participation.

## Relationship Sets

A relationship is used to describe the relation between entities. Diamond or rhombus is used to represent the relationship.



Image29: Relationship

Reference: <https://static.javatpoint.com/dbms/images/dbms-er-model-concept9.png>

Types of relationship are as follows:

### One-to-One Relationship

When only one instance of an entity is associated with the relationship, then it is known as one-to-one relationship.

**For example,** A female can marry to one male, and a male can marry to one female.



Image30: One-to-One Relationship

Reference: <https://static.javatpoint.com/dbms/images/dbms-er-model-concept10.png>

### One-to-many relationship

When only one instance of the entity on the left, and more than one instance of an entity on the right associates with the relationship then this is known as a one-to-many relationship.

**For example,** scientists can invent many inventions, but the invention is done by only specific scientist.

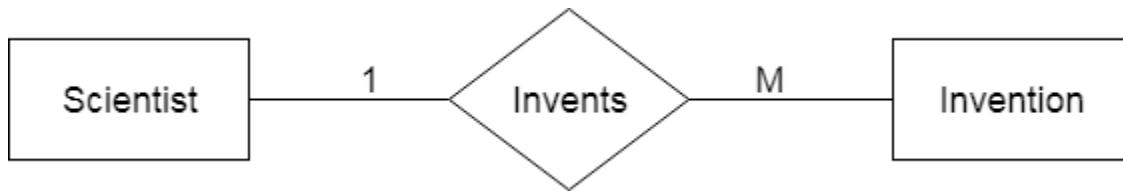


Image31: One-to-Many Relationship

Reference: <https://static.javatpoint.com/dbms/images/dbms-er-model-concept11.png>

### Many-to-one relationship

When more than one instance of the entity on the left, and only one instance of an entity on the right associates with the relationship then it is known as a many-to-one relationship.

**For example,** Student enrolls for only one course, but a course can have many students.

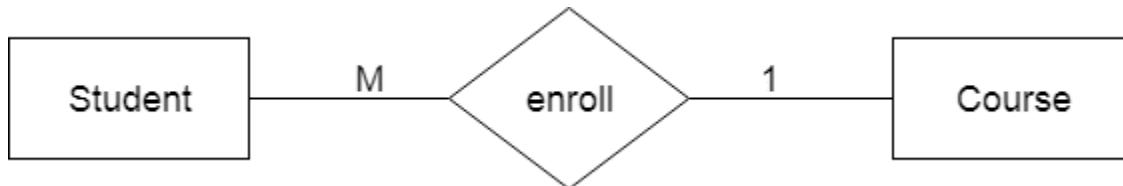


Image32: Many-to-one Relationship

Reference: <https://static.javatpoint.com/dbms/images/dbms-er-model-concept12.png>

### Many-to-many relationship

When more than one instance of the entity on the left, and more than one instance of an entity on the right associates with the relationship then it is known as a many-to-many relationship.

**For example,** employees can be assigned to many projects and projects can have many employees.



Image33: Many-to-Many Relationship

Reference: <https://static.javatpoint.com/dbms/images/dbms-er-model-concept13.png>

## Concept of Attributes

### Introduction

Attributes are the descriptive properties which are owned by each entity of an Entity Set.

There exists a specific domain or set of values for each attribute from where the attribute can take its values.

It is a single-valued property of either an entity-type or a relationship-type. For example, a lecture might have attributes: time, date, duration, place, etc. An attribute is represented by an Ellipse

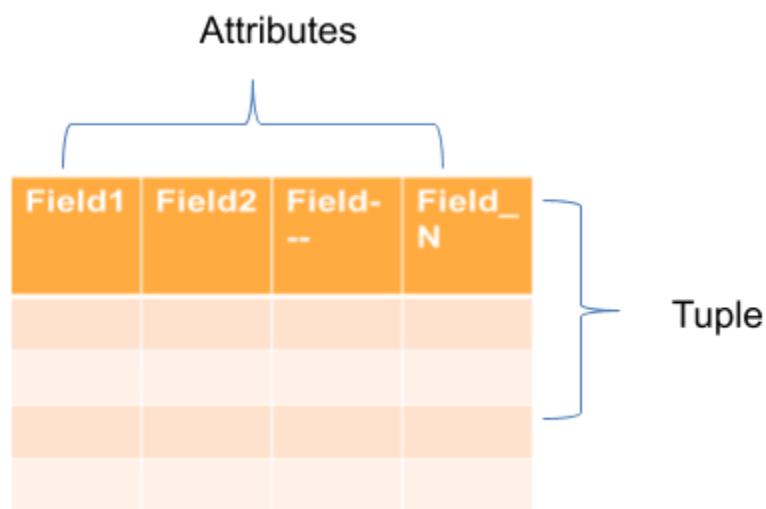
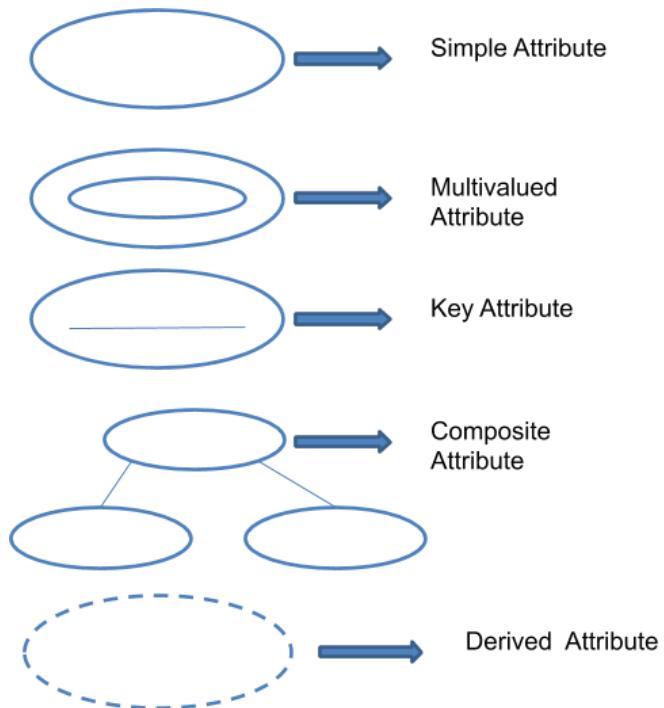


Image 1 – Attributes identification

## Symbols used to represent attributes



## Types of Attributes

- Simple Attributes
- Composite Attributes
- Single Valued Attributes
- Key Attributes
- Derived Attributes
- Multivalued Attributes

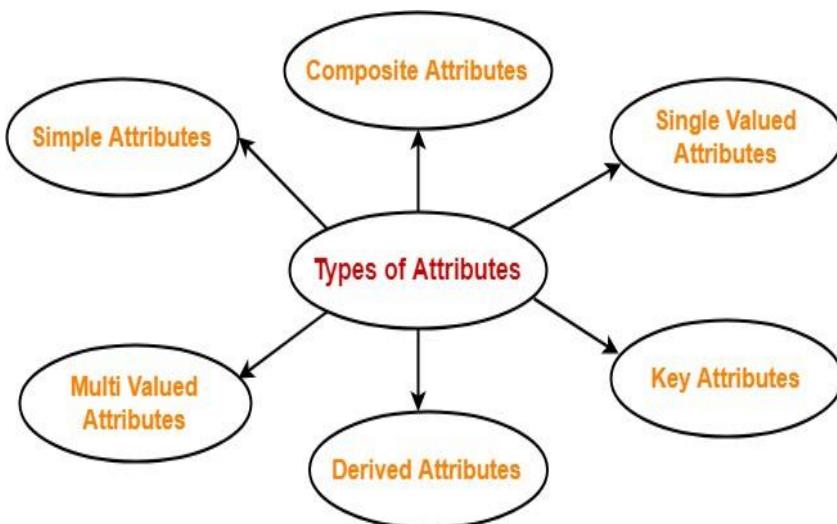


Image 2 – Type of Attributes

Reference - <https://www.gatevidyalay.com/wp-content/uploads/2018/06/Attributes-in-DBMS-Types.png>

## Simple Attributes

A simple attribute is identified entity from an entity set. Simple attribute represents oval symbol with its value.

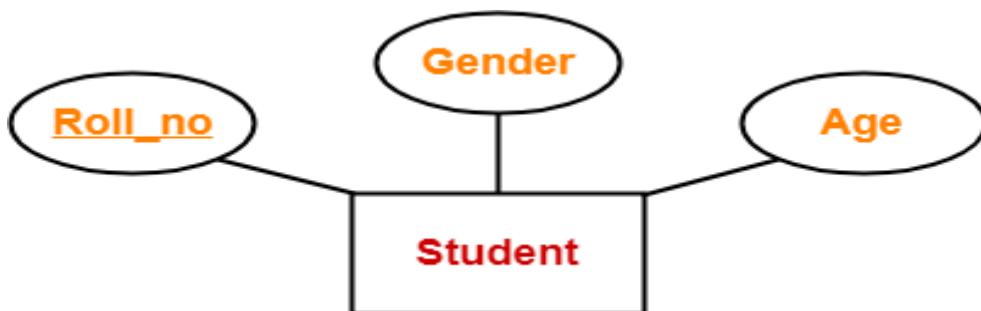


Image 3 – Simple Attribute

Reference - <https://www.gatevidyalay.com/wp-content/uploads/2018/06/Key-Attributes-Example.png>

Here, all the attributes are simple attributes as they cannot be divided further.

## Single Valued Attributes

Single valued attributes are those attributes which can take only one value for a given entity from an entity set.

## Key Attributes

A key attribute is uniquely identified entity from an entity set.

Key attribute represents oval symbol same as like other with underline.

## Derived Attributes

Derived attributes are those attributes which can be derived from other attribute(s).

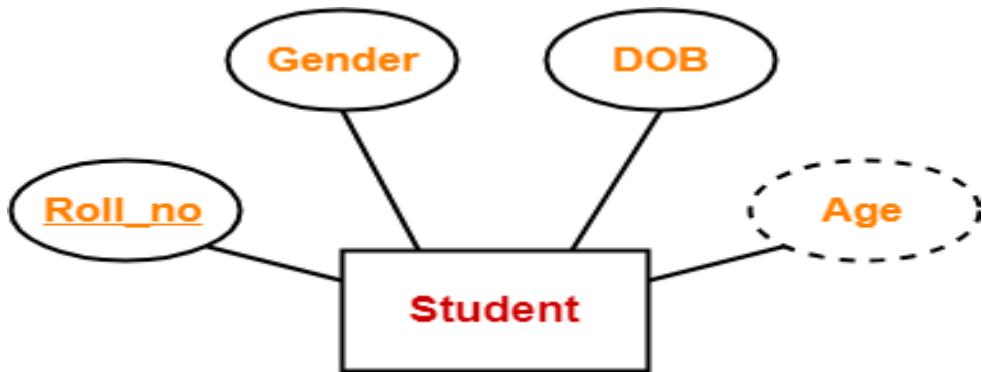


Image 4 – Key Attribute

Reference - <https://www.gatevidyalay.com/wp-content/uploads/2018/06/Derived-Attributes-Example.png>

Here, the attribute “Age” is a derived attribute as it can be derived from the attribute “DOB”.

## Multivalued Attributes

Multi valued attributes are those attributes which can take more than one value for a given entity from an entity set.

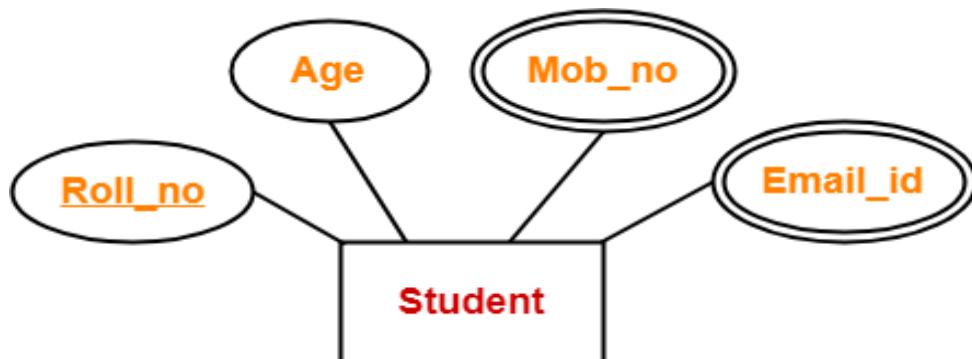


Image 5 – Multivalued Attribute

Reference - <https://www.gatevidyalay.com/wp-content/uploads/2018/06/Multi-Valued-Attributes-Example.png>

Here, the attributes “Mob\_no” and “Email\_id” are multi valued attributes as they can take morethan one values for a given entity.

### Example 1 – Student Book Relation

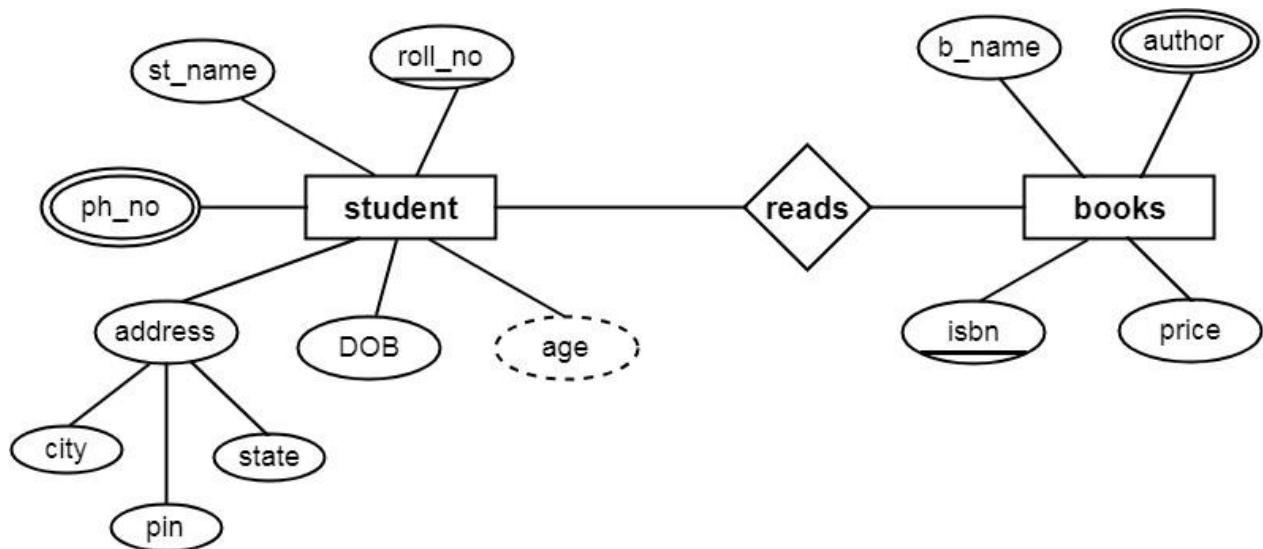


Image 6 – Student Book Relation

Reference - <https://www.csetutor.com/wp-content/uploads/2018/09/ER-Diagram-in-DBMS-Example.png>

## Example 2 – Product Order System

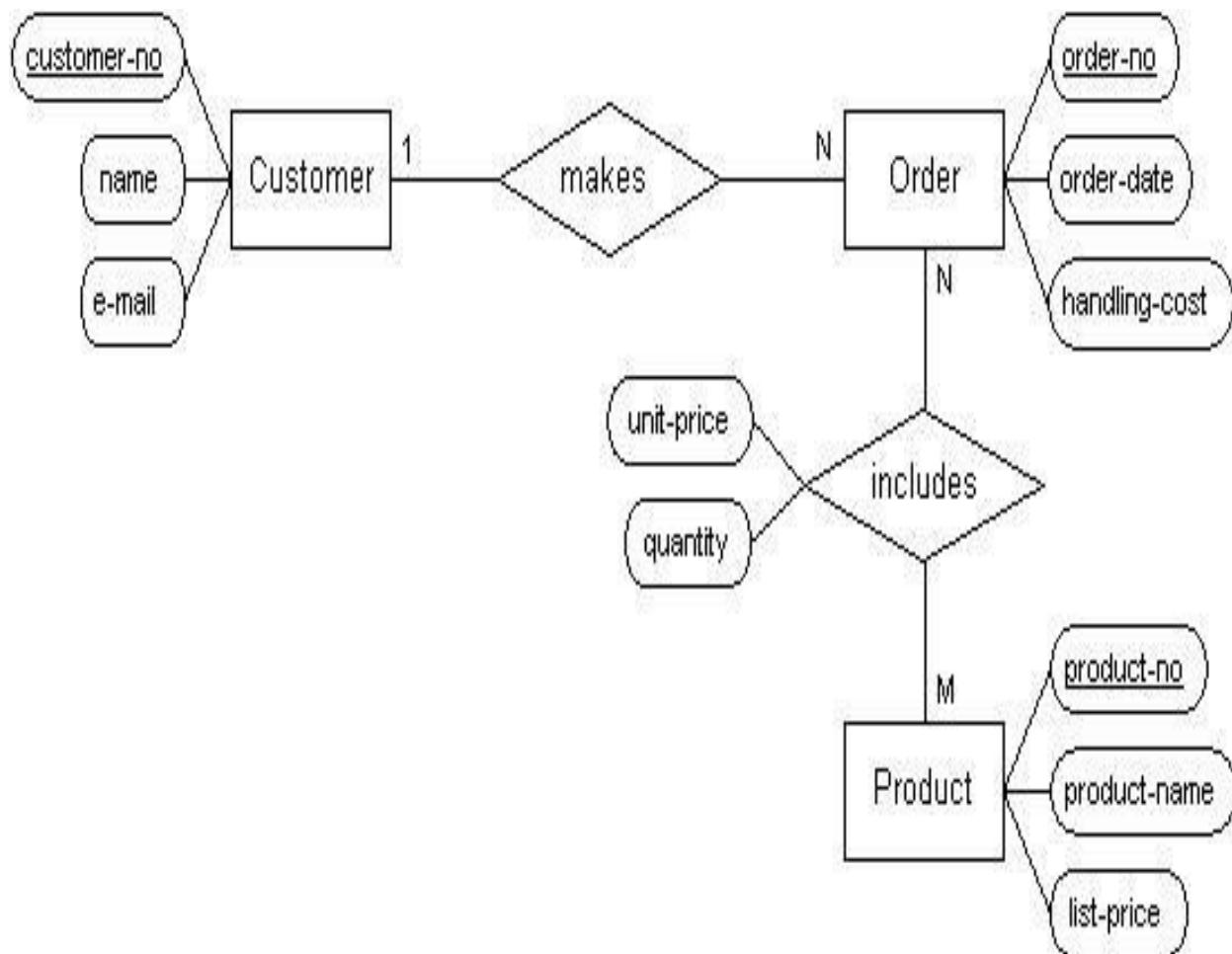


Image 7 – Product Order System

Reference - <https://codeandwork.github.io/courses/cs/media/erdiagram2.jpg>

## Example 3 –Employee Management

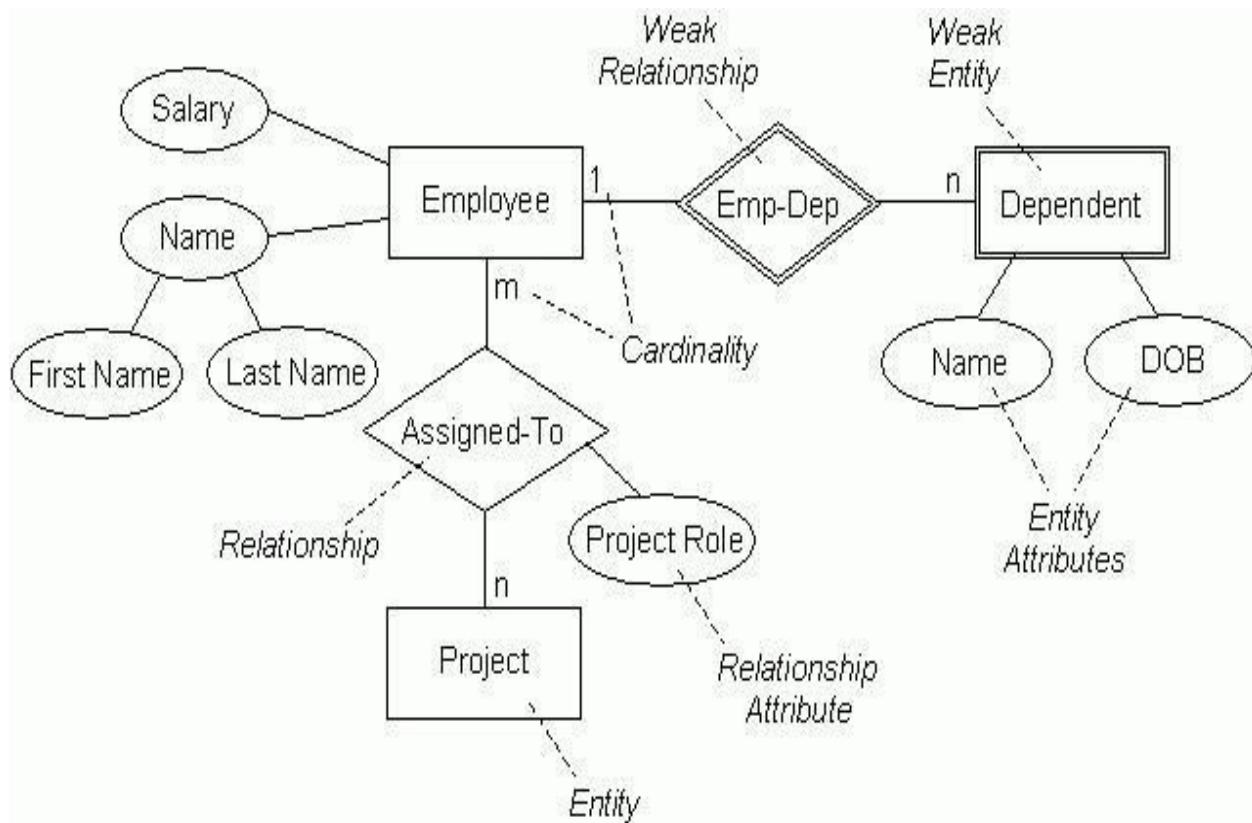


Image 8 – Employee Management

Reference - <https://codeandwork.github.io/courses/cs/media/erd-employee.jpg>

## Concept of Relationship

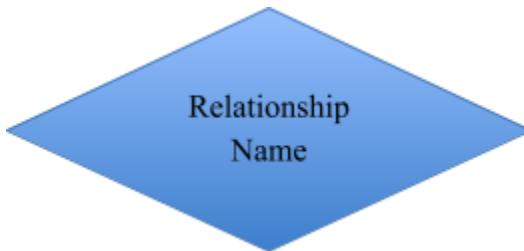
### Introduction

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. These entities can have attributes that define its properties. By defining the entities, their attributes, and showing the relationships between them, an ER diagram illustrates the logical structure of databases

Relationship is nothing but an association among two or more entities.

Entities take part in relationships. We can often identify relationships with verbs or verb phrases.

### Symbol



### Example

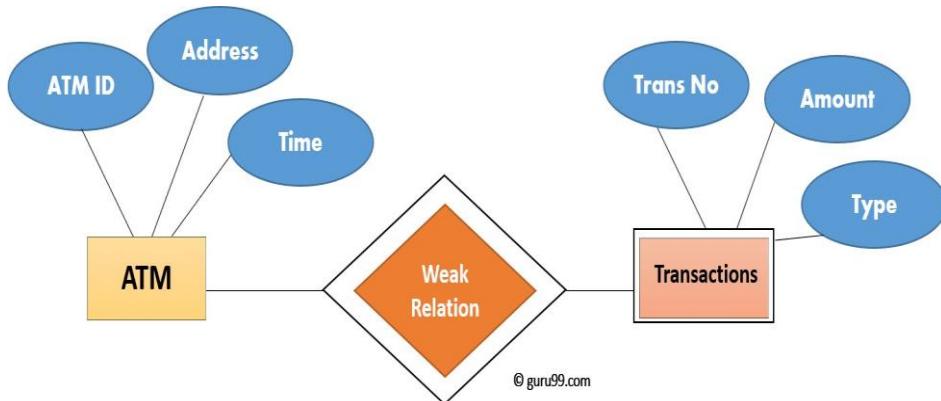


Image 9 – Transaction

Reference - [https://www.guru99.com/images/1/100518\\_0621\\_ERDiagramTu5.png](https://www.guru99.com/images/1/100518_0621_ERDiagramTu5.png)

## Types of Keys

### Keys

Keys play an important role in the relational database.

It is used to uniquely identify any record or row of data from the table. It is also used to establish and identify relationships between tables.

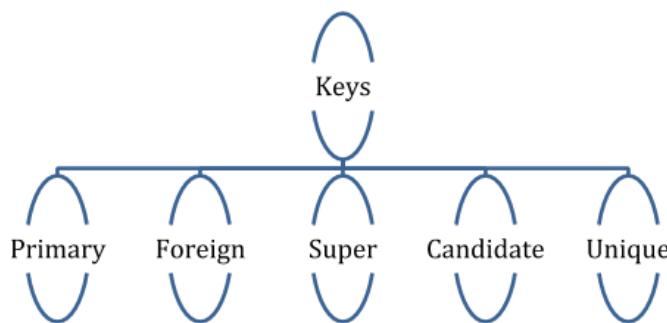


Image 10 – Transaction

### Primary Key

**PRIMARY KEY** It is the first key which is used to identify one and only one instance of an entity uniquely.

Rules Defining Primary Key –

- Two rows can't have the same primary key value
- It must for every row to have a primary key value.
- The primary key field cannot be null.
- The value in a primary key column can never be modified or updated if any foreign key refers to that primary key.
- There can be more than one candidate key in relation out of which one can be chosen as the primary key.

For Example, STUD\_NO, as well as STUD\_PHONE both, are candidate keys for relation STUDENT but STUD\_NO can be chosen as the primary key (only one out of many candidate keys).

**STUDENT**

STUD_NO	STUD_NAME	STUD_PHONE	STUD_STATE	STUD_COUNT RY	STUD_AGE
1	RAM	9716271721	Haryana	India	20
2	RAM	9898291281	Punjab	India	19
3	SUJIT	7898291981	Rajasthan	India	18
4	SURESH		Punjab	India	21

**Table 1**

**STUDENT\_COURSE**

STUD_NO	COURSE_NO	COURSE_NAME
1	C1	DBMS
2	C2	Computer Networks
1	C2	Computer Networks

**Table 2**

Image 11 – Student table

Reference - <https://media.geeksforgeeks.org/wp-content/uploads/image7.png>

## Foreign Key

FOREIGN KEY is a column that creates a relationship between two tables.

The purpose of Foreign keys is to maintain data integrity and allow navigation between two different instances of an entity.

Rules defining foreign key –

- Foreign key columns must use their referenced column's type.
- Each column cannot belong to more than 1 Foreign Key constraint.
- Cannot be a computed column.
- Foreign key columns must be indexed.

Based on Image 9 if an attribute can only take the values which are present as values of some other attribute, it will be a foreign key to the attribute to which it refers. The relation which is being

referenced is called referenced relation and the corresponding attribute is called referenced attribute and the relation which refers to the referenced relation is called referencing relation and the corresponding attribute is called referencing attribute. The referenced attribute of the referenced relation should be the primary key for it. For Example, STUD\_NO in STUDENT\_COURSE is a foreign key to STUD\_NO in STUDENT relation.

It may be worth noting that unlike, Primary Key of any given relation, Foreign Key can be NULL as well as may contain duplicate tuples i.e. it need not follow uniqueness constraint.

For Example, STUD\_NO in STUDENT\_COURSE relation is not unique. It has been repeated for the first and third tuple. However, the STUD\_NO in STUDENT relation is a primary key and it needs to be always unique and it cannot be null.

## Candidate Key

A candidate key is an attribute or set of an attribute which can uniquely identify a tuple.

The remaining attributes except for primary key are considered as a candidate key. The candidate keys are as strong as the primary key.

Rules defining candidate key –

- It must contain unique values
- Candidate key may have multiple attributes
- Must not contain null values
- It should contain minimum fields to ensure uniqueness
- Uniquely identify each record in a table

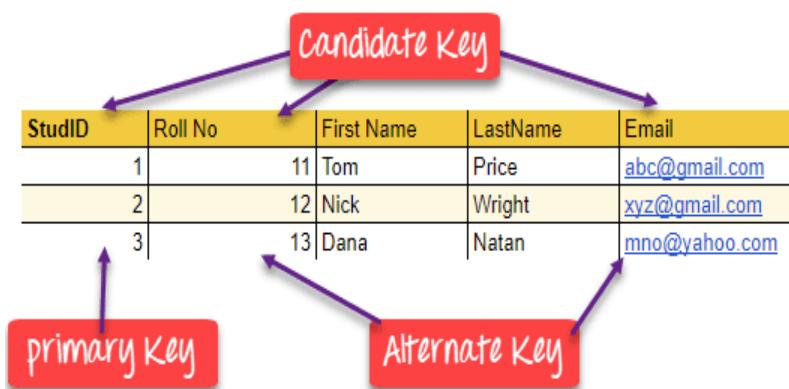


Image 12 – Candidate Key

Reference - [https://www.guru99.com/images/1/100518\\_0517\\_DBMSKeysPri1.png](https://www.guru99.com/images/1/100518_0517_DBMSKeysPri1.png)

## Super Key

A superkey is a group of single or multiple keys which identifies rows in a table.

A Super key may have additional attributes that are not needed for unique identification. In the above-given example, EmpNo and Emp\_Name are superkeys.

The set of attributes which can uniquely identify a tuple is known as Super Key. For Example, STUD\_NO, (STUD\_NO, STUD\_NAME) etc.

Adding zero or more attributes to candidate key generates super key. A candidate key is a super key but vice versa is not true.

## Compound Key

COMPOUND KEY has two or more attributes that allow you to uniquely recognize a specific record.

It is possible that each column may not be unique by itself within the database. Order ID and Product ID could be used as it uniquely identified each record.

In database design, a composite key is a candidate key that consists of two or more attributes (table columns) that together uniquely identify an entity occurrence (table row). A compound key is a composite key for which each attribute that makes up the key is a simple (foreign) key in its own right.

## Alternate Key

ALTERNATE KEYS is a column or group of columns in a table that uniquely identify every row in that table.

A table can have multiple choices for a primary key but only one can be set as the primary key. All the keys which are not primary key are called an Alternate Key.

The candidate key other than the primary key is called an alternate key. For Example, STUD\_NO, as well as STUD\_PHONE both, are candidate keys for relation STUDENT but STUD\_PHONE will be alternate key (only one out of many candidate keys).

## Unique Key

A unique key is a set of one or more than one fields/columns of a table that uniquely identify a record in a database table.

In database relational modeling and implementation, a unique key (also known as a candidate key or just a key) is a set of attributes (columns) within a relational database table (also called a relation), such that:

the table does not have two distinct rows or records with the same values for these columns;

this set of columns is minimal; i.e., removing any column from the key would result in duplicate values in the resulting subset.

When a column or set of columns is defined as unique to the database management system, the system verifies that each set of values is unique before assigning the constraint. After the columns are defined as unique, an error will occur if an insertion is attempted with values that already exist. Some systems will not allow key values to be updated, all systems will not allow duplicates. This ensures that uniqueness is maintained in both the primary table and any relations that are later bound to it.

## Mapping Constraints

### Introduction

A mapping constraint is a data constraint that expresses the number of entities to which another entity can be related via a relationship set.

Mapping constraints define how many entities can be related to another entity via a relationship.

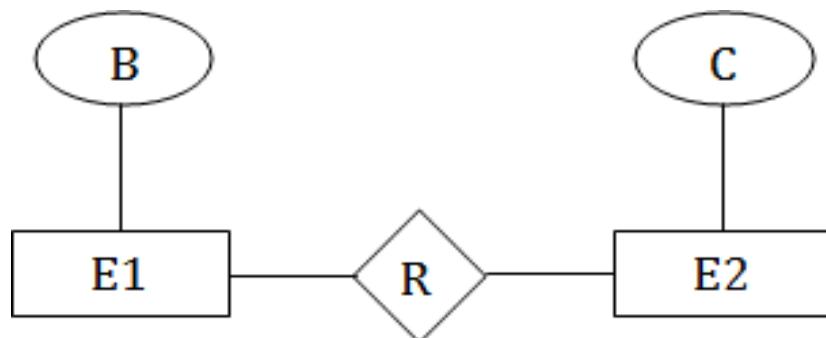


Image 13 – Mapping Constraint

Reference - <https://static.javatpoint.com/dbms/images/dbms-mapping-constraints4.png>

It is most useful in describing the relationship sets that involve more than two entity sets.

For binary relationship set R on an entity set A and B, there are four possible mapping cardinalities. These are as follows:

### Types of Mapping Constraints

- One to one (1:1)
- One to many (1:M)
- Many to one (M:1)
- Many to many (M:M)

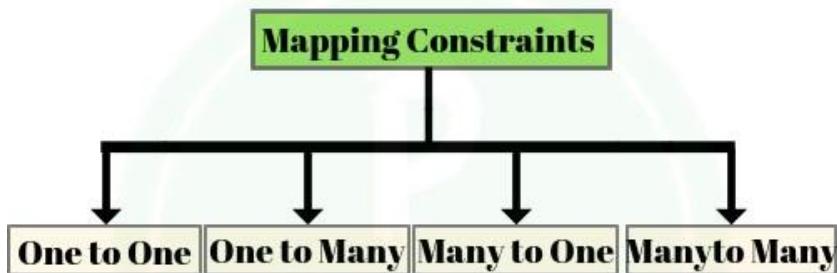


Image 14 – Mapping Constraint

Reference - <https://prepinsta.com/wp-content/uploads/2019/07/30-300x300.png>

### One-to-One

In one-to-one mapping, an entity in E1 is associated with at most one entity in E2, and an entity in E2 is associated with at most one entity in E1.

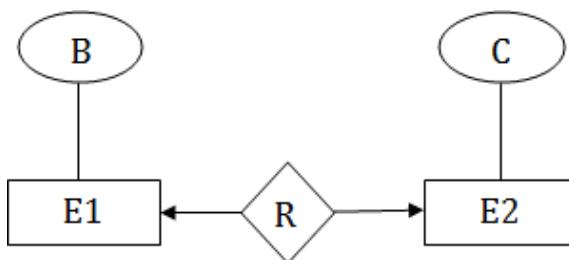


Image 15 – One to One

Reference - <https://static.javatpoint.com/dbms/images/dbms-mapping-constraints.png>

## One-to-many

In one-to-many mapping, an entity in E1 is associated with any number of entities in E2, and an entity in E2 is associated with at most one entity in E1.

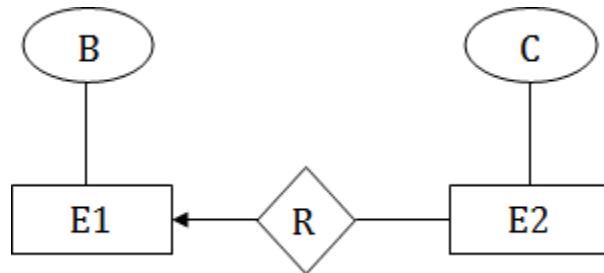


Image 16 – One to Many

Reference - <https://static.javatpoint.com/dbms/images/dbms-mapping-constraints2.png>

## Many-to One

In one-to-many mapping, an entity in E1 is associated with at most one entity in E2, and an entity in E2 is associated with any number of entities in E1.

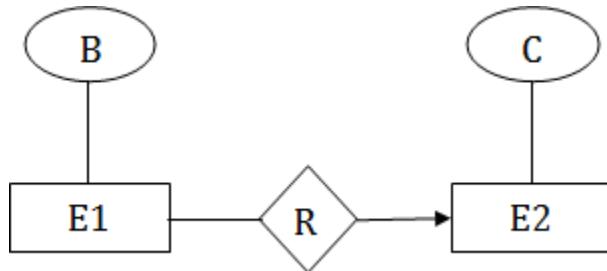


Image 17 – One to Many

Reference - <https://static.javatpoint.com/dbms/images/dbms-mapping-constraints2.png>

## Many-to-Many

In many-to-many mapping, an entity in E1 is associated with any number of entities in E2, and an entity in E2 is associated with any number of entities in E1.

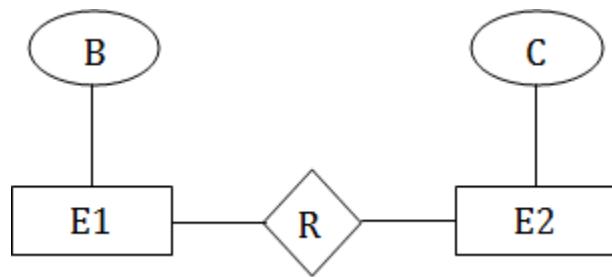


Image 18 – One to Many

Reference - <https://static.javatpoint.com/dbms/images/dbms-mapping-constraints4.png>

## Example of Employee Management System

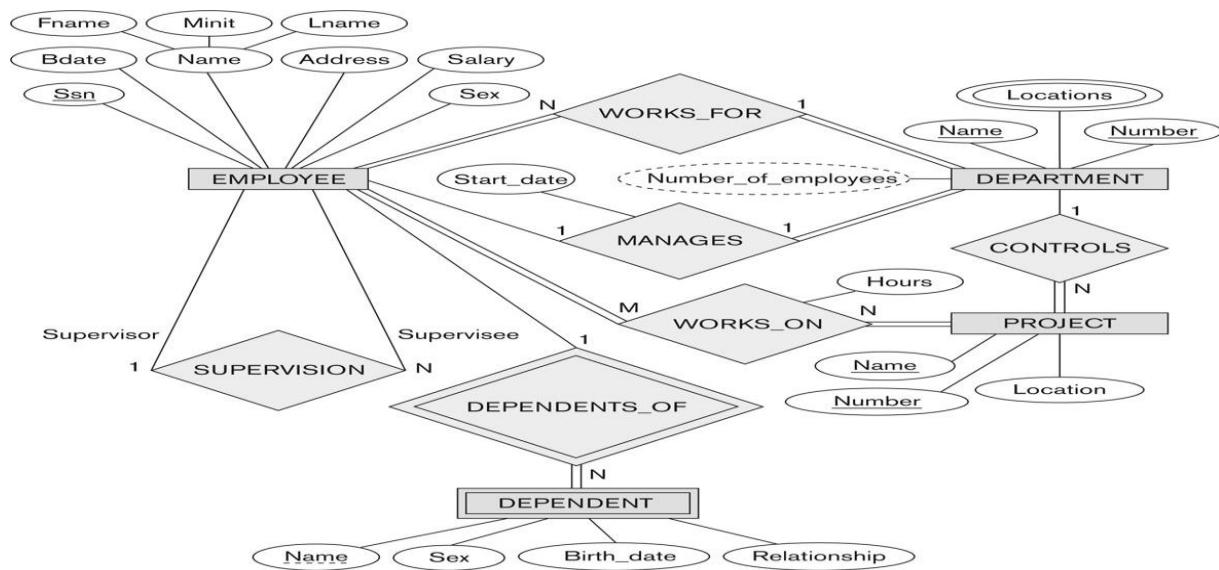


Image 19 – Employee Management

Reference - <http://pld.cs.luc.edu/database/images/fig7.2.png>

## Example of Movie Ticket Management

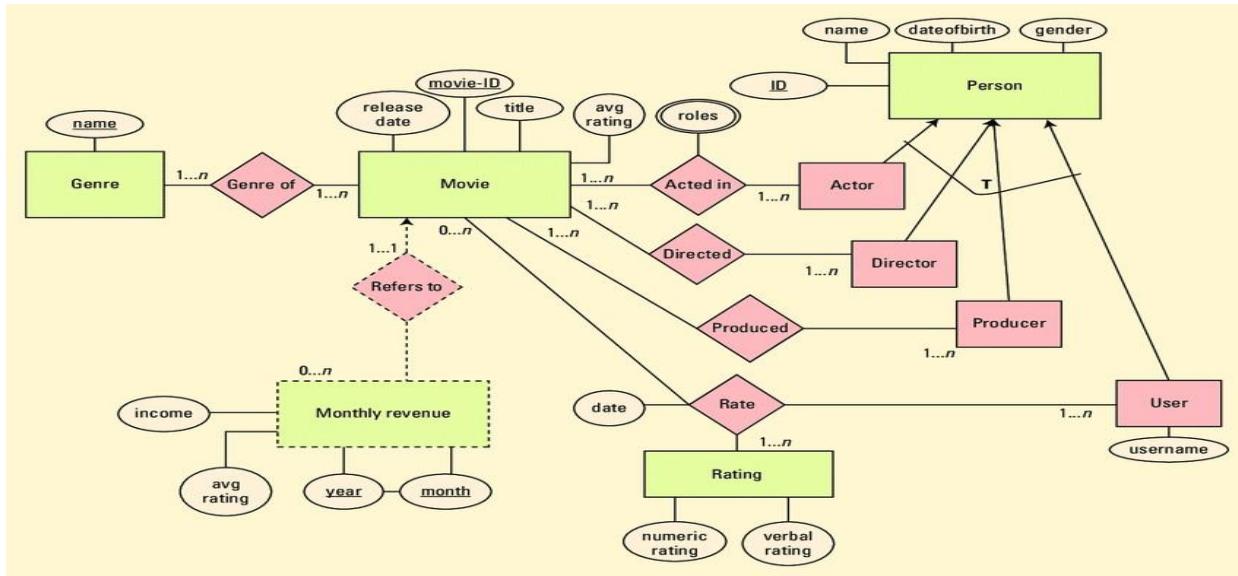


Image 20- Movie Ticket Management

Reference -

[https://www.researchgate.net/profile/Peretz\\_Shoval/publication/321352935/figure/fig1/AS:571569942142976@1513284301301/The-entity-relationship-diagram-for-the-movie-recommendation-system-Subtypes-are-not.png](https://www.researchgate.net/profile/Peretz_Shoval/publication/321352935/figure/fig1/AS:571569942142976@1513284301301/The-entity-relationship-diagram-for-the-movie-recommendation-system-Subtypes-are-not.png)

## Entity Relationship Diagram

### Introduction

ER model stands for an Entity-Relationship model. It is a high-level data model. This model is used to define the data elements and relationship for a specified system.

It develops a conceptual design for the database. It also develops a very simple and easy to design view of data.

In ER modeling, the database structure is portrayed as a diagram called an entity-relationship diagram.

### Components of ER Diagram

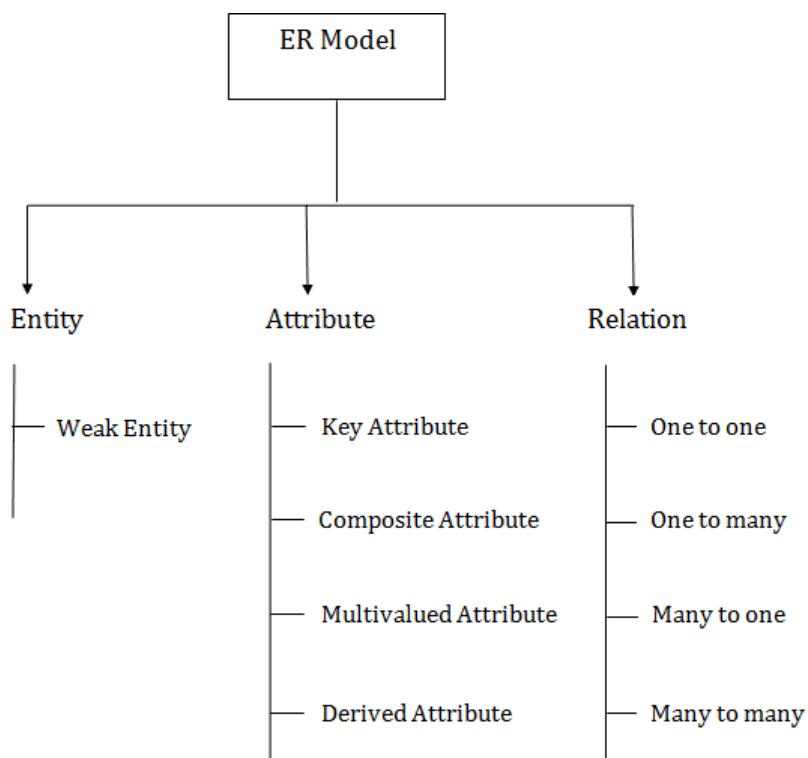


Image 21 – Components of ERD

Reference - <https://static.javatpoint.com/dbms/images/dbms-er-model-concept-diagram.png>

## Entity

An entity may be any object, class, person or place. In the ER diagram, an entity can be represented as rectangles.

Consider an organization as an example- manager, product, employee, department etc. can be taken as an entity.

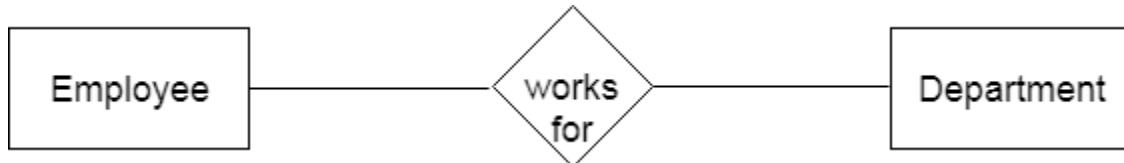


Image 22 – Entity

Reference - <https://static.javatpoint.com/dbms/images/dbms-er-model-concept2.png>

### a. Weak Entity

#### Entity

An entity that depends on another entity called a weak entity. The weak entity doesn't contain any key attribute of its own. The weak entity is represented by a double rectangle.

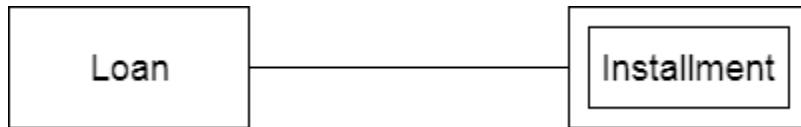


Image 23 – Week Entity

Reference - <https://static.javatpoint.com/dbms/images/dbms-er-model-concept3.png>

Examples of entities:

Person: Employee, Student, PatientPlace: Store, Building

Object: Machine, product, and Car

Event: Sale, Registration, RenewalConcept: Account, Course

### Difference between Strong and Week Entity

Strong Entity Set	Weak Entity Set
Strong entity set always has a primary key.	It does not have enough attributes to build a primary key.
It is represented by a rectangle symbol.	It is represented by a double rectangle symbol.
It contains a Primary key represented by the underline symbol.	It contains a Partial Key which is represented by a dashed underline symbol.
The member of a strong entity set is called as dominant entity set.	The member of a weak entity set called as a subordinate entity set.
Primary Key is one of its attributes which helps to identify its member.	In a weak entity set, it is a combination of primary key and partial key of the strong entity set.

In the ER diagram the relationship between two strong entity set shown by using a diamond symbol.

The connecting line of the strong entity set with the relationship is single.

The relationship between one strong and a weak entity set shown by using the double diamond symbol.

The line connecting the weak entity set for identifying relationship is double.

## Attribute

The attribute is used to describe the property of an entity. Eclipse is used to represent an attribute.

**For example**, id, age, contact number, name, etc. can be attributes of a student.

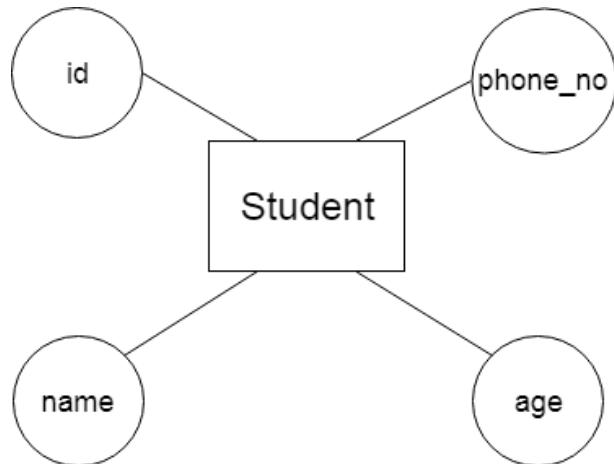


Image 23 – Attribute

Reference - <https://static.javatpoint.com/dbms/images/dbms-er-model-concept4.png>

### a. Key Attribute

The key attribute is used to represent the main characteristics of an entity. It represents a primary key. The key attribute is represented by an ellipse with the text underlined.

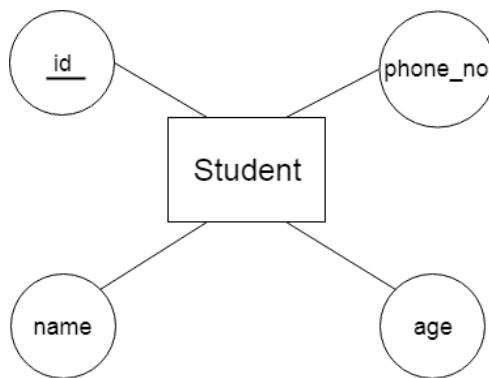


Image 24 – Key Attribute

Reference - <https://static.javatpoint.com/dbms/images/dbms-er-model-concept5.png>

### b. Composite Attribute

An attribute that composed of many other attributes is known as a composite attribute. The composite attribute is represented by an ellipse, and those ellipses are connected with an ellipse.

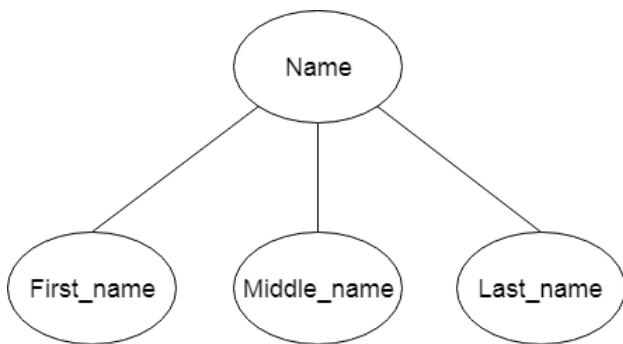


Image 25 – Composite Attribute

Reference - <https://static.javatpoint.com/dbms/images/dbms-er-model-concept6.png>

### c. Multivalued Attribute

An attribute can have more than one value. These attributes are known as a multivalued attribute. The double oval is used to represent multivalued attribute.

**For example**, a student can have more than one phone number.

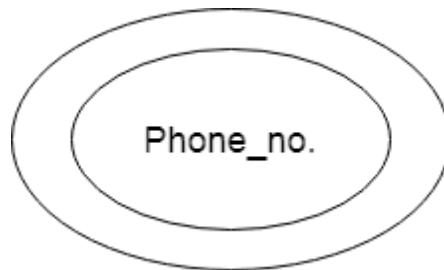


Image 26 – Multivalued Attribute

Reference - <https://static.javatpoint.com/dbms/images/dbms-er-model-concept7.png>

### d. Derived Attribute

An attribute that can be derived from other attribute is known as a derived attribute. It can be represented by a dashed ellipse.

**For example**, A person's age changes over time and can be derived from another attribute like Date of birth.

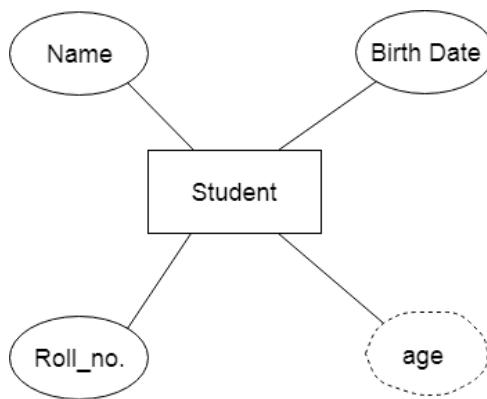


Image 27 – Derived Attribute

Reference - <https://static.javatpoint.com/dbms/images/dbms-er-model-concept8.png>

### 3. Relationship

A relationship is used to describe the relation between entities. Diamond or rhombus is used to represent the relationship.



Image 28 – Relationship Attribute

Reference - <https://static.javatpoint.com/dbms/images/dbms-er-model-concept9.png>

#### b. Weak Relationship

Transaction ID is weak relation between bank and customer account money withdrawal

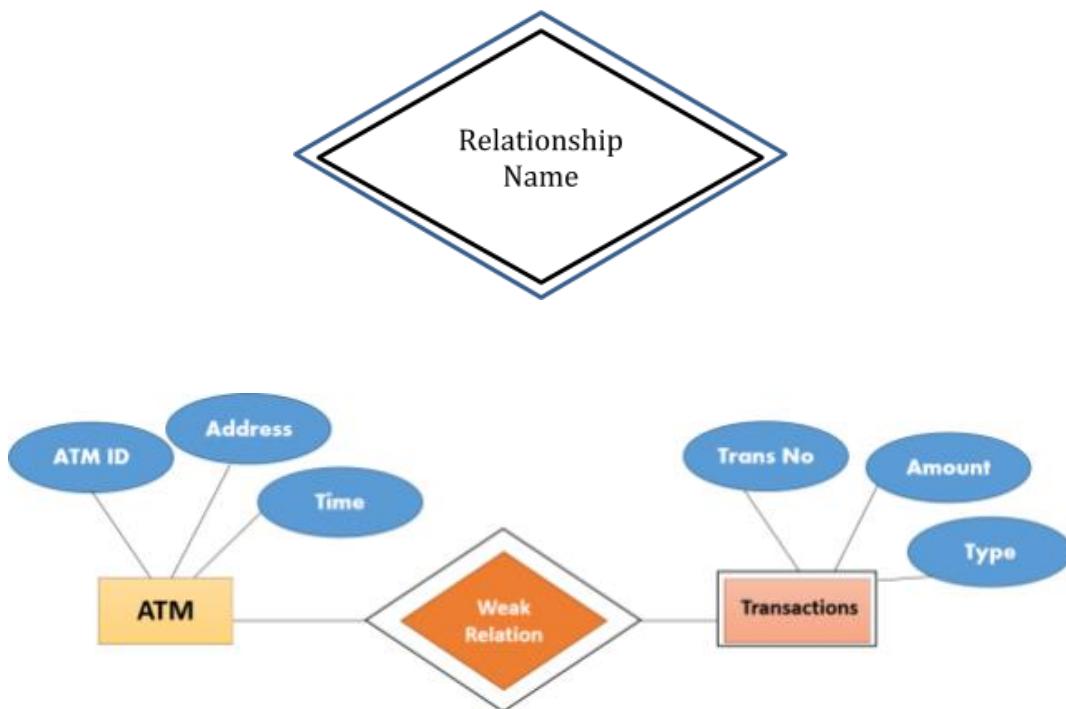


Image 29 – Relationship Attribute

Reference - [https://www.guru99.com/images/1/100518\\_0621\\_ERDiagramTu5.png](https://www.guru99.com/images/1/100518_0621_ERDiagramTu5.png)

## Example: Library Book Management

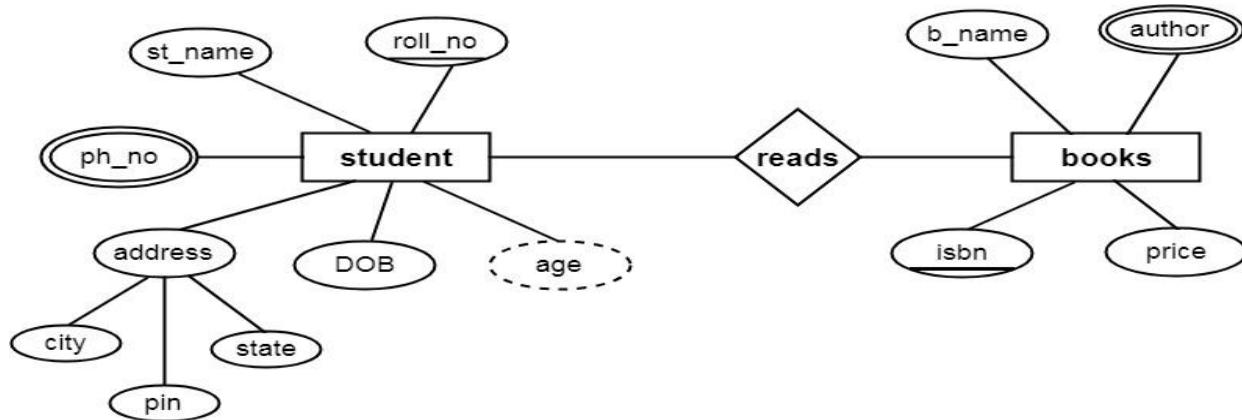


Image 30 – Library Book Management

Reference - <https://www.csetutor.com/wp-content/uploads/2018/09/ER-Diagram-in-DBMS-Example.png>

## Example: Employee Management System

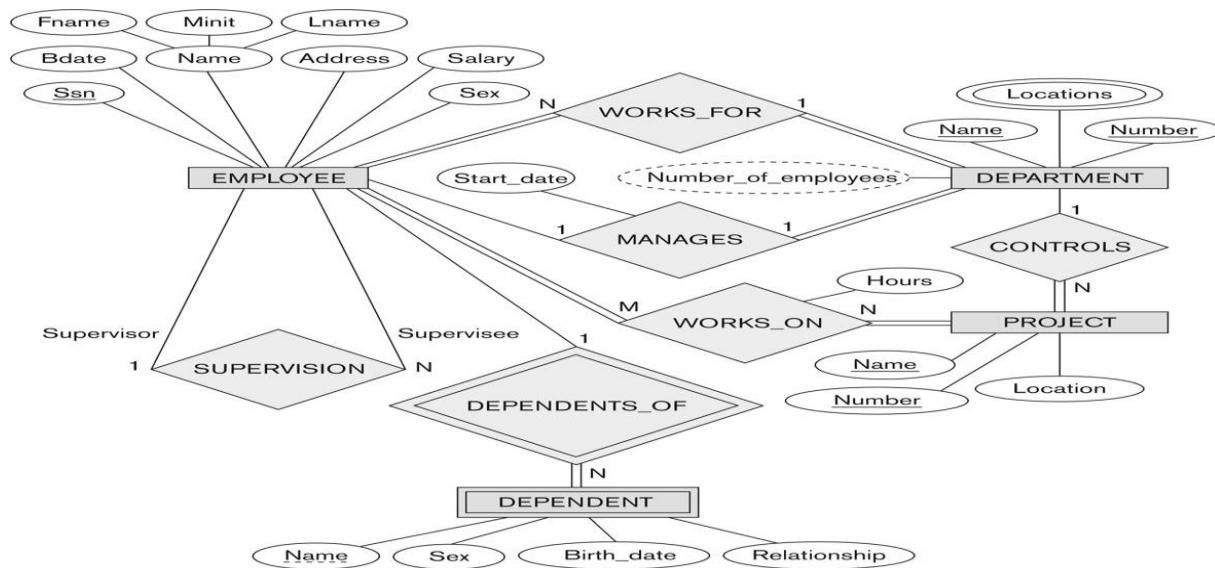


Image 31 – Employee Management

Reference - <http://pld.cs.luc.edu/database/images/fig7.2.png>

## Example: Banking Organization

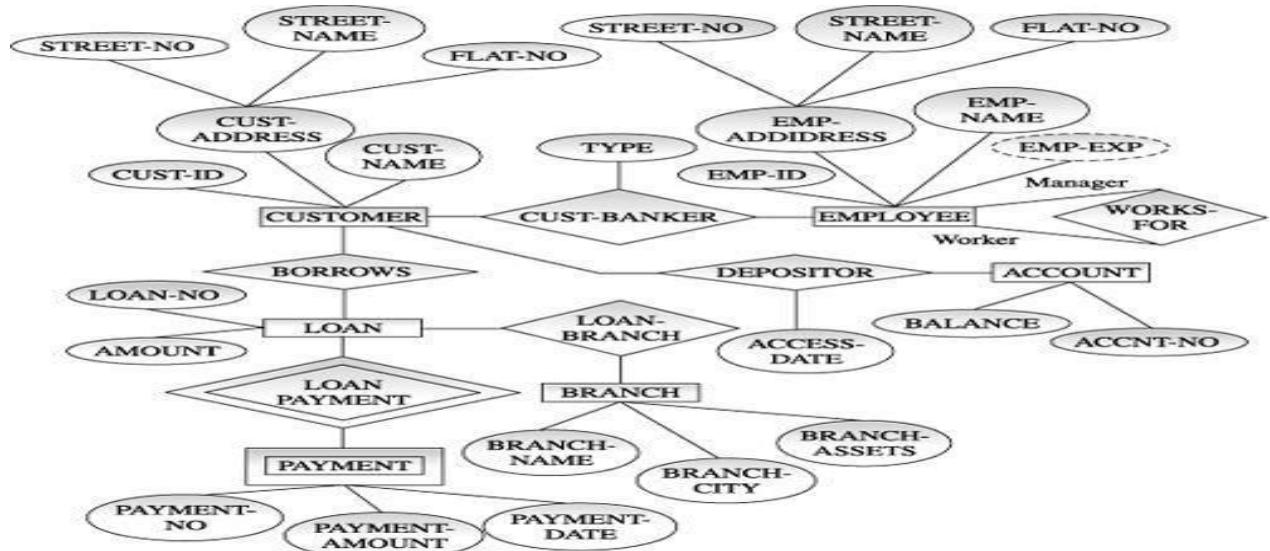


Image 32 – Banking Management

Reference - [https://www.oreilly.com/library/view/database-systems-concepts/9788177585674/9788177585674\\_ch06lev1sec5\\_image01.jpeg](https://www.oreilly.com/library/view/database-systems-concepts/9788177585674/9788177585674_ch06lev1sec5_image01.jpeg)

## Example: Hospital Management



Image 33 – Banking Management

Reference -

[https://lh3.googleusercontent.com/proxy/ApY5GXZPliyg9sRl18JwNH8N4Dyd3wAZQNwwtlEwk4Q89tIIJq\\_P4RwtIKUvVEQw0zaS4VydBz1soUEARVNfpHv9YLu5y17jQ9EdTrGGD1zgWnJYvx09MMc9C7Y70](https://lh3.googleusercontent.com/proxy/ApY5GXZPliyg9sRl18JwNH8N4Dyd3wAZQNwwtlEwk4Q89tIIJq_P4RwtIKUvVEQw0zaS4VydBz1soUEARVNfpHv9YLu5y17jQ9EdTrGGD1zgWnJYvx09MMc9C7Y70)

## Relational Model

### Introduction

Relational Model was proposed by E.F. Codd to model data in the form of relations or tables. After designing the conceptual model of Database using ER diagram, we need to convert the conceptual model in the relational model which can be implemented using any RDBMS languages like Oracle SQL, MySQL etc. So we will see what Relational Model is.

### What is Relational Model?

Relational Model represents how data is stored in Relational Databases. A relational database stores data in the form of relations (tables). Consider a relation STUDENT with attributes ROLL\_NO, NAME, ADDRESS, PHONE and AGE shown in Table 1

<b>STUDENT</b>				
<b>ROLL_NO</b>	<b>NAME</b>	<b>ADDRESS</b>	<b>PHONE</b>	<b>AGE</b>
1	RAM	DELHI	9455123451	18
2	RAMESH	GURGAON	9652431543	18
3	SUJIT	ROHTAK	9156253131	20
4	SURESH	DELHI	9156253132	18

### IMPORTANT TERMINOLOGIES

**Attribute:** Attributes are the properties that define a relation. e.g.; ROLL\_NO, NAME

**Relation Schema:** A relation schema represents name of the relation with its attributes. e.g.; STUDENT (ROLL\_NO, NAME, ADDRESS, PHONE and AGE) is relation schema for STUDENT. If a schema has more than 1 relation, it is called Relational Schema.

**Tuple:** Each row in the relation is known as tuple. The above relation contains 4 tuples, one of which is shown as:

1      RAM    DELHI      9455123451    18

**Relation Instance:** The set of tuples of a relation at a particular instance of time is called as relation instance. Table 1 shows the relation instance of STUDENT at a particular time. It can change whenever there is insertion, deletion or updation in the database.

**Degree:** The number of attributes in the relation is known as degree of the relation. The STUDENT relation defined above has degree 5.

**Cardinality:** The number of tuples in a relation is known as cardinality. The STUDENT relation defined above has cardinality 4.

**Column:** Column represents the set of values for a particular attribute. The column ROLL\_NO is extracted from relation STUDENT.

ROLL\_NO 1234

**NULL Values:** The value which is not known or unavailable is called NULL value. It is represented by blank space. e.g.; PHONE of STUDENT having ROLL\_NO 4 is NULL.

### Constraints in Relational Model

While designing Relational Model, we define some conditions which must hold for data present in database are called Constraints. These constraints are checked before performing any operation (insertion, deletion and updation) in database. If there is a violation in any of constraints, operation will fail.

**Domain Constraints:** These are attribute level constraints. An attribute can only take values which lie inside the domain range. e.g.; If a constraint AGE>0 is applied on STUDENT relation, inserting negative value of AGE will result in failure.

**Key Integrity:** Every relation in the database should have at least one set of attributes which defines a tuple uniquely. Those set of attributes is called key. e.g.; ROLL\_NO in STUDENT is a key. No two students can have same roll number. So a key has two properties:

It should be unique for all tuples. It can't have NULL values.

**Referential Integrity:** When one attribute of a relation can only take values from other attribute of same relation or any other relation, it is called referential integrity. Let us suppose we have 2 relations

## STUDENT

<b>ROLL_N O</b>	<b>NAME</b>	<b>ADDRESS</b>	<b>PHONE</b>	<b>AG E</b>	<b>BRANCH_COD E</b>
1	RAM	DELHI	945512345 1	18	CS
2	RAMES H	GURGAO N	965243154 3	18	CS
3	SUJIT	ROHTAK	915625313 1	20	ECE
4	SURESH	DELHI		18	IT

## BRANCH

### **BRANCH\_CODE    BRANCH\_NAME**

CS	COMPUTER SCIENCE
IT	INFORMATION TECHNOLOGY
ECE	ELECTRONICS AND COMMUNICATION ENGINEERING
CV	CIVIL ENGINEERING

BRANCH\_CODE of STUDENT can only take the values which are present in BRANCH\_CODE of BRANCH which is called referential integrity constraint. The relation which is referencing to other relation is called REFERENCING RELATION (STUDENT in this case) and the relation to which other relations refer is called REFERENCED RELATION(BRANCH in this case).

An anomaly is an irregularity, or something which deviates from the expected or normal state. When designing databases, we identify three types of anomalies: Insert, Update and Delete.

Insertion Anomaly in Referencing Relation:

We can't insert a row in REFERENCING RELATION if referencing attribute's value is not present in referenced attribute value. e.g.; Insertion of a student with BRANCH\_CODE 'ME' in

---

STUDENT relation will result in error because 'ME' is not present in BRANCH\_CODE of BRANCH.

### Deletion/ Updation Anomaly in Referenced Relation:

We can't delete or update a row from REFERENCED RELATION if value of REFERENCED ATTRIBUTE is used in value of REFERENCING ATTRIBUTE. e.g; if we try to delete tuple from BRANCH having BRANCH\_CODE 'CS', it will result in error because 'CS' is referenced by BRANCH\_CODE of STUDENT, but if we try to delete the row from BRANCH with BRANCH\_CODE CV, it will be deleted as the value is not been used by referencing relation. It can be handled by following method:

**ON DELETE CASCADE:** It will delete the tuples from REFERENCING RELATION if value used by REFERENCING ATTRIBUTE is deleted from REFERENCED RELATION. e.g; if we delete a row from BRANCH with BRANCH\_CODE 'CS', the rows in STUDENT relation with BRANCH\_CODE CS (ROLL\_NO 1 and 2 in this case) will be deleted.

**ON UPDATE CASCADE:** It will update the REFERENCING ATTRIBUTE in REFERENCING RELATION if attribute value used by REFERENCING ATTRIBUTE is updated in REFERENCED RELATION. e.g; if we update a row from BRANCH with BRANCH\_CODE 'CS' to 'CSE', the rows in STUDENT relation with BRANCH\_CODE CS (ROLL\_NO 1 and 2 in this case) will be updated with BRANCH\_CODE 'CSE'.

### SUPER KEYS:

Any set of attributes that allows us to identify unique rows (tuples) in a given relation are known as super keys. Out of these super keys we can always choose a proper subset among these which can be used as a primary key. Such keys are known as Candidate keys. If there is a combination of two or more attributes which is being used as the primary key then we call it as a Composite key.

**Attribute** - Each column in a Table. Attributes are the properties which define a relation. e.g., Student\_Rollno, NAME,etc.

**Tables** – In the Relational model the, relations are saved in the table format. It is stored alongwith its entities. A table has two properties rows and columns. Rows represent records and columns represent attributes.

**Tuple** – It is nothing but a single row of a table, which contains a single record.

**Relation Schema:** A relation schema represents the name of the relation with its attributes.

- The total number of attributes which in the relation is called the degree of the relation. **Cardinality**
- Total number of rows present in the Table.

**Column** - The column represents the set of values for a specific attribute.

**Relation instance** – Relation instance is a finite set of tuples in the RDBMS system. Relation instances never have duplicate tuples.

**Relation Key** - Every row has one, two or multiple attributes, which is called relation key.

**Attribute Domain** – Every attribute has some pre-defined value and scope which is known as attribute domain.

### Properties of Relation

- Name of the relation is distinct from all other relations.
- Each relation cell contains exactly one atomic (single) value
- Each attribute contains a distinct name
- Attribute domain has no significance
- Tuple has no duplicate value
- Order of tuple can have a different sequence

### Example 1

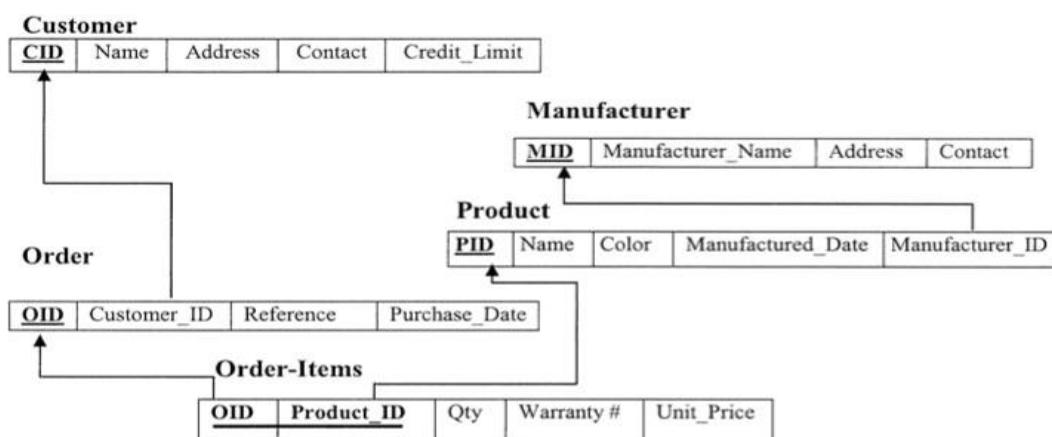


Image 34 – Production Management

Reference - <https://d1whlypfis84e.cloudfront.net/guides/wp-content/uploads/2019/01/01111318/Relational-databse.png>

## Example 2

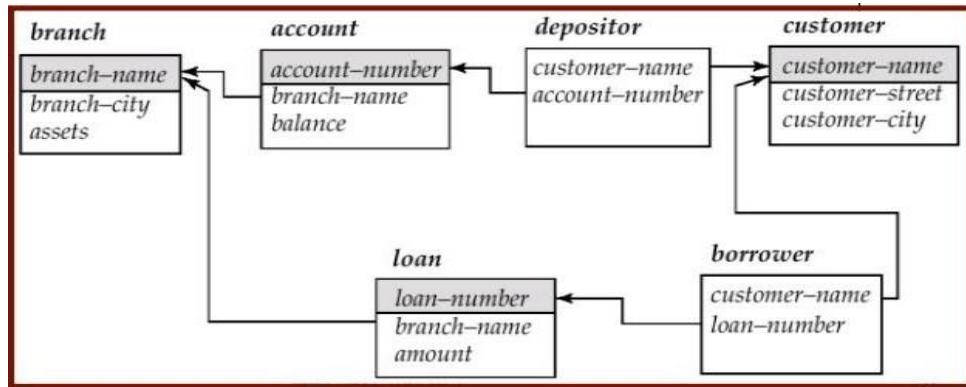


Image 35 – Production Management

Reference -

[https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.slideshare.net%2FProsantaGhosh%2Fdbms-ii-mcach4relational-model2013&psi=g=AOvVaw1UFOBY4OLPKvlkom0Oijh9&ust=1587013292216000&source=images&cd=vfe&ved=0CAIQjRxqFwoTCJC5\\_K\\_T6egCFQAAA AAdAAAAABAD](https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.slideshare.net%2FProsantaGhosh%2Fdbms-ii-mcach4relational-model2013&psi=g=AOvVaw1UFOBY4OLPKvlkom0Oijh9&ust=1587013292216000&source=images&cd=vfe&ved=0CAIQjRxqFwoTCJC5_K_T6egCFQAAA AAdAAAAABAD)

---

## Network Model

### Introduction

Charles Bachman was the original inventor of the network model. In 1969, the Conference on Data Systems Languages (CODASYL) Consortium developed the network model into a standard specification. A second publication was introduced in 1971, which later turned into the basis for virtually all implementations.

The benefits of the network model include:

**Simple Concept:** Similar to the hierarchical model, this model is simple and the implementation is effortless.

**Ability to Manage More Relationship Types:** The network model has the ability to manage one-to-one (1:1) as well as many-to-many (N: N) relationships.

**Easy Access to Data:** Accessing the data is simpler when compared to the hierarchical model.

**Data Integrity:** In a network model, there's always a connection between the parent and the child segments because it depends on the parent-child relationship.

**Data Independence:** Data independence is better in network models as opposed to the hierarchical models.

The drawbacks of the network model include:

**System Complexity:** Each and every record has to be maintained with the help of pointers, which makes the database structure more complex.

**Functional Flaws:** Because a great number of pointers is essential, insertion, updates, and deletion become more complex.

**Lack of Structural Independence:** A change in structure demands a change in the application as well, which leads to lack of structural independence.

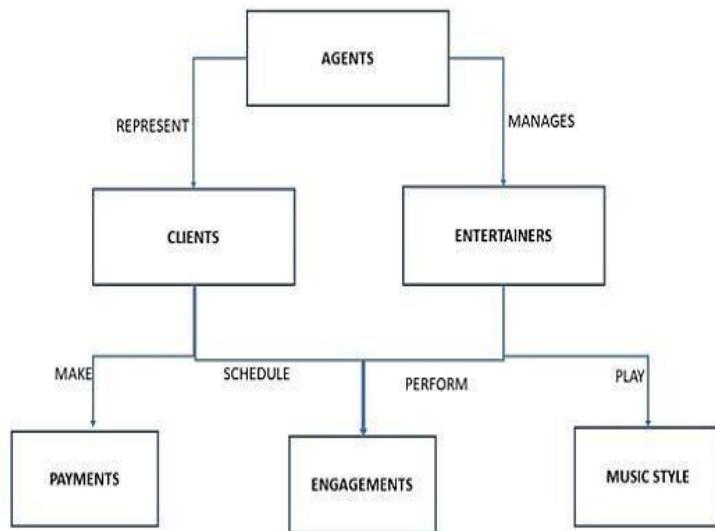


Image 36 – Network Model

Reference - <https://www.tutorialspoint.com/assets/questions/images/120543-1532343127.jpg>

The network model is the extension of the hierarchical structure because it allows many-to-many relationships to be managed in a tree-like structure that allows multiple parents.

It can represent redundancy in data more efficiently than that in the hierarchical model. There can be more than one path from a previous node to successor node/s.

The operations of the network model are maintained by indexing structure of linked list (circular) where a program maintains a current position and navigates from one record to another by following the relationships in which the record participates.

Records can also be located by supplying key values.

## Hierarchical Model

### Introduction

A hierarchical model represents the data in a tree-like structure in which there is a single parent for each record. To maintain order there is a sort field which keeps sibling nodes into a recorded manner. These types of models are designed basically for the early mainframe database management systems, like the Information Management System (IMS) by IBM.

This model structure allows the one-to-one and a one-to-many relationship between two/ various types of data. This structure is very helpful in describing many relationships in the real world; table of contents, any nested and sorted information.

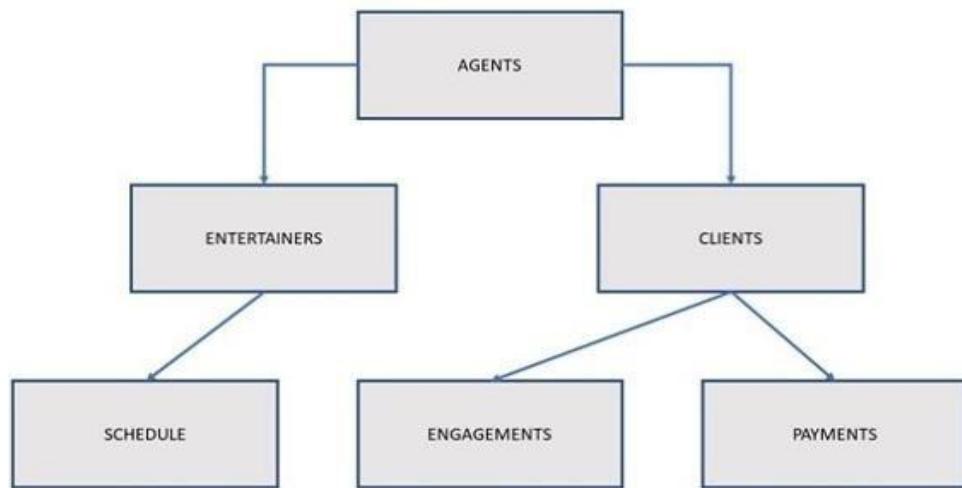


Image 37 – Hierarchical Model

Reference - <https://www.tutorialspoint.com/assets/questions/images/154411-1532346635.jpg>

### Advantages

A user can retrieve data very quickly due to the presence of explicit links between the table structures.

The referential integrity is built in and automatically enforced due to which a record in a child table must be linked to an existing record in a parent table, along with that if a record deleted in the parent table then that will cause all associated records in the child table to be deleted as well.

## Disadvantages

When a user needs to store a record in a child table that is currently unrelated to any record in a parent table, it gets difficulty in recording and user must record an additional entry in the parent table.

This type of database cannot support complex relationships, and there is also a problem of redundancy, which can result in producing inaccurate information due to the inconsistent recording of data at various sites.

### Example

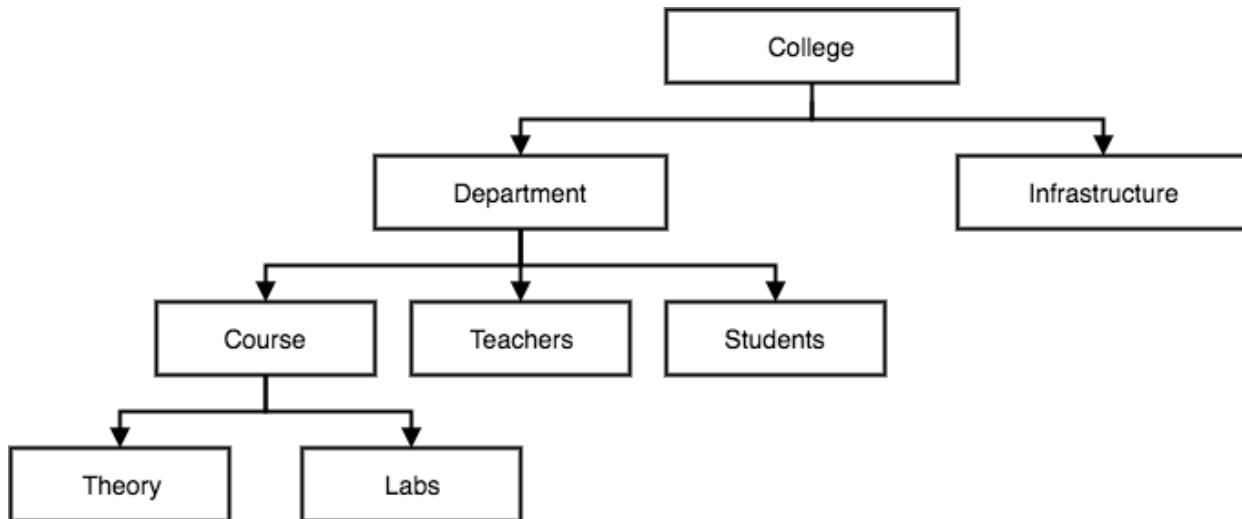


Image 38 – Hierarchical Model

Reference - <https://www.studytonight.com/dbms/images/hierarchical-dbms-model.png>

## Relational Database Management System

### Introduction

RDBMS stands for Relational Database Management Systems.

All modern database management systems like SQL, MS SQL Server, IBM DB2, ORACLE, MySQL and Microsoft Access are based on RDBMS.

It is called Relational Data Base Management System (RDBMS) because it is based on relational model introduced by E.F. Codd.

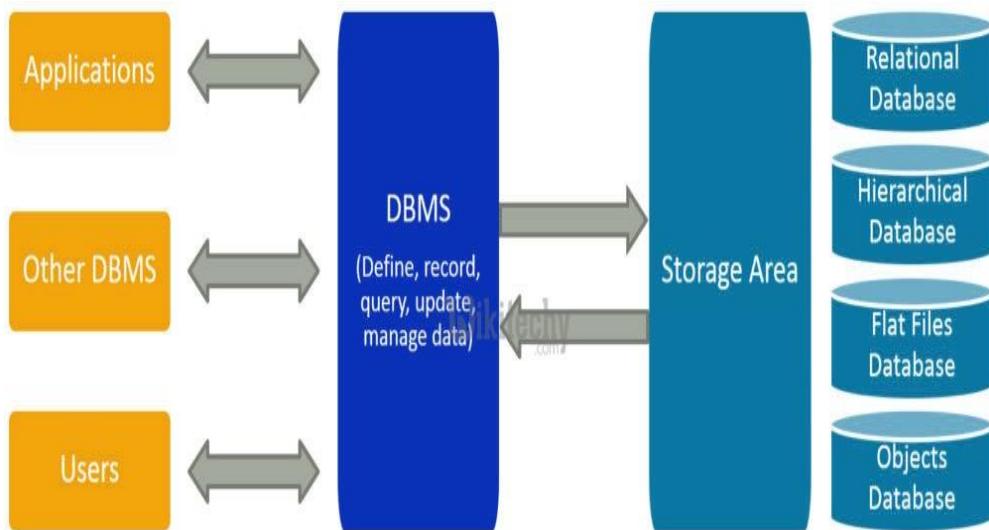


Image 39 – RDBMS Model

Reference - <https://d5ngkkf53wl41.cloudfront.net/interview-questions/dbms/what-is-dbms.png>

It is the process of making a description of the execution of the database on secondary storage, which describes the base relations, file organizations as well as indexes used to gain efficient access to the data and any associated integrity constraints and security measures.

## Characteristics of physical database design

It typically illustrates data requirements for a single project or application. Sometimes even apart of an application

May be incorporated into other physical data models by means of a repository of shared entities

It typically includes 10-1000 tables; although these numbers are highly variable, depending on the scope of the data model

It has the relationships between tables that address cardinality and null ability (optionality) of the relationships

Designed and developed to be reliant on a specific version of a DBMS, storage location of data or technology

Database columns will have data types with accurate precisions and lengths assigned to them. Columns will have null ability (optional) assigned. Tables and columns will have specific definitions. Steps required to implement physical database.

The steps of the physical database design methodology are as follows:

Transform the logical data model for target DBMS

- Design base relations
- Design representation of derived data
- Design general constraints
- Design file organizations and indexes
- Analyze transactions
- Choose file organizations
- Choose indexes
- Estimate disk space requirements
- Design user views
- Design security mechanisms
- Consider the introduction of controlled redundancy
- Monitor and tune the operational system

---

## Database Engine

A database engine (or storage engine) is the underlying software component that a database management system (DBMS) uses to create, read, update and delete (CRUD) data from a database. Most database management systems include their own application programming interface (API) that allows the user to interact with their underlying engine without going through the user interface of the DBMS.

The term "database engine" is frequently used interchangeably with "database server" or "database management system". A 'database instance' refers to the processes and memory structures of the running database engine.

## Database Schema

The database schema of a database is its structure described in a formal language supported by the database management system (DBMS). The term "schema" refers to the organization of data as a blueprint of how the database is constructed (divided into database tables in the case of relational databases). The formal definition of a database schema is a set of formulas (sentences) called integrity constraints imposed on a database.[citation needed] These integrity constraints ensure compatibility between parts of the schema. All constraints are expressible in the same language. A database can be considered a structure in realization of the database language. The states of a created conceptual schema are transformed into an explicit mapping, the database schema. This describes how real-world entities are modeled in the database.

"A database schema specifies, based on the database administrator's knowledge of possible applications, the facts that can enter the database, or those of interest to the possible end-users."

The notion of a database schema plays the same role as the notion of theory in predicate calculus. A model of this "theory" closely corresponds to a database, which can be seen at any instant of time as a mathematical object. Thus a schema can contain formulas representing integrity constraints specifically for an application and the constraints specifically for a type of database, all expressed in the same database language.

---

In a relational database, the schema defines the tables, fields, relationships, views, indexes, packages, procedures, functions, queues, triggers, types, sequences, materialized views, synonyms, database links, directories, XML schemas, and other elements.

A database generally stores its schema in a data dictionary. Although a schema is defined in text database language, the term is often used to refer to a graphical depiction of the database structure. In other words, schema is the structure of the database that defines the objects in the database.

## Advantages of relational database management system

The use of an RDBMS can be beneficial to most organizations; the systematic view of raw data helps companies better understand and execute the information while enhancing the

decision-making process. The use of tables to store data also improves the security of information stored in the databases. Users are able to customize access and set barriers to limit the content that is made available. This feature makes the RDBMS particularly useful to companies in which the manager decides what data is provided to employees and customers.

Furthermore, RDBMSes make it easy to add new data to the system or alter existing tables while ensuring consistency with the previously available content.

Other advantages of the RDBMS include:

- Flexibility -- updating data is more efficient since the changes only need to be made in one place.
- Maintenance -- database administrators can easily maintain, control and update data in the database. Backups also become easier since automation tools included in the RDBMS automate these tasks.
- Data structure -- the table format used in RDBMSes is easy to understand and provides an organized and structural manner through which entries are matched by firing queries.

## Disadvantages of RDBMS

- Software is expensive.
- Complex software refers to expensive hardware and hence increases overall cost to avail the RDBMS service.

- It requires skilled human resources to implement. Certain applications are slow in processing.
- It is difficult to recover the lost data.

## Applications of RDBMS

<b>Sector</b>	<b>Use of DBMS</b>
Banking	For customer information, account activities, payments, deposits, loans, etc.
Airlines	For reservations and schedule information.
Universities	For student information, course registrations, colleges and grades.
—	It helps to keep call records, monthly bills, maintaining
Finance	For storing information about stock, sales, and purchases of financial instruments like stocks and bonds.
Sales	Use for storing customer, product & sales information.

Manufacturing

It is used for the management of supply chain and for tracking production of items. Inventories status in warehouses.

HR Management

For information about employees, salaries, payroll,

## Structure of DBMS

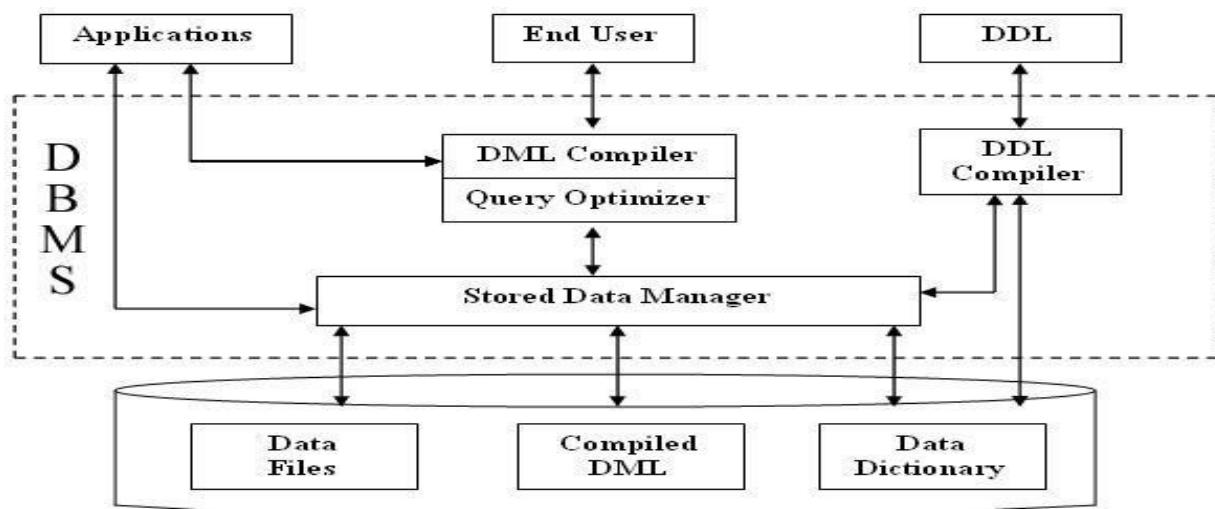


Image 40 – Structure of DBMS

Reference -

[http://1.bp.blogspot.com/GmlmEbglhWk/Vc12mfuuTYI/AAAAAAAABak/jYO7eiIdprw/s1600/Structure\\_of\\_DBMS\\_DBMSbasics.blogspot.com.jpg](http://1.bp.blogspot.com/GmlmEbglhWk/Vc12mfuuTYI/AAAAAAAABak/jYO7eiIdprw/s1600/Structure_of_DBMS_DBMSbasics.blogspot.com.jpg)

At very high level, a database is considered as shown in below diagram. Let us see them in detail below.

**Applications:** – It can be considered as a user-friendly web page where the user enters the requests. Here he simply enters the details that he needs and presses buttons to get the data.

**End User:** – They are the real users of the database. They can be developers, designers, administrator or the actual users of the database.

**DDL:** – Data Definition Language (DDL) is a query fired to create database, schema, tables, mappings etc in the database. These are the commands used to create the objects like tables, indexes in the database for the first time. In other words, they create structure of the database.

**DDL Compiler:** – This part of database is responsible for processing the DDL commands. That means this compiler actually breaks down the command into machine understandable codes. It is also responsible for storing the metadata information like table name, space used by it, number of columns in it, mapping information etc.

**DML Compiler:** – When the user inserts, deletes, updates or retrieves the record from the database, he will be sending request which he understands by pressing some buttons. But for the database to work/understand the request, it should be broken down to object code. This is done by this compiler. One can imagine this as when a person is asked some question, how this is broken down into waves to reach the brain!

**Query Optimizer:** – When user fires some request, he is least bothered how it will be fired on the database. He is not all aware of database or its way of performance. But whatever be the request, it should be efficient enough to fetch, insert, update or delete the data from the database. The query optimizer decides the best way to execute the user request which is received from the DML compiler. It is similar to selecting the best nerve to carry the waves to brain!

**Stored Data Manager:** – This is also known as Database Control System. It is one of the main central system of the database. It is responsible for various tasks

It converts the requests received from query optimizer to machine understandable form. It makes actual request inside the database. It is like fetching the exact part of the brain to answer.

It helps to maintain consistency and integrity by applying the constraints. That means, it does not allow inserting / updating / deleting any data if it has child entry. Similarly, it does not allow entering any duplicate value into database tables.

It controls concurrent access. If there are multiple users accessing the database at the same time, it makes sure, all of them see correct data. It guarantees that there is no data loss or data mismatch happens between the transactions of multiple users.

It helps to back up the database and recover data whenever required. Since it is a huge database and when there is any unexpected exploit of transaction, and reverting the changes are not easy. It maintains the backup of all data, so that it can be recovered.

**Data Files:** – It has the real data stored in it. It can be stored as magnetic tapes, magnetic disks or optical disks.

**Compiled DML:** – Some of the processed DML statements (insert, update, delete) are stored init so that if there is similar requests, it will be re-used.

**Data Dictionary:** – It contains all the information about the database. As the name suggests, it is the dictionary of all the data items. It contains description of all the tables, view, materialized views, constraints, indexes, triggers etc.

### DBMS VS File System

DBMS	File System
In DBMS, the user is not required to write the procedures.	In this system, the user has to write the procedures for managing the database.
DBMS gives an abstract view of data that hides the details.	File system provides the detail of the data representation and storage of data.
DBMS provides a crash recovery mechanism, i.e., DBMS protects the user from the system failure.	File system doesn't have a crash mechanism, i.e., if the system crashes while entering some data, then the content of the file will be lost.
DBMS provides a good protection mechanism.	It is very difficult to protect a file under the file system.
DBMS contains a wide variety of sophisticated techniques to store and retrieve the data.	File system can't efficiently store and retrieve the data.
DBMS takes care of Concurrent access of data using some form of locking.	In the File system, concurrent access has many problems like redirecting the file while other deleting some information or updating some information.

## DBMS VS RDBMS

No.	DBMS	RDBMS
1)	DBMS applications store data as file.	RDBMS applications store data in a tabular form.
2)	In DBMS, data is generally stored in either a hierarchical form or a navigational form.	In RDBMS, the tables have an identifier called primary key and the data values are stored in the form of tables.
3)	Normalization is not present in DBMS.	Normalization is present in RDBMS.
4)	DBMS does not apply any security with regards to data manipulation.	RDBMS defines the integrity constraint for the purpose of ACID (Atomocity, Consistency, Isolation and Durability) property.
5)	DBMS uses file system to store data, so there will be no relation between the tables.	In RDBMS, data values are stored in the form of tables, so a relationship between these data values will be stored in the form of a table as well.
6)	DBMS does not support distributed database.	RDBMS supports distributed database.
7)	DBMS is meant to be for small organization and deal with small data. It supports single user.	RDBMS is designed to handle large amount of data. It supports multiple users.
8)	Examples of DBMS are filesystems, XML etc.	Examples of RDBMS are MySQL, PostgreSQL, SQL Server, Oracle etc.

## 1-Tier Architecture

- In this architecture, the database is directly available to the user. It means the user can directly sit on the DBMS and use it.
- Any changes done here will directly be done on the database itself. It doesn't provide a handy tool for end users.
- The 1-Tier architecture is used for development of the local application, where programmers can directly communicate with the database for the quick response.

## 2-Tier Architecture

- The 2-Tier architecture is same as basic client-server. In the two-tier architecture, applications on the client end can directly communicate with the database at the serverside. For this interaction, API's like: **ODBC, JDBC** are used.
- The user interfaces and application programs are run on the client-side.
- The server side is responsible to provide the functionalities like: query processing and transaction management.
- To communicate with the DBMS, client-side application establishes a connection with the server side.

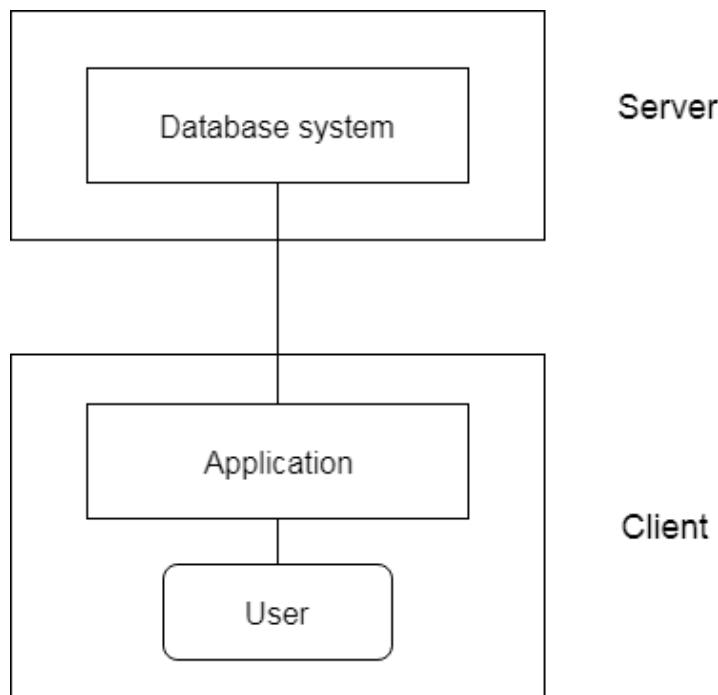


Image 41 – 2 tier Architecture

Reference - <https://static.javatpoint.com/dbms/images/dbms-2-tier-architecture.png>

### 3-Tier Architecture

- The 3-Tier architecture contains another layer between the client and server. In this architecture, client can't directly communicate with the server.
- The application on the client-end interacts with an application server which further communicates with the database system.
- End user has no idea about the existence of the database beyond the application server. The database also has no idea about any other user beyond the application.
- The 3-Tier architecture is used in case of large web application.

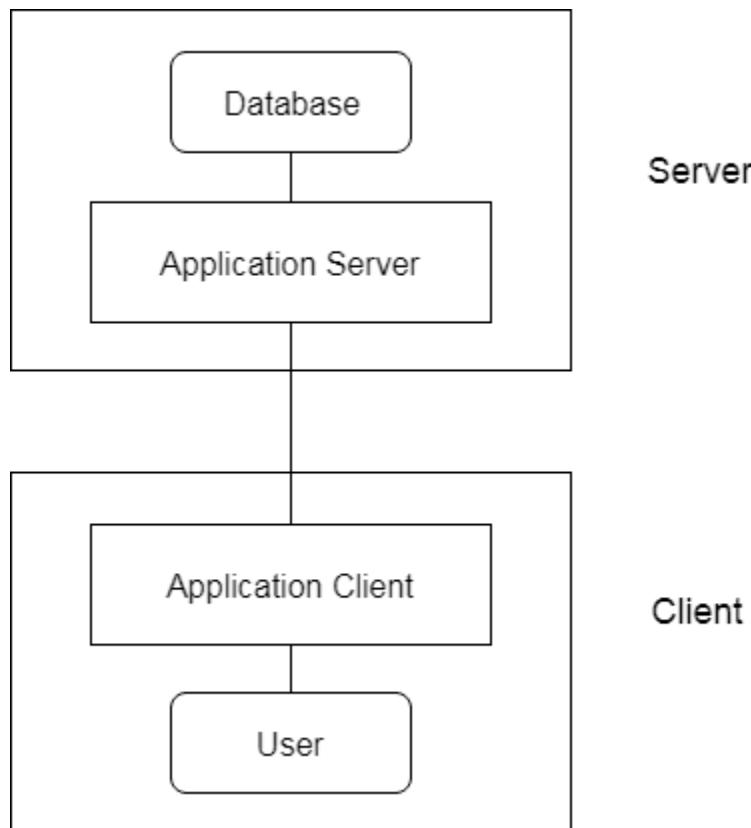


Image 42 – 3 tier Architecture

Reference - <https://static.javatpoint.com/dbms/images/dbms-3-tier-architecture.png>

## Three Schema Architecture

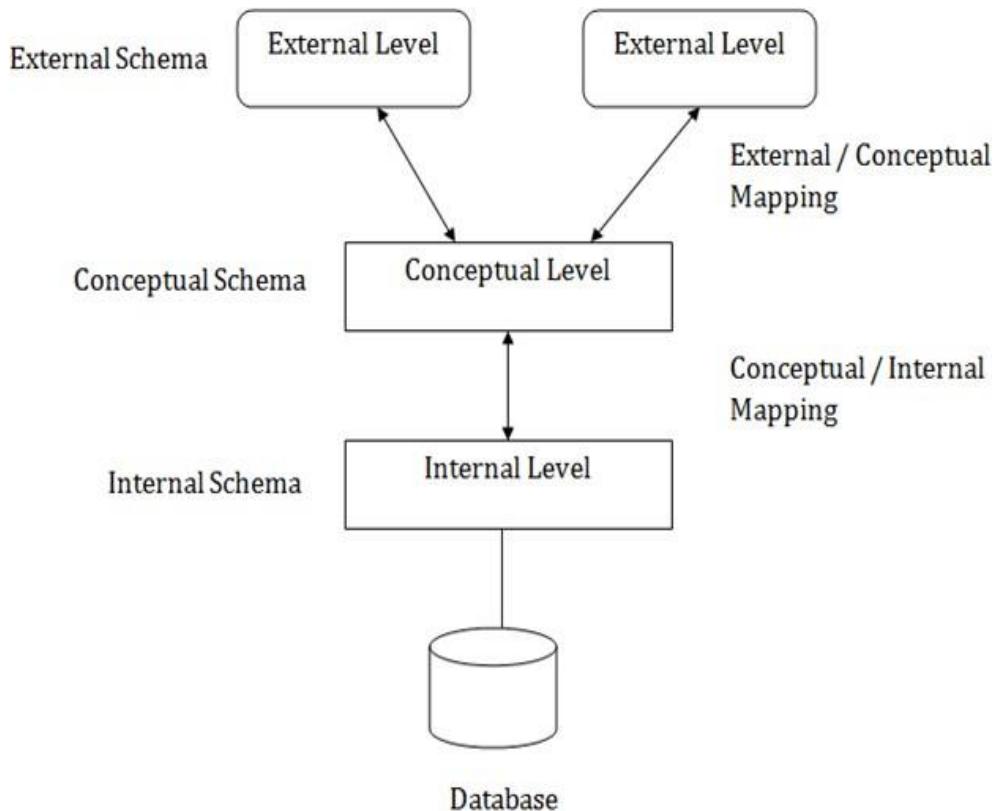


Image 43 – 3 Schema Architecture

Reference - <https://static.javatpoint.com/dbms/images/dbms-three-schema-architecture.png>

This framework is used to describe the structure of a specific database system.

The three schema architecture is also used to separate the user applications and physical database.

The three schema architecture contains three-levels. It breaks the database down into three different categories:

### Internal Level

The internal level has an internal schema which describes the physical storage structure of the database.

The internal schema is also known as a physical schema. It uses the physical data model.

It is used to define that how the data will be stored in a block.

---

The physical level is used to describe complex low-level data structures in detail.

## Conceptual Level

The conceptual schema describes the design of a database at the conceptual level. Conceptual level is also known as logical level.

The conceptual schema describes the structure of the whole database.

The conceptual level describes what data are to be stored in the database and also describes what relationship exists among those data.

Programmers and database administrators work at this level.

## External Level

At the external level, a database contains several schemas that sometimes called as subschema. The subschema is used to describe the different view of the database.

An external schema is also known as view schema.

The view schema describes the end user interaction with database systems.

## **Relational Algebra**

Relational Algebra is procedural query language, which takes Relation as input and generate relation as output.

Relational algebra operations are performed recursively on a relation.

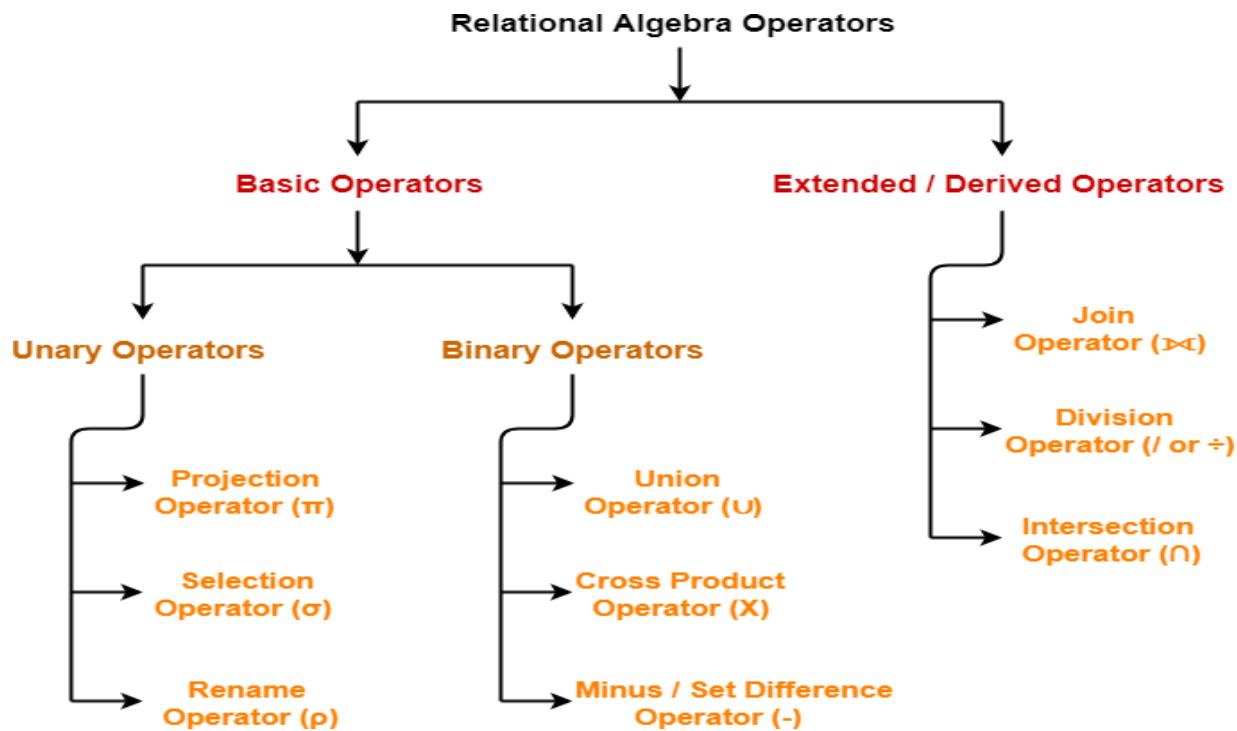


Image 44 – Relational Algebra

Reference -

[https://encrypted-tbn0.gstatic.com/images?q=tbn%3AANd9GcRrt8Z79xBi\\_fs1wYROuo33pDaGmXFbTyRBWowkH\\_UGkPdrf2Ft&usqp=CAU](https://encrypted-tbn0.gstatic.com/images?q=tbn%3AANd9GcRrt8Z79xBi_fs1wYROuo33pDaGmXFbTyRBWowkH_UGkPdrf2Ft&usqp=CAU)

## Select Operation

The select operation selects tuples that satisfy a given predicate. It is denoted by sigma ( $\sigma$ ).

Notation:  $\sigma p(r)$  Where:

$\sigma$  is used for selection prediction  
 $r$  is used for relation

$p$  is used as a propositional logic formula which may use connectors like: AND OR and NOT. These relational can use as relational operators like  $=, \neq, \geq, \leq, <, \leq$ .

For example: LOAN Relation

BRANCH_NAME	LOAN_NO	AMOUNT
Downtown	L-17	1000
Redwood	L-23	2000

Perryride	L-15	1500
Downtown	L-14	1500
Mianus	L-13	500
Roundhill	L-11	900
Perryride	L-16	1300

**Input:**

$\sigma \text{BRANCH\_NAME} = \text{"perryride"} (\text{LOAN})$

**Output:**

BRANCH_NAME	LOAN_NO	AMOUNT
Perryride	L-15	1500
Perryride	L-16	1300

## Project Operation

This operation shows the list of those attributes that we wish to appear in the result. Rest of the attributes are eliminated from the table.

It is denoted by  $\prod$ . Notation:  $\prod A_1, A_2, A_n (r)$  Where

$A_1, A_2, A_3$  is used as an attribute name of relation  $r$ .

Example: CUSTOMER RELATION

NAME	STREET	CITY
Jones	Main	Harrison
Smith	North	Rye
Hays	Main	Harrison
Curry	North	Rye
Johnson	Alma	Brooklyn

Brooks	Senator	Brooklyn
--------	---------	----------

**Input:**

$\Pi$  NAME, CITY (CUSTOMER)

**Output:**

NAME	CITY
Jones	Harrison
Smith	Rye
Hays	Harrison

### Union Operation

Suppose there are two tuples R and S. The union operation contains all the tuples that are either in R or S or both in R & S.

It eliminates the duplicate tuples. It is denoted by  $\cup$ . Notation:  $R \cup S$

A union operation must hold the following condition: R and S must have the attribute of the same number. Duplicate tuples are eliminated automatically.

Example:

### DEPOSITOR RELATION

CUSTOMER_NAME	ACCOUNT_NO
Johnson	A-101
Smith	A-121
Mayes	A-321
Turner	A-176
Johnson	A-273
Jones	A-472
Lindsay	A-284

## BORROW RELATION

CUSTOMER_NAME	LOAN_NO
Jones	L-17
Smith	L-23
Hayes	L-15
Jackson	L-14
Curry	L-93
Smith	L-11
Williams	L-17

$\prod \text{CUSTOMER\_NAME} (\text{BORROW}) \cup {}^U \prod \text{CUSTOMER\_NAME} (\text{DEPOSITOR})$

### Output:

CUSTOMER\_NAME

Johnson Smith Hayes Turner Jones Lindsay Jackson Curry Williams Mayes

### Set Intersection

Suppose there are two tuples R and S. The set intersection operation contains all tuples that are in both R & S.

It is denoted by intersection  $\cap$ . Notation:  $R \cap S$

Example: Using the above DEPOSITOR table and BORROW table

### Input:

$\prod \text{CUSTOMER\_NAME} (\text{BORROW}) \cap \prod \text{CUSTOMER\_NAME} (\text{DEPOSITOR})$

### Output:

CUSTOMER\_NAME

Smith Jones

## Set Difference

Suppose there are two tuples R and S. The set intersection operation contains all tuples that are in R but not in S.

It is denoted by intersection minus (-). Notation: R - S

Example: Using the above DEPOSITOR table and BORROW table

### Input:

$\prod \text{CUSTOMER\_NAME (BORROW)} - \prod \text{CUSTOMER\_NAME (DEPOSITOR)}$

### Output:

CUSTOMER\_NAME

Jackson Hayes Willians Curry

## Cartesian Product

The Cartesian product is used to combine each row in one table with each row in the other table. It is also known as a cross product.

It is denoted by X.

Notation: E X D

Example: EMPLOYEE

EMP_ID	EMP_NAME	EMP_DEPT
1	Smith	A
2	Harry	C
3	John	B

## DEPARTMENT

DEPT_NO	DEPT_NAME
A	Marketing
B	Sales
C	Legal

### Input:

EMPLOYEE X DEPARTMENT

### Output:

EMP_ID	EMP_NAME	EMP_DEPT	DEPT_NO	DEPT_NAME
1	Smith	A	A	Marketing
1	Smith	A	B	Sales
1	Smith	A	C	Legal
2	Harry	C	A	Marketing
2	Harry	C	B	Sales
2	Harry	C	C	Legal
3	John	B	A	Marketing
3	John	B	B	Sales
3	John	B	C	Legal

## Rename Operation

The rename operation is used to rename the output relation. It is denoted by rho ( $\rho$ ). Example: We can use the rename operator to rename STUDENT relation to STUDENT1.

$\rho(\text{STUDENT1}, \text{STUDENT})$

## Join Operation

Join operation is essentially a Cartesian product followed by a selection criterion. Join operation denoted by  $\bowtie$ .

JOIN operation also allows joining variously related tuples from different relations.

### Inner Join

Natural join between two or more relations will result in all the combination of tuples where they have equal values for the common attribute

### Theta Join

The general case of JOIN operation is called a Theta join. It is denoted by symbol  $\theta$

For example:

$A \bowtie A.\text{column 2} > B.\text{column 2} (B)$

### EQUI Join

When a theta join uses only equivalence condition, it becomes a equi join.

For example:

$A \bowtie A.\text{column 2} = B.\text{column 2} (B)$

Example

$A \bowtie \theta B$  Theta join can use any conditions in the selection criteria.

### Left Outer Join

In the left outer join, operation allows keeping all tuple in the left relation. However, if there is no matching tuple is found in right relation, then the attributes of right relation in the join result are filled with null values.

Example:  $A \bowtie B$

Example: Select students whose ROLL\_NO is greater than EMP\_NO of employees and detail of other students as well

---

STUDENT  $\bowtie$  STUDENT.ROLL\_NO > EMPLOYEE.EMP\_NO EMPLOYEE

### Right Outer Join

In the left outer join, operation allows keeping all tuple in the left relation. However, if there is no matching tuple is found in right relation, then the attributes of right relation in the join result are filled with null values.

Example: Select students whose ROLL\_NO is greater than EMP\_NO of employees and detailsof other Employees as well

STUDENT  $\bowtie$  STUDENT.ROLL\_NO > EMPLOYEE.EMP\_NO EMPLOYEE

### Full Outer Join

In the left outer join, operation allows keeping all tuple in the left relation. However, if there is no matching tuple is found in right relation, then the attributes of right relation in the join result are filled with null values.

Example: Select students whose ROLL\_NO is greater than EMP\_NO of employees and detailsof other Employees as well and other Students as well

STUDENT  $\bowtie$  STUDENT.ROLL\_NO > EMPLOYEE.EMP\_NO EMPLOYEE

### Division Operator

Division operator  $A \div B$  can be applied if and only if:

Attributes of B is proper subset of Attributes of A.

The relation returned by division operator will have attributes = (All attributes of A – All Attributes of B)

The relation returned by division operator will return those tuples from relation A which are associated to every B's tuple.

Consider the relation STUDENT\_SPORTS and ALL\_SPORTS given in Table 2 and Table 3 above.

To apply division operator as **STUDENT\_SPORTS  $\div$  ALL\_SPORTS**

## Relational Calculus

### Introduction

Relational calculus is a non-procedural query language, and instead of algebra, it uses mathematical predicate calculus.

Tuple relational calculus which was originally proposed by Codd in the year 1972 and Domain relational calculus which was proposed by Lacroix and Pirotte in the year 1977

Relational calculus is a non-procedural query language. In the non-procedural query language, the user is concerned with the details of how to obtain the end results.

The relational calculus tells what to do but never explains how to do.

### Types of Relational Calculus

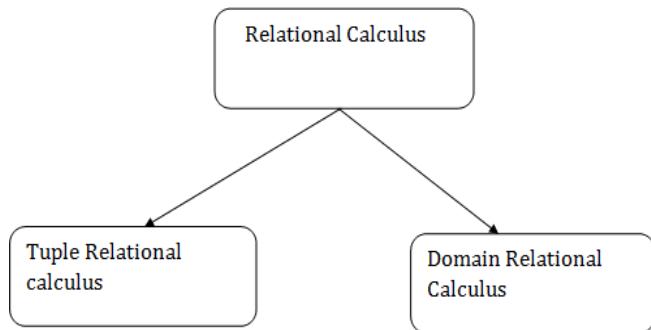


Image 45 – Relational Calculus

Reference - <https://static.javatpoint.com/dbms/images/dbms-relational-calculus.png>

### Tuple Relational Calculus

In tuple relational calculus, we work on filtering tuples based on the given condition.

Syntax: { T | Condition }

In domain relational calculus, filtering is done based on the domain of the attributes and not based on the tuple values.

Syntax: { c1, c2, c3, ..., cn | F(c1, c2, c3, ..., cn) }

The tuple relational calculus is specified to select the tuples in a relation. In TRC, filtering variable uses the tuples of a relation.

The result of the relation can have one or more tuples. Notation:

$\{T \mid P(T)\}$  or  $\{T \mid \text{Condition}(T)\}$  Where

T is the resulting tuples

$P(T)$  is the condition used to fetch T. For example:

$\{ T.name \mid \text{Author}(T) \text{ AND } T.article = \text{'database'} \}$

**OUTPUT:** This query selects the tuples from the AUTHOR relation. It returns a tuple with 'name' from Author who has written an article on 'database'.

The symbols used for logical operators are:  $\wedge$  for AND,  $\vee$  for OR and  $\neg$  for NOT. Variables can be constrained by quantified statements to tuples in a single relation: –

**Existential Quantifier.**  $\exists T \in R(\text{Cond})$  will succeed if Cond succeeds for at least one tuple in T.

**Universal Quantifier.**  $\forall T \in R(\text{Cond})$  will succeed if Cond succeeds for all tuples in T

### Examples

$\{t \mid P(t)\}$  or  $\{t \mid \text{condition}(t)\}$  —

this is also known as expression of relational calculus

Where t is the resulting tuples,  $P(t)$  is the condition used to fetch t.

$\{t \mid \text{EMPLOYEE}(t) \text{ and } t.SALARY > 10000\}$  –

implies that it selects the tuples from EMPLOYEE relation such that resulting employee tuples will have salary greater than 10000. It is example of selecting a range of values.

$\{t \mid \text{EMPLOYEE}(t) \text{ AND } t.DEPT\_ID = 10\}$  –

this select all the tuples of employee name who work for Department 10.

$\{ T.name \mid \text{FACULTY}(T) \text{ AND } T.DeptId = 0 \text{ CS0 } \}$

can be read as: “Find all tuples T field such that T is a tuple in the FACULTY relation and the value of DeptId field is ‘CS’. Return a tuple with a single field name which is equivalent to the name field of one such T tuple”.

$\{R \mid \exists T \in \text{FACULTY} (T.\text{DeptId} = 0 \text{CS0 AND } R.\text{name} = T.\text{name})\}$

can be read as: “Find all tuples R such that there exists a tuple T in FACULTY with the DeptId field value ‘CS’, and the value of the name field of R is equivalent to the name field of this tuple T.”

Example 1 Find the loan number, branch, amount of loans of greater than or equal to 10000 amount.

$\{t \mid t \in \text{loan} \wedge t[\text{amount}] \geq 10000\}$

Example 2 Find the loan number for each loan of an amount greater or equal to 10000.

$\{t \mid \exists s \in \text{loan} (t[\text{loan number}] = s[\text{loan number}] \wedge s[\text{amount}] \geq 10000)\}$

Example 3 Find the names of all customers who have a loan and an account at the bank.

$\{t \mid \exists s \in \text{borrower} (t[\text{customer-name}] = s[\text{customer-name}] \wedge \exists u \in \text{depositor} (t[\text{customer-name}] = u[\text{customer-name}]))\}$

### Domain Relational Calculus

In domain relational calculus, filtering variable uses the domain of attributes.

Domain relational calculus uses the same operators as tuple calculus. It uses logical connectives  $\wedge$  (and),  $\vee$  (or) and  $\neg$  (not).

It uses Existential ( $\exists$ ) and Universal Quantifiers ( $\forall$ ) to bind the variable.

The domain variables those will be in resulting relation must appear before  $\mid$  within  $\prec$  and  $\succ$  and all the domain variables must appear in which order they are in original relation or table..

Notation:

$$\{ a1, a2, a3, \dots, an \mid P (a1, a2, a3, \dots, an) \}$$

Were,

a1, a2 are attributes

P stands for formula built by inner attributes

Example 1

**Find the loan number, branch, amount of loans of greater than or equal to 100 amount.**

$$\{ \langle l, b, a \rangle \mid \langle l, b, a \rangle \in \text{loan} \wedge (a \geq 100) \}$$

Example 2

**Find the loan number for each loan of an amount greater or equal to 150.**

$$\{ \langle l \rangle \mid \exists b, a \ (\langle l, b, a \rangle \in \text{loan} \wedge (a \geq 150)) \}$$

Example 3

**Find the names of all customers having a loan at the “Main” branch and find the loan amount.**

$$\{ \langle c, a \rangle \mid \exists l \ (\langle c, l \rangle \in \text{borrower} \wedge \exists b \ (\langle l, b, a \rangle \in \text{loan} \wedge (b = \text{“Main”}))) \}$$

Example 4

$$\{ \langle \text{name}, \text{age} \rangle \mid \in \text{Student} \wedge \text{age} < 21 \}$$

Again, the above query will return the names and ages of the students in the table Student who are not greater than 21 years old

Example 5

$$\{ \langle \text{Fname}, \text{Emp_ID} \rangle \mid \in \text{Employee} \wedge \text{Salary} > 10000 \}$$

The result here will be returning the Fname and Emp\_ID values for all the rows in the employee table where salary is greater than 10000.

## RDBMS Technologies

### Oracle Database Technology Features

- Data Concurrency and Consistency
- Manageability
- Backup & Recovery
- Business Intelligence
- High Availability
- Very Large Databases
- Content Management

### MySQL Database Technology Features

- Data Concurrency and Consistency
- Scalability and Limit
- Backup & Recovery
- Connectivity
- High Availability
- Clients and Tools
- Workbench tool

### MongoDB Database Technology Features

- Indexing
- Replication
- Backup & Recovery
- Load Balancing
- Map Reducing and Aggregation
- Stores files of any size easily without complicating your stack.
- Cloud Support

### Microsoft SQL Server Database Technology Features

- Highest performing data warehouses
- End to End Mobile BI
- Backup & Recovery
- Load Balancing
- Built-in Analytics
- Mission Critical Availability
- Cloud Support

## Relational Data Structure/Relational Model

The **relational model (RM)** for database management is an approach to managing data using a structure and language consistent with first-order-predicate logic, first described in 1969 by English computer scientist Edgar F Codd, where all data is represented in terms of tuples, grouped into relations. A database organized in terms of the relational model is a relational database.

**Note:** First-order logic—also known as predicate logic, is a collection of formal systems used in mathematics, philosophy, linguistics, and computer science.

### What is a Relational Model?

**RELATIONAL MODEL (RM)** represents the database as a collection of relations. A relation is nothing but a table of values. Every row in the table represents a collection of related data values. These rows in the table denote a real-world entity or relationship.

- The table name and column names are helpful to interpret the meaning of values in each row.
- The data are represented as a set of relations.
- In the relational model, data is stored as tables.
- However, the physical storage of the data is independent of the way the data are logically organized.

Some popular Relational Database management systems are:

- DB2 and Informix Dynamic Server - IBM
- Oracle and RDB – Oracle
- SQL Server and Access - Microsoft

### Relational Model Concepts

1. **Attribute:** Each column in a Table. Attributes are the properties which define a relation.  
e.g., Student\_Rollno, NAME,etc.
2. **Tables:** In the Relational model the, relations are saved in the table format. It is stored along with its entities. A table has two properties: rows and columns. Rows represent records and columns represent attributes.
3. **Tuple:** It is nothing but a single row of a table, which contains a single record.
4. **Relation Schema:** A relation schema represents the name of the relation with its attributes.
5. **Degree:** The total number of attributes which in the relation is called the degree of the relation.
6. **Cardinality:** Total number of rows present in the Table.

7. **Column:** The column represents the set of values for a specific attribute.
8. **Relation instance:** Relation instance is a finite set of tuples in the RDBMS system. Relation instances never have duplicate tuples.
9. **Relation key:** Every row has one, two or multiple attributes, which is called relation key.
10. **Attribute domain:** Every attribute has some predefined value and scope which is known as attribute domain

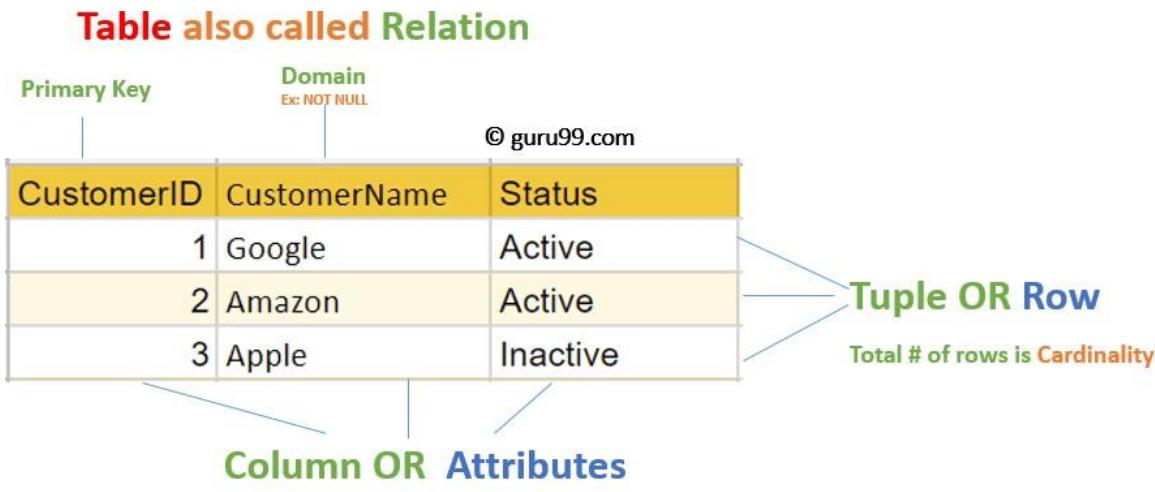


Image 1- Shows some of the Relational Model Concepts

Reference- <https://www.guru99.com/relational-data-model-dbms.html>

## Relational Integrity constraints

Relational Integrity constraints are referred to conditions which must be present for a valid relation. These integrity constraints are derived from the rules in the mini-world that the database represents.

- Integrity constraints are a set of rules. It is used to maintain the quality of information.
- Integrity constraints ensure that the data insertion, updating, and other processes have to be performed in such a way that data integrity is not affected.
- Thus, integrity constraint is used to guard against accidental damage to the database.

## Types of integrity constraints

1. Domain constraints
2. Entity Integrity constraints
3. Referential integrity constraints
4. Key constraints

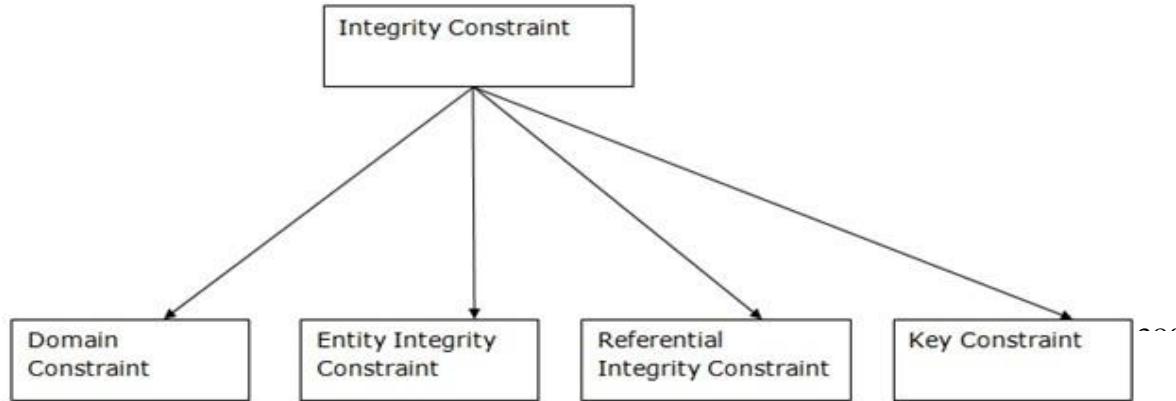


Image 2- Types of integrity constraints

Reference- <https://www.javatpoint.com/dbms-integrity-constraints>

### 1. Domain constraints

- Domain constraints can be defined as the definition of a valid set of values for an attribute.
- The data type of domain includes string, character, integer, time, date, currency, etc. The value of the attribute must be available in the corresponding domain.

Example:

ID	NAME	SEMESTER	AGE
1000	Tom	1 <sup>st</sup>	17
1001	Johnson	2 <sup>nd</sup>	24
1002	Leonardo	5 <sup>th</sup>	21
1003	Kate	3 <sup>rd</sup>	19
1004	Morgan	8 <sup>th</sup>	A

Not allowed. Because AGE is an integer attribute

Image 3- Example of Domain Constraint

Reference- <https://www.javatpoint.com/dbms-integrity-constraints>

## 2. Entity integrity constraints

- The entity integrity constraint states that primary key value can't be null.
- This is because the primary key value is used to identify individual rows in relation and if the primary key has a null value, then we can't identify those rows.
- A table can contain a null value other than the primary key field.

Example:

### EMPLOYEE

EMP_ID	EMP_NAME	SALARY
123	Jack	30000
142	Harry	60000
164	John	20000
	Jackson	27000

Not allowed as primary key can't contain a NULL value

Image 4- Example of Entity integrity constraint

Reference- <https://www.javatpoint.com/dbms-integrity-constraints>

### 3. Referential Integrity Constraints

- A referential integrity constraint is specified between two tables.
- In the Referential integrity constraints, if a foreign key in Table 1 refers to the PrimaryKey of Table 2, then every value of the Foreign Key in Table 1 must be null or be available in Table 2.

Example:

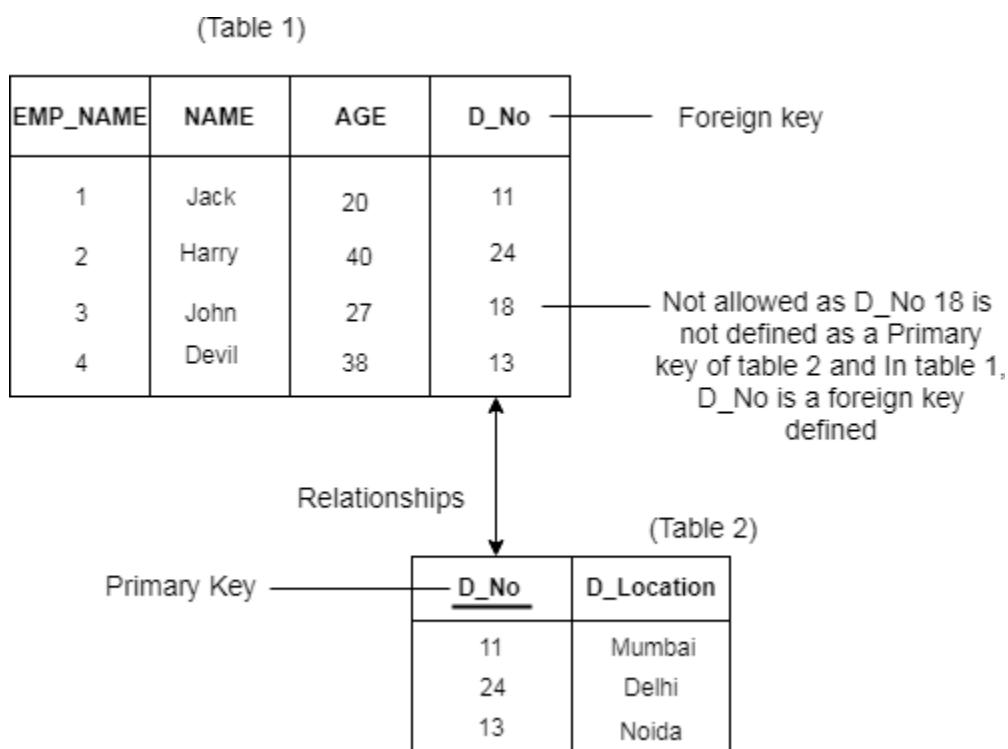


Image 5- referential integrity constraint

Reference- <https://www.javatpoint.com/dbms-integrity-constraints>

### 4. Key constraints

- Keys are the entity set that is used to identify an entity within its entity set uniquely.
- An entity set can have multiple keys, but out of which one key will be the primary key. A primary key can contain a unique and null value in the relational table.

Example:

ID	NAME	SEMENSTER	AGE
1000	Tom	1 <sup>st</sup>	17
1001	Johnson	2 <sup>nd</sup>	24
1002	Leonardo	5 <sup>th</sup>	21
1003	Kate	3 <sup>rd</sup>	19
1002	Morgan	8 <sup>th</sup>	22

Not allowed. Because all row must be unique

Image 6- Example of Key Constraint

Reference- <https://www.javatpoint.com/dbms-integrity-constraints>

## Operations in the Relational Model

Four basic update operations performed on the relational database model are Insert, update, delete and select.

- Insert is used to insert data into the relation
- Delete is used to delete tuples from the table.
- Modify allows you to change the values of some attributes in existing tuples.
- Select allows you to choose a specific range of data.

Whenever one of these operations are applied, integrity constraints specified on the relational database schema must never be violated.

### Insert Operation:

The insert operation gives values of the attribute for a new tuple which should be inserted into a relation.



CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive
4	Alibaba	Active

Image 7- Insert

Reference- <https://www.guru99.com/relational-data-model-dbms.html>

## Update Operation

You can see that in the below-given relation table CustomerName= 'Apple' is updated from Inactive to Active.



CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive
4	Alibaba	Active

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Active
4	Alibaba	Active

Image 8- Update

Reference- <https://www.guru99.com/relational-data-model-dbms.html>

## Delete Operation

To specify deletion, a condition on the attributes of the relation selects the tuple to be deleted.



CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Active
4	Alibaba	Active

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
4	Alibaba	Active

Image 9- Delete

Reference- <https://www.guru99.com/relational-data-model-dbms.html>

In the above-given example, CustomerName= "Apple" is deleted from the table.

The Delete operation could violate referential integrity if the tuple which is deleted is referenced by foreign keys from other tuples in the same database.

## Select Operation



CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
4	Alibaba	Active

CustomerID	CustomerName	Status
2	Amazon	Active

Image 10- Select

Reference- <https://www.guru99.com/relational-data-model-dbms.html>

In the above-given example, CustomerName="Amazon" is selected.

## Advantages of using the Relational model

- **Simplicity:** A relational data model is simpler than the hierarchical and network model.
- **Structural Independence:** The relational database is only concerned with data and not with a structure. This can improve the performance of the model.
- **Easy to use:** The relational model is easy as tables consisting of rows and columns are quite natural and simple to understand.
- **Query capability:** It makes possible for a high-level query language like SQL to avoid complex database navigation.
- **Data independence:** The structure of a database can be changed without having to change any application.
- **Scalable:** Regarding a number of records, or rows, and the number of fields, a database should be enlarged to enhance its usability.

## Disadvantages of using Relational model

- Few relational databases have limits on field lengths which can't be exceeded.
- Relational databases can sometimes become complex as the amount of data grows, and the relations between pieces of data become more complicated.
- Complex relational database systems may lead to isolated databases where the information cannot be shared from one system to another.

## Key and Relational Data Manipulation

### Keys

#### What are Keys?

A DBMS key is an attribute or set of an attribute which helps you to identify a row(tuple) in a relation(table). They allow you to find the relation between two tables. Keys help you uniquely identify a row in a table by a combination of one or more columns in that table.

#### Example:

Student ID	FirstName	LastName
1	Qanith	Khan
2	Rajesh	Singh
3	john	Hale

In the above-given example, Student ID is a primary key because it uniquely identifies an Student record. In this table, no other Student can have the same Student ID.

#### Why do we need a Key?

Here are reasons for using Keys in the DBMS system.

- Keys help you to identify any row of data in a table. In a real-world application, a table could contain thousands of records. Moreover, the records could be duplicated. Keys ensure that you can uniquely identify a table record despite these challenges.
- Allows you to establish a relationship between and identify the relation between tables
- Help you to enforce identity and integrity in the relationship.

## Various Keys

Each keys have their different functionality:

- 
- Super Key
  - Primary Key
  - Candidate Key
  - Alternate Key
  - Foreign Key
  - Compound Key
  - Composite Key
  - Surrogate Key

Super key:

A superkey is a group of single or multiple keys which identifies rows in a table. A Super key may have additional attributes that are not needed for unique identification.

Example:

<b>EmpSSN</b>	<b>EmpNum</b>	<b>Empname</b>
9812345098	AB05	Shown
9876512345	AB06	Roslyn
9199937890	AB07	James

In the above-given example, EmpSSN and EmpNum name are superkeys.

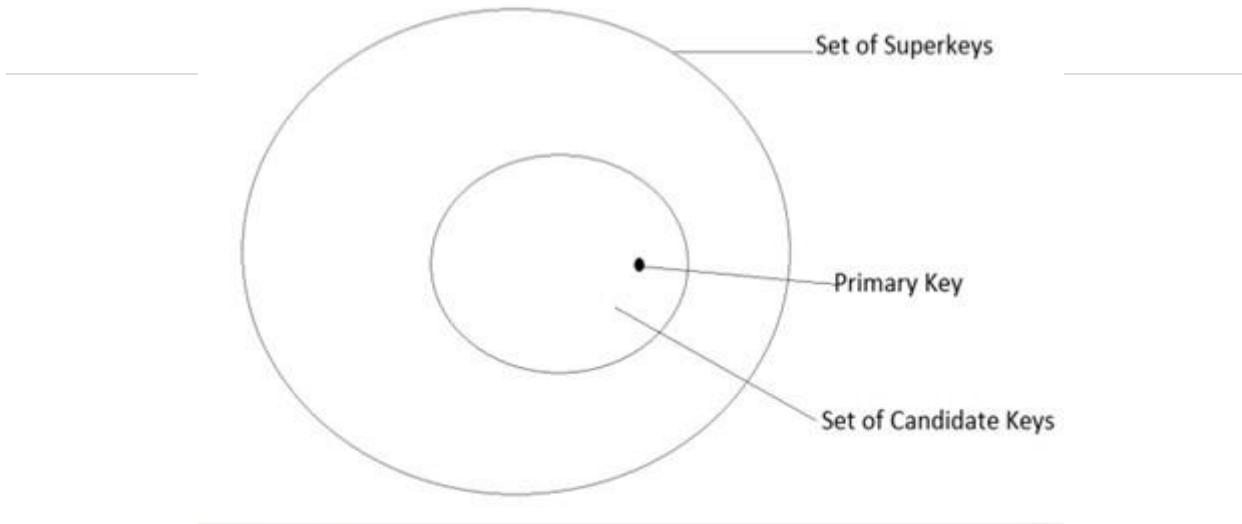


Image 11- Relation b/w keys

Reference-

[https://www.slideshare.net/TechtudNetwork/relation-between-super-key-candidate-key-and-primary-key?qid=c7bb9eac-8135-480c-92ee-890bf4c744aa&v=&b=&from\\_search=1](https://www.slideshare.net/TechtudNetwork/relation-between-super-key-candidate-key-and-primary-key?qid=c7bb9eac-8135-480c-92ee-890bf4c744aa&v=&b=&from_search=1)

## Primary Key

**PRIMARY KEY** is a column or group of columns in a table that uniquely identify every row in that table. The Primary Key can't be a duplicate meaning the same value can't appear more than once in the table. A table cannot have more than one primary key.

Rules for defining Primary key:

- Two rows can't have the same primary key value
- It must for every row to have a primary key value.
- The primary key field cannot be null.
- The value in a primary key column can never be modified or updated if any foreign key refers to that primary key.

Example:

In the following example, `StudID` is a Primary Key.

StudID	Roll No	First Name	LastName	Email
1	11	Tom	Price	abc@gmail.com
2	12	Nick	Wright	xyz@gmail.com
3	13	Dana	Natan	mno@yahoo.com

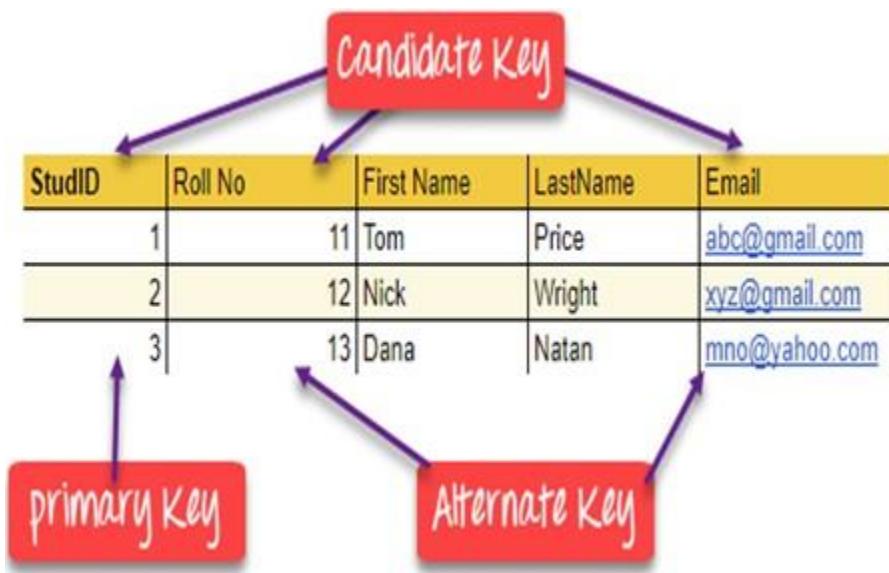


Image 12- Primary Key

### Alternate key

**ALTERNATE KEYS** is a column or group of columns in a table that uniquely identify every row in that table. A table can have multiple choices for a primary key but only one can be set as the primary key. All the keys which are not primary key are called an Alternate Key.

Example:

In this table, StudID, Roll No, Email are qualified to become a primary key. But since StudID is the primary key, Roll No, Email becomes the alternative key.

StudID	Roll No	First Name	LastName	Email
1	11	Tom	Price	abc@gmail.com
2	12	Nick	Wright	xyz@gmail.com
3	13	Dana	Natan	mno@yahoo.com

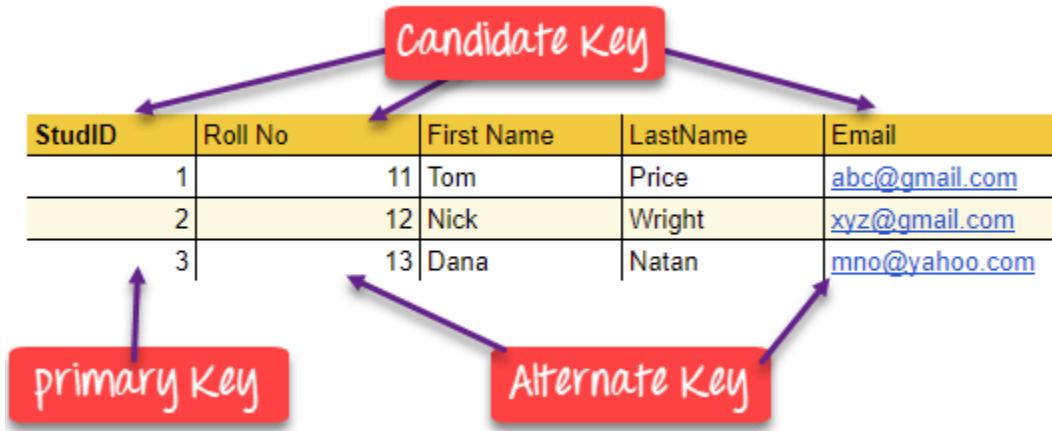


Image 13- Alternate Key

## Candidate Key

**CANDIDATE KEY** is a set of attributes that uniquely identify tuples in a table. Candidate Key is a super key with no repeated attributes. The Primary key should be selected from the candidate keys. Every table must have at least a single candidate key. A table can have multiple candidate keys but only a single primary key.

### Properties of Candidate key:

- It must contain unique values
- Candidate key may have multiple attributes
- Must not contain null values
- It should contain minimum fields to ensure uniqueness
- Uniquely identify each record in a table

Example: In the given table Stud ID, Roll No, and email are candidate keys which help us to uniquely identify the student record in the table.

StudID	Roll No	First Name	LastName	Email
1	11	Tom	Price	abc@gmail.com
2	12	Nick	Wright	xyz@gmail.com
3	13	Dana	Natan	mno@yahoo.com

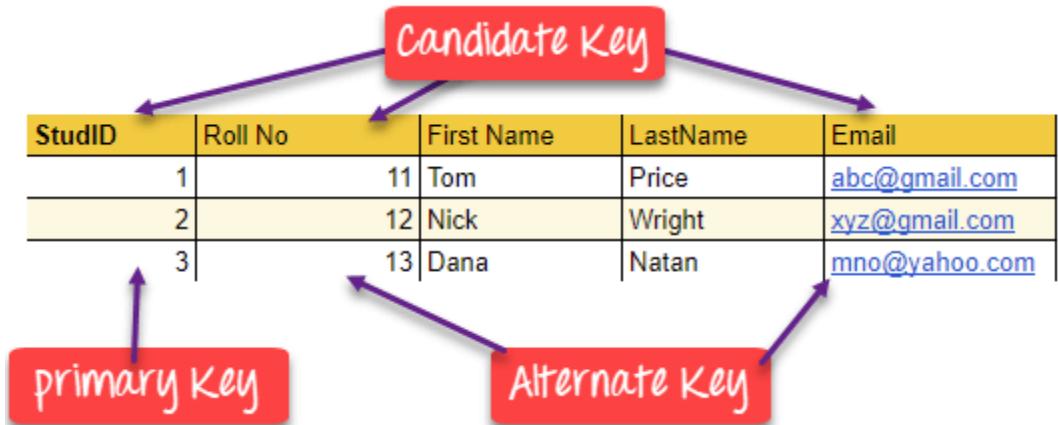


Image 14- Candidate Key

Reference-<https://www.guru99.com/dbms-keys.html>

## Foreign key

**FOREIGN KEY** is a column that creates a relationship between two tables. The purpose of foreign keys is to maintain data integrity and allow navigation between two different instances of an entity. It acts as a cross-reference between two tables as it references the primary key of another table.

Example:

<b>DeptCode</b>	<b>DeptName</b>
001	Science
002	English
005	Computer

<b>Teacher ID</b>	<b>Fname</b>	<b>Lname</b>
B002	David	Warner
B017	Sara	Joseph
B009	Mike	Brunton

In this example, we have two table, teach and department in a school. However, there is no way to see which teacher works in which department.

In this table, adding the foreign key in Deptcode to the Teacher name, we can create a relationship between the two tables.

<b>Teacher ID</b>	<b>DeptCode</b>	<b>Fname</b>	<b>Lname</b>
B002	002	David	Warner
B017	002	Sara	Joseph
B009	001	Mike	Brunton

This concept is also known as Referential Integrity.

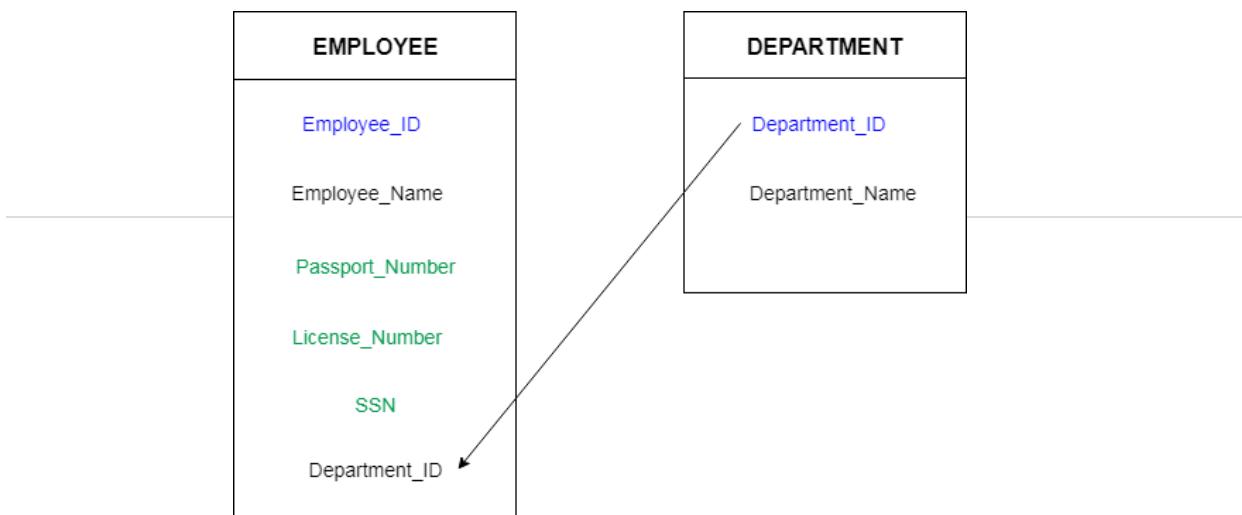


Image 15- Foreign key

Reference- <https://www.javatpoint.com/dbms-keys>

## Compound key

**COMPOUND KEY** has two or more attributes that allow you to uniquely recognize a specific record. It is possible that each column may not be unique by itself within the database. However, when combined with the other column or columns the combination of composite keys become unique. The purpose of compound key is to uniquely identify each record in the table.

Example:

OrderNo	ProductID	Product Name	Quantity
B005	JAP102459	Mouse	5
B005	DKT321573	USB	10
B005	OMG446789	LCD Monitor	20
B004	DKT321573	USB	15
B002	OMG446789	Laser Printer	3

In this example, OrderNo and ProductID can't be a primary key as it does not uniquely identify a record. However, a compound key of Order ID and Product ID could be used as it uniquely identified each record.

### Composite key

**COMPOSITE KEY** is a combination of two or more columns that uniquely identify rows in a table. The combination of columns guarantees uniqueness, though individually uniqueness is not guaranteed. Hence, they are combined to uniquely identify records in a table.

The difference between compound and the composite key is that any part of the compound key can be a foreign key, but the composite key may or maybe not a part of the foreign key.

### Surrogate Key

An artificial key which aims to uniquely identify each record is called a surrogate key. These kinds of key are unique because they are created when you don't have any natural primary key.

Fnam e	Lastnam e	Start Time	End Time
Anne	Smith	09:00	18:00
Jack	Francis	08:00	17:00
Anna	McLean	11:00	20:00
Shown	Willam	14:00	23:00

They do not lend any meaning to the data in the table. Surrogate key is usually an integer.

Above, given example, shown shift timings of the different employee. In this example, a surrogate key is needed to uniquely identify each employee.

Surrogate keys are allowed when

- No property has the parameter of the primary key.
- In the table when the primary key is too big or complicated.

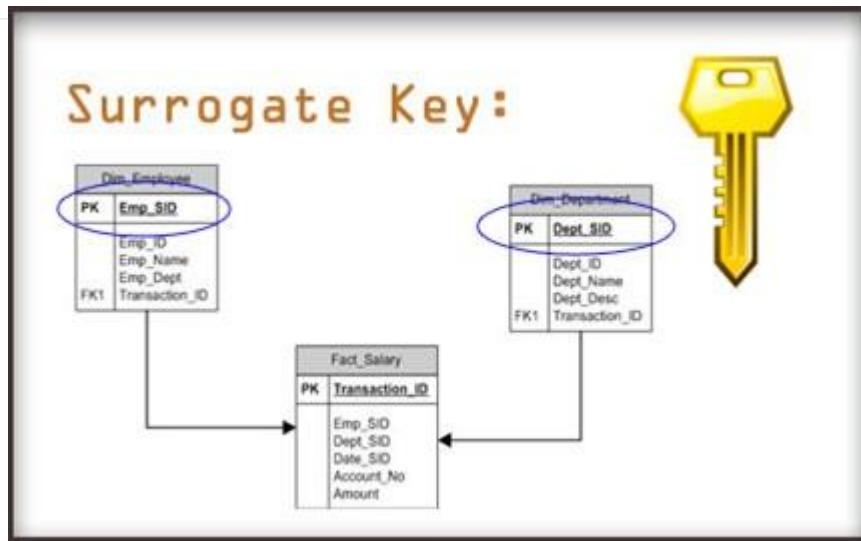


Image 16- Surrogate Key

Reference-

[https://www.google.com/search?q=Surrogate+Key+&tbm=isch&ved=2ahUKEwiX99ycj-roAhUI5DgGHffQBnQQ2-cCegQIABAA&oq=Surrogate+Key+&gs\\_lcp=CgNpbWcQAzIECAAQzICCAAyAggAMgIIADICCAAyAggAMgIIADICCAAyAggAMgIIADFD-nBpY\\_pwaYJuhGmgAcA\\_B4IAIBswGIAbMBkgEDMC4xmAEAoAEBqgELZ3dzLXdpei1pbWc&client=img&ei=89OWXtexAojl4-EP96GboAc&bih=657&biw=1366#imgrc=JViJUIE4gOGs0M](https://www.google.com/search?q=Surrogate+Key+&tbm=isch&ved=2ahUKEwiX99ycj-roAhUI5DgGHffQBnQQ2-cCegQIABAA&oq=Surrogate+Key+&gs_lcp=CgNpbWcQAzIECAAQzICCAAyAggAMgIIADICCAAyAggAMgIIADICCAAyAggAMgIIADFD-nBpY_pwaYJuhGmgAcA_B4IAIBswGIAbMBkgEDMC4xmAEAoAEBqgELZ3dzLXdpei1pbWc&client=img&ei=89OWXtexAojl4-EP96GboAc&bih=657&biw=1366#imgrc=JViJUIE4gOGs0M)

Let's see the difference between keys Primary Key and Foreign Key:

S.NO.	PRIMARY KEY	FOREIGN KEY
1	A primary key is used to ensure data in the specific column is unique.	A foreign key is a column or group of columns in a relational database table that provides a link between data in two tables.
2	It uniquely identifies a record in the relational database table.	It refers to the field in a table which is the primary key of another table.

<b>3</b>	Only one primary key is allowed in a table.	Whereas more than one foreign key are allowed in a table.
<b>4</b>	It is a combination of UNIQUE and Not Null constraints.	It can contain duplicate values and a table in a relational database.
<b>5</b>	It does not allow NULL values.	It can also contain NULL values.
<b>6</b>	Its value cannot be deleted from the parent table.	Its value can be deleted from the child table.
<b>7</b>	It constraint can be implicitly defined on the temporary tables.	It constraint cannot be defined on the local or global temporary tables.

### Primary and Candidate Key:

<b>S.NO</b>	<b>PRIMARY KEY</b>	<b>CANDIDATE KEY</b>
<b>1</b>	Primary key is a minimal super key. So there is one and only one primary key in a relation.	While in a relation there can be more than one candidate key.
<b>2</b>	Any attribute of Primary key can not contain NULL value.	While in Candidate key any attribute can contain NULL value.
<b>3</b>	Primary key can be optional to specify any relation.	But without candidate key there can't be specified any relation.
<b>4</b>	Primary key specifies the important attribute for the relation.	Candidate specifies the key which can qualify for primary key.
<b>5</b>	It's confirmed that a primary key is a candidate key.	But it's confirmed that a candidate key can be a primary key.

## Super Key and Candidate Key

S.NO	<b>SUPER KEY</b>	<b>CANDIDATE KEY</b>
<b>1</b>	Super Key is an attribute (or set of attributes) that is used to uniquely identifies all attributes in a relation.	Candidate Key is a proper subset of a super key.
<b>2</b>	All super keys can't be candidate keys.	But all candidate keys are super keys.
<b>3</b>	Various super keys together make the criteria to select the candidate keys.	Various candidate keys together make the criteria to select the primary keys.
<b>4</b>	In a relation, number of super keys are more than number of candidate keys.	While in a relation, number of candidate keys are less than number of super keys.
<b>5</b>	Super key's attributes can contain NULL values.	Candidate key's attributes can also contain NULL values.

## Primary key and Unique key

<b>PARAMETER</b>	<b>PRIMARY KEY</b>	<b>UNIQUE KEY</b>
Basic	Used to serve as a unique identifier for each row in a table.	Uniquely determines a row which isn't primary key.
NULL value acceptance	Cannot accept NULL values.	Can accept one NULL value.
Number of keys that can be defined in the table	Only one primary key	More than one unique key
Index	Creates clustered index	Creates non-clustered index

## Data Manipulation Language (DML)

### Introduction to DML

- DML stands for **Data Manipulation Language**.
- It is a language used for selecting, inserting, deleting and updating data in a database.
- It is used to retrieve and manipulate data in a relational database.

DDL commands are as follows:

1. SELECT
2. INSERT
3. UPDATE
4. DELETE

DML performs read-only queries of data.

### 1. SELECT COMMAND

- **SELECT command** is used to retrieve data from the database.
- This command allows database users to retrieve the specific information they desire from an operational database.
- It returns a result set of records from one or more tables.

SELECT Command has many optional clauses are as stated below:

Clause	Description
WHERE	It specifies which rows to retrieve.

GROUP BY	It is used to arrange the data into groups.
HAVING	It selects among the groups defined by the GROUP BY clause.
ORDER BY	It specifies an order in which to return the rows.
AS	It provides an alias which can be used to temporarily rename tables or columns.

**Syntax:**

```
SELECT * FROM <table_name>;
```

**Example:** Select Command

```
SELECT * FROM employee;OR
```

```
SELECT * FROM employee where salary >=10,000;
```

## 2. INSERT COMMAND

- **INSERT command** is used for inserting data into a table.
- Using this command, you can add one or more records to any single table in a database.
- It is also used to add records to an existing code.

**Syntax:**

```
INSERT INTO <table_name>(`column_name1` <datatype>, `column_name2` <datatype>, . . . ,  
`column_name_n` <datatype>) VALUES(`value1`, `value2`, . . . , `value n`);
```

**Example:**

```
INSERT INTO employee(`eid` int, `ename` varchar(20),  
`city` varchar(20))VALUES(`1`, `ABC`, `PUNE`);
```

### 3. UPDATE COMMAND

- **UPDATE command** is used to modify the records present in existing table.
- This command updates existing data within a table.
- It changes the data of one or more records in a table.

#### Syntax:

```
UPDATE <table_name> SET <column_name = value> WHERE condition;
```

#### Example: Update Command

```
UPDATE employee SET salary=20000 WHERE ename='ABC';
```

### 4. DELETE COMMAND

- **DELETE command** is used to delete some or all records from the existing table.
- It deletes all the records from a table.

#### Syntax:

```
DELETE FROM <table_name> WHERE <condition>;
```

#### Example: Delete Command

```
DELETE FROM employee WHERE emp_id = '001';
```

If we does not write the WHERE condition, then all rows will get deleted.

## Relational Algebra

### Relational Algebra

**RELATIONAL ALGEBRA** is a widely used procedural query language. It collects instances of relations as input and gives occurrences of relations as output. It uses various operations to perform this action. Relational algebra operations are performed recursively on a relation. The output of these operations is a new relation, which might be formed from one or more input relations.

- First created by Edgar F. Codd
- The main application of relational algebraic providing a theoretical foundation for relational databases, particularly query languages for such databases, chief among which is SQL.

## Relational Algebraic Operations

Relational Algebra divides in various groups

1. Unary Relational Operations
  - 1.1 SELECT (symbol:  $\sigma$ )
  - 1.2 PROJECT (symbol:  $\pi$ )
  - 1.3 RENAME (symbol:  $\rho$ )
2. Relational Algebra Operations From Set Theory
  - 2.1 UNION ( $\cup$ )
  - 2.2 INTERSECTION ( $\cap$ ),
  - 2.3 DIFFERENCE (-)
  - 2.4 CARTESIAN PRODUCT ( $\times$ )
3. Binary Relational Operations
  - 3.1 JOIN

## Unary Relational Operations

### SELECT ( $\sigma$ )

The SELECT operation is used for selecting a subset of the tuples according to a given selection condition. Sigma( $\sigma$ ) symbol denotes it. It is used as an expression to choose tuples which meet the selection condition. Select operation selects tuples that satisfy a given predicate.

$\sigma p(r)$

$\sigma$  is the predicate

r stands for relation which is the name of the table p is prepositional logic

### Example 1

$\sigma \text{topic} = \text{"Database"} (\text{Tutorials})$

**Output** - Selects tuples from Tutorials where topic = 'Database'.

### Example 2

$\sigma \text{topic} = \text{"Database"} \text{ and } \text{author} = \text{"Edunet"} (\text{Tutorials})$

**Output** - Selects tuples from Tutorials where the topic is 'Database' and 'author' is Edunet.

### Example 3

$\sigma \text{sales} > 50000 (\text{Customers})$

**Output** - Selects tuples from Customers where sales is greater than 50000

Projection( $\pi$ )

The projection eliminates all attributes of the input relation but those mentioned in the projection list. The projection method defines a relation that contains a vertical subset of Relation.

This helps to extract the values of specified attributes to eliminate duplicate values. ( $\pi$ ) The symbol used to choose attributes from a relation. This operation helps you to keep specific columns from a relation and discards the other columns.

### Example of Projection:

Consider the following table

CustomerID	CustomerName	Status

<b>1</b>	Google	<b>Active</b>
<b>2</b>	Amazon	<b>Active</b>
<b>3</b>	Apple	<b>Inactive</b>
<b>4</b>	Alibaba	<b>Active</b>

Here, the projection of CustomerName and status will give  $\Pi$  CustomerName, Status (Customers)

CustomerName	Status
Google	Active
Amazon	Active
Apple	Inactive
Alibaba	Active

$\text{Rename}(\rho)$

The results of relational algebra are also relations but without any name. The rename operation allows us to rename the output relation. 'rename' operation is denoted with small Greek letter rho  $\rho$ .

Notation –  $\rho$  x (E)

Where the result of expression E is saved with the name of x.

**Example:**  $\rho(\text{RelationNew}, \text{RelationOld})$

Relational Algebra Operations From Set Theory Union operation ( $\cup$ )

UNION is symbolized by  $\cup$  symbol. It includes all tuples that are in tables A or in B. It also eliminates duplicate tuples. So, set A UNION set B would be expressed as:

The result  $\leftarrow A \cup B$

For a union operation to be valid, the following conditions must hold -

- R and S must be the same number of attributes.
- Attribute domains need to be compatible.
- Duplicate tuples should be automatically removed.

**Example**

Consider the following tables.

Table A		Table B	
column 1	column 2	column 1	column 2
1	1	1	1
1	2	1	3

$A \cup B$  gives

<b>Table A <math>\cup</math> B</b>	
<b>column 1</b>	<b>column 2</b>
1	1
1	2
1	3

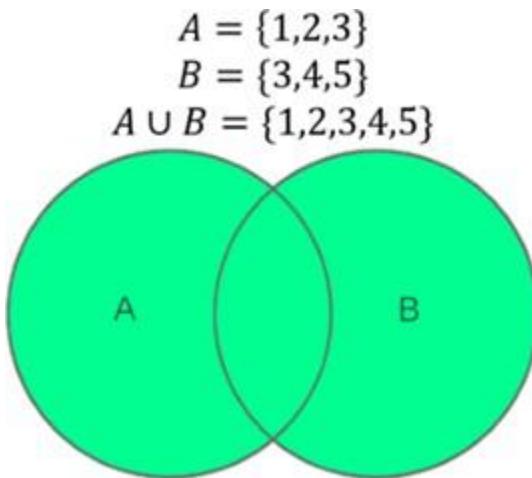


Image 17- Union Example

Reference- <https://www.codeproject.com/articles/1172312/just-enough-set-theory-when-sets-collide-part-of>

## Intersection

An intersection is defined by the symbol  $\cap$ .

$$A \cap B$$

Defines a relation consisting of a set of all tuple that are in both A and B. However, A and B must be union-compatible.

**Example:**  $A \cap B$

**Table  $A \cap B$**

**column 1**                    **column 2**

1	1
---	---

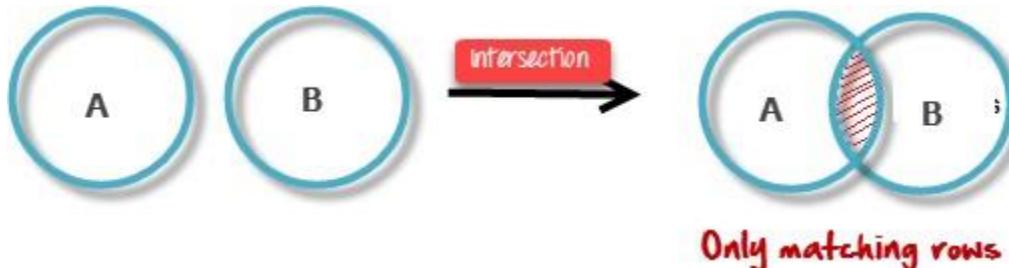


Image 18- Intersection Example

Reference- <https://www.guru99.com/relational-algebra-dbms.html>

### Set Difference (-)

- Symbol denotes it. The result of  $A - B$ , is a relation which includes all tuples that are in A but not in B.
  - The attribute name of A has to match with the attribute name in B.
  - The two-operand relations A and B should be either compatible or Union compatible.
  - It should be a defined relation consisting of the tuples that are in relation A, but not in B.

### Example

A-B

**Table A - B****column 1      column 2**

1                    2

$$\begin{aligned}A &= \{1,2,3\} \\B &= \{3,4,5\} \\B \setminus A &= \{4,5\}\end{aligned}$$

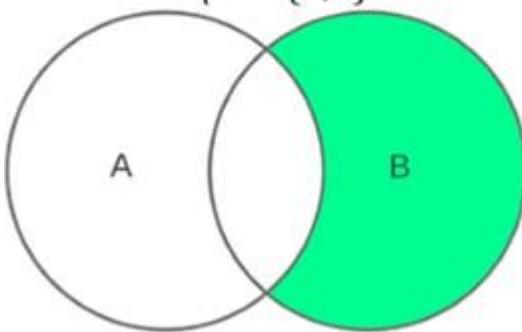


Image 19- Set Difference

**Cartesian product(X)**

This type of operation is helpful to merge columns from two relations. Generally, a Cartesian product is never a meaningful operation when it performs alone. However, it becomes meaningful when it is followed by other operations.

**Example – Cartesian product** $\sigma \text{ column 2} = '1' (A \times B)$ 

**Output** – The above example shows all rows from relation A and B whose column 2 has value 1

**σ column 2 = '1' (A X B)**

## column 1      column 2

1

A1(Name, Roll No)	
Name	Roll No
Anoop	1
Anurag	2

X

A2(Name, Roll No)	
Name	Roll No
Anoop	1
Anurag	2
Ganesh	3

Query Output(A1 X A2)			
Name	Roll No	Name	Roll No
Anoop	1	Anoop	1
Anoop	1	Anurag	2
Anoop	1	Ganesh	3
Anurag	2	Anoop	1
Anurag	2	Anurag	2
Anurag	2	Ganesh	3

## Image 20- Cartesian Product

Reference- <https://www.minigranth.com/dbms-tutorial/relational-algebra/>

## Binary Relational Operations

## JOIN

Join operation is essentially a cartesian product followed by a selection criterion. Join operation denoted by  $\bowtie$ .

JOIN operation also allows joining variously related tuples from different relations.

## Types of JOINS

Various forms of join operation are:

## 1. Inner Joins:

- Theta join
  - EQUI join
  - Natural join

2. Outer join:

- Left Outer Join
- Right Outer Join
- Full Outer Join

Inner Join:

In an inner join, only those tuples that satisfy the matching criteria are included, while the rest are excluded. Let's study various types of Inner Joins:

Theta Join:

The general case of JOIN operation is called a Theta join. It is denoted by symbol  $\theta$ .

**Example:**  $A \bowtie \theta B$

Theta join can use any conditions in the selection criteria.

**For example:**

$A \bowtie A.\text{column 2} > B.\text{column 2} (B)$

**A  $\bowtie$  A.column 2 > B.column 2 (B)**

column 1	column 2
1	2

EQUI join:

When a theta join uses only equivalence condition, it becomes a equi join.

**For example:**

$A \bowtie A.\text{column 2} = B.\text{column 2} (B)$

**A  $\bowtie$  A.column 2 = B.column 2 (B)**

1

EQUI join is the most difficult operations to implement efficiently in an RDBMS and one reason why RDBMS have essential performance problems.

## NATURAL JOIN ( $\bowtie$ )

Natural join can only be performed if there is a common attribute (column) between the relations. The name and type of the attribute must be same.

## Example

Consider the following two tables

C

2 4  
3 9

D

2 8  
3 27

$C \bowtie D$

C $\bowtie$ D		
Num	Square	Cube
2	4	4
3	9	27

## OUTER JOIN

In an outer join, along with tuples that satisfy the matching criteria, we also include some or all tuples that do not match the criteria.

### Left Outer Join (A $\bowtie$ B)

In the left outer join, operation allows keeping all tuple in the left relation. However, if there is no matching tuple is found in right relation, then the attributes of right relation in the join result are filled with null values.



Image 21- Left Outer Join

Reference- <https://www.guru99.com/relational-algebra-dbms.html>

Consider the following 2 Tables

A	
Num	Square
2	4
3	9

**4**

**16**

<b>B</b>	
<b>Num</b>	<b>Cube</b>
<b>2</b>	<b>8</b>
<b>3</b>	<b>18</b>
<b>5</b>	<b>75</b>

A  $\bowtie$  B

$\bowtie$		
<b>A <math>\bowtie</math> B</b>		
<b>Num</b>	<b>Square</b>	<b>Cube</b>
<b>2</b>	<b>4</b>	<b>4</b>
<b>3</b>	<b>9</b>	<b>9</b>
<b>4</b>	<b>16</b>	-

Right Outer Join: ( A  $\bowtie$  B )

In the right outer join, operation allows keeping all tuple in the right relation. However, if there is no matching tuple is found in the left relation, then the attributes of the left relation in the join result are filled with null values.

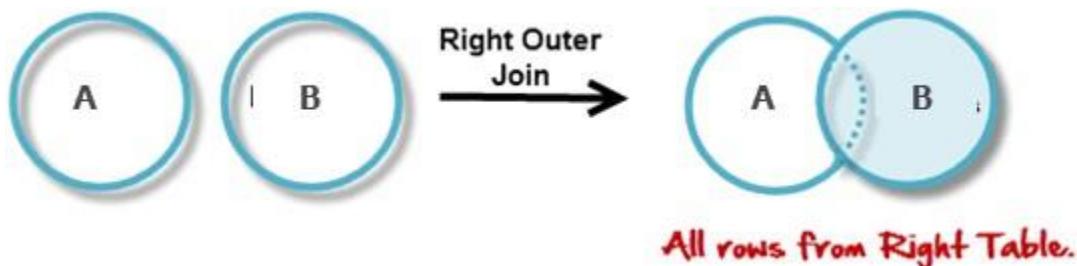


Image 22- Right Outer Join

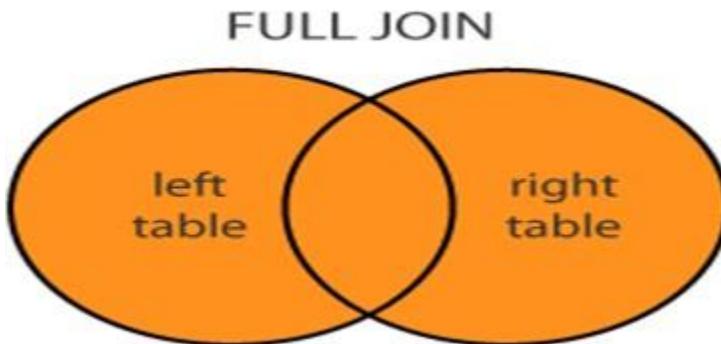
Reference- <https://www.guru99.com/relational-algebra-dbms.html>

$$A \bowtie B$$

Num	Cube	Square
2	8	4
3	18	9
5	75	-

Full Outer Join: (  $A \bowtie B$  )

In a full outer join, all tuples from both relations are included in the result, irrespective of the matching condition.



$$\bowtie$$

Image 23- Full Outer Join

$$\bowtie$$

**A  $\bowtie$  B**

<b>A <math>\bowtie</math> B</b>		
<b>Num</b>	<b>Cube</b>	<b>Square</b>
<b>2</b>	4	<b>8</b>
<b>3</b>	9	<b>18</b>
<b>4</b>	16	-
<b>5</b>	-	75

## Set Operations

The SQL Set operation is used to combine the two or more SQL SELECT statements.

### Types of Set Operation

1. Union
2. UnionAll
3. Intersect
4. Minus



Image 24- Set Operations

Reference- <https://www.javatpoint.com/dbms-sql-set-operation>

## 1. Union

- The SQL Union operation is used to combine the result of two or more SQL SELECT queries.
- In the union operation, all the number of datatype and columns must be same in both the tables on which UNION operation is being applied.
- The union operation eliminates the duplicate rows from its resultset.

### Syntax

```
SELECT column_name FROM table1
```

```
UNION
```

```
SELECT column_name FROM table2;
```

### Example:

#### The First table

ID	NAME
1	Jack
2	Harry
3	Jackson

### The Second table

ID	NAME
3	Jackson
4	Stephan
5	David

Union SQL query will be:

```
SELECT * FROM First
UNION
SELECT * FROM Second;
```

The resultset table will look like:

ID	NAME
1	Jack
2	Harry
3	Jackson
4	Stephan
5	David

## 2. Union All

Union All operation is equal to the Union operation. It returns the set without removing duplication and sorting the data.

### Syntax:

```
SELECT column_name FROM table1
```

```
UNION ALL
```

```
SELECT column_name FROM table2;
```

**Example:** Using the above First and Second table. Union All query will be like:

```
SELECT * FROM First
```

```
UNION ALL
```

```
SELECT * FROM Second;
```

The resultset table will look like:

ID	NAME
1	Jack
2	Harry
3	Jackson
3	Jackson
4	Stephan
5	David

### 3. Intersect

- It is used to combine two SELECT statements. The Intersect operation returns the common rows from both the SELECT statements.
- In the Intersect operation, the number of datatype and columns must be the same.
- It has no duplicates and it arranges the data in ascending order by default.

#### Syntax

```
SELECT column_name FROM table1
```

```
INTERSECT
```

```
SELECT column_name FROM table2;
```

#### Example:

##### **Using the above First and Second table.**

Intersect query will be:

```
SELECT * FROM First
```

```
INTERSECT
```

```
SELECT * FROM Second;
```

The resultset table will look like:

ID	NAME
3	Jackson

#### 4. Minus

- It combines the result of two SELECT statements. Minus operator is used to display the rows which are present in the first query but absent in the second query.
- It has no duplicates and data arranged in ascending order by default.

##### **Syntax:**

```
SELECT column_name FROM table1
```

```
MINUS
```

```
SELECT column_name FROM table2;
```

##### **Example**

###### **Using the above First and Second table.**

Minus query will be:

```
SELECT * FROM First
```

```
MINUS
```

```
SELECT * FROM Second;
```

**The resultset table will look like:**

ID	NAME
1	Jack
2	Harry

## Fundamental Operations

### SELECT ( $\sigma$ )

The SELECT operation is used for selecting a subset of the tuples according to a given selection condition. Sigma( $\sigma$ )Symbol denotes it. It is used as an expression to choose tuples which meet the selection condition. Select operation selects tuples that satisfy a given predicate.

$\sigma p(r)$

$\sigma$  is the predicate

r stands for relation which is the name of the tablep is prepositional logic

#### Example 1

$\sigma$  topic = "Database" (Tutorials)

Output - Selects tuples from Tutorials where topic = 'Database'.

#### Example 2

$\sigma$  topic = "Database" and author = "Edunet"( Tutorials)

Output - Selects tuples from Tutorials where the topic is 'Database' and 'author' is Edunet.

#### Example 3

$\sigma$  sales > 50000 (Customers)

Output - Selects tuples from Customers where sales is greater than 50000

### Projection( $\pi$ )

The projection eliminates all attributes of the input relation but those mentioned in the projection list. The projection method defines a relation that contains a vertical subset of Relation.

This helps to extract the values of specified attributes to eliminate duplicate values. ( $\pi$ ) Thesymbol used to choose attributes from a relation. This operation helps you to keep specific columns from a relation and discards the other columns.

Example of Projection:

Consider the following table

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive
4	Alibaba	Active

Here, the projection of CustomerName and status will give  $\Pi$  CustomerName, Status (Customers)

CustomerName	Status
Google	Active
Amazon	Active
Apple	Inactive
Alibaba	Active

### Cartesian product(X)

This type of operation is helpful to merge columns from two relations. Generally, a Cartesian product is never a meaningful operation when it performs alone. However, it becomes meaningful when it is followed by other operations.

## Example – Cartesian product

$\sigma$  column 2 = '1' (A X B)

Output – The above example shows all rows from relation A and B whose column 2 has value 1

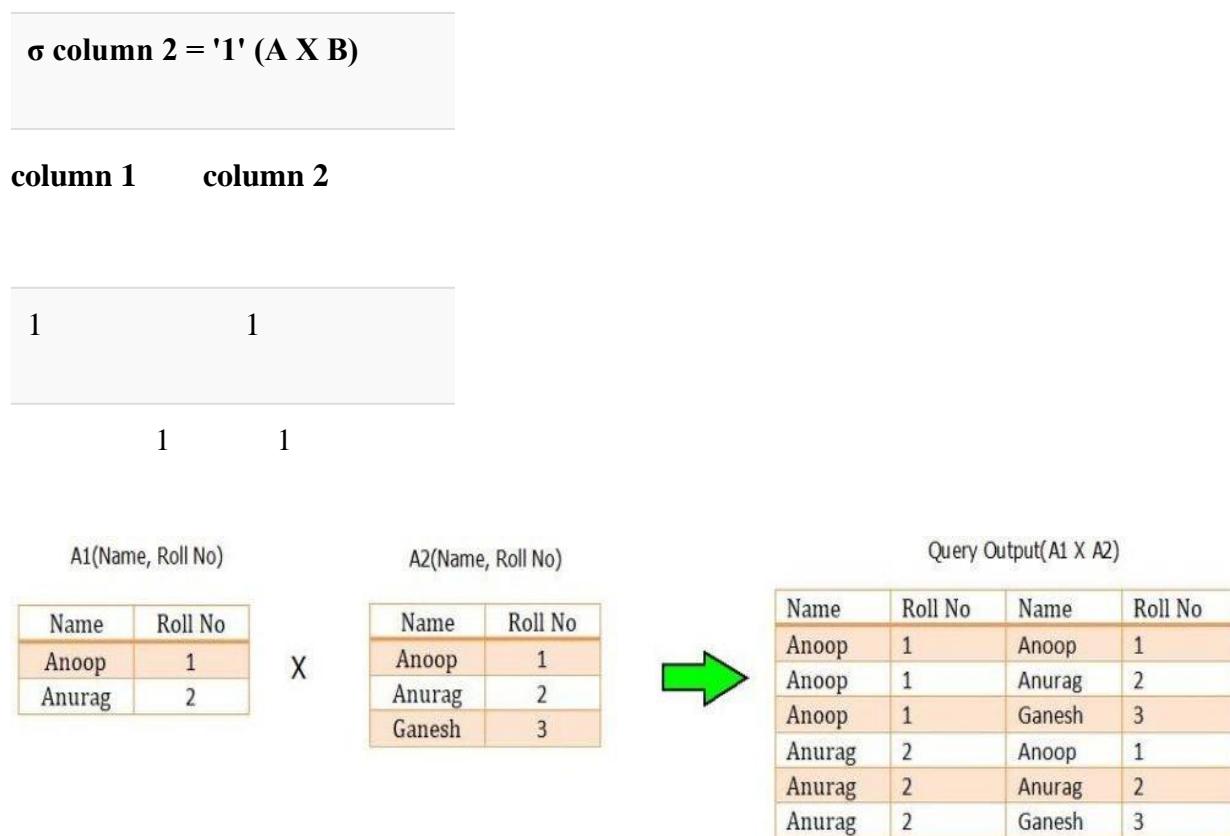


Image 20- Cartesian Product

Reference- <https://www.minigranth.com/dbms-tutorial/relational-algebra/>

## Union operation (v)

UNION is symbolized by  $\cup$  symbol. It includes all tuples that are in tables A or in B. It also eliminates duplicate tuples. So, set A UNION set B would be expressed as:

The result  $\leftarrow A \cup B$

For a union operation to be valid, the following conditions must hold -

- R and S must be the same number of attributes.
- Attribute domains need to be compatible.
- Duplicate tuples should be automatically removed.

## Example

Consider the following tables.

**Table A**

**Table B**

**column 1    column 2**

**column 1    column 2**

1                    2

1                    3

$A \cup B$  gives

**Table A  $\cup$  B**

**column 1    column 2**

1                    2

1                    2

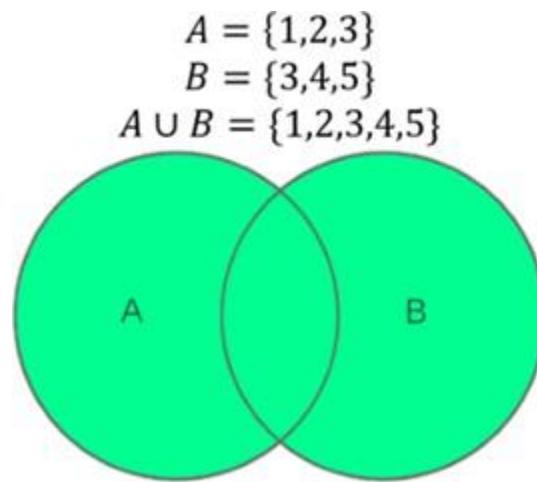


Image 17- Union Example

Reference- <https://www.codeproject.com/articles/1172312/just-enough-set-theory-when-sets-collide-part-of>

### Set Difference (-)

- Symbol denotes it. The result of  $A - B$ , is a relation which includes all tuples that are in A but not in B.
  - The attribute name of A has to match with the attribute name in B.
  - The two-operand relations A and B should be either compatible or Union compatible.
  - It should be a defined relation consisting of the tuples that are in relation A, but not in B.

### Example

A-B

#### Table A - B

column 1	column 2
1	2

## Relational Calculus

### What is Relational Calculus?

Contrary to Relational Algebra which is a procedural query language to fetch data and which also explains how it is done, **Relational Calculus** is a non-procedural query language and has no description about how the query will work or the data will be fetched. It only focuses on what to do, and not on how to do it.

Relational Calculus exists in two forms:

1. Tuple Relational Calculus (TRC)
2. Domain Relational Calculus (DRC)

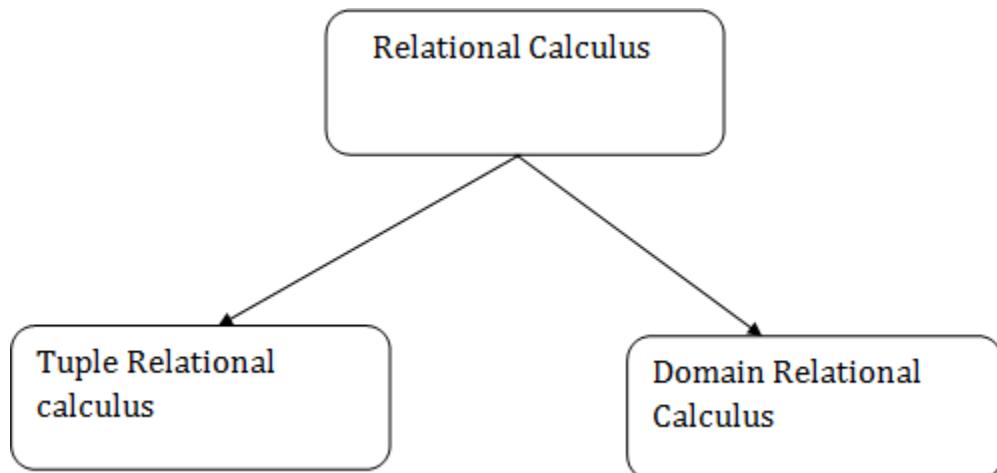


Image 25- Types of Relational Calculus

Reference- <https://www.javatpoint.com/dbms-relational-calculus>

## 1. Tuple Relational Calculus (TRC)

Tuple Relational Calculus is a **non-procedural query language** unlike relational algebra. Tuple Calculus provides only the description of the query but it does not provide the methods to solve it. Thus, it explains what to do but not how to do.

In Tuple Calculus, a query is expressed as

$\{t \mid P(t)\}$

where  $t$  = resulting tuples,

$P(t)$  = known as Predicate and these are the conditions that are used to fetch  $t$  Thus, it generates a set of all tuples  $t$ , such that Predicate  $P(t)$  is true for  $t$ .

$P(t)$  may have various conditions logically combined with OR ( $\vee$ ), AND ( $\wedge$ ), NOT( $\neg$ ).

It also uses quantifiers:

$\exists t \in r (Q(t))$  = "there exists" a tuple in  $t$  in relation  $r$  such that predicate  $Q(t)$  is true.

$\forall t \in r (Q(t))$  =  $Q(t)$  is true "for all" tuples in relation  $r$ .

Example:

**Table-1: Customer**

CUSTOMER NAME	STREET	CITY
Saurabh	A7	Patiala
Mehak	B6	Jalandhar
Sumiti	D9	Ludhiana
Ria	A5	Patiala

**Table-2: Branch**

BRANCH NAME	BRANCH CITY
-------------	-------------

ABC

Patiala

DEF

Ludhiana

GHI

Jalandhar

**Table-3: Account**

ACCOUNT NUMBER	BRANCH NAME	BALANCE
----------------	-------------	---------

1111

ABC

50000

1112

DEF

10000

1113

GHI

9000

1114

ABC

7000

**Table-4: Loan**

LOAN NUMBER	BRANCH NAME	AMOUNT
L33	ABC	10000
L35	DEF	15000
L49	GHI	9000
L98	DEF	65000

**Table-5: Borrower**

CUSTOMER NAME	LOAN NUMBER
Saurabh	L33
Mehak	L49
Ria	L98

**Table-6: Depositor**

CUSTOMER NAME	ACCOUNT NUMBER
---------------	----------------

Saurabh 1111

Mehak	1113
Sumiti	1114

**Queries-1:** Find the loan number, branch, amount of loans of greater than or equal to 10000 amount.

$\{t \mid t \in \text{loan} \wedge t[\text{amount}] \geq 10000\}$

**Resulting relation:**

LOAN NUMBER	BRANCH NAME	AMOUNT
L33	ABC	10000
L35	DEF	15000
L98	DEF	65000

In the above query,  $t[\text{amount}]$  is known as tuple variable.

**Queries-2:** Find the loan number for each loan of an amount greater or equal to 10000.

$$\{t \mid \exists s \in \text{loan}(t[\text{loan number}] = s[\text{loan number}]$$
$$\wedge s[\text{amount}] \geq 10000)\}$$

**Resulting relation:**

LOAN NUMBER
L33
L35
.....L98

L33

L35

.....L98

**Queries-3:** Find the names of all customers who have a loan and an account at the bank.

$$\{t \mid \exists s \in \text{borrower}(t[\text{customer-name}] = s[\text{customer-name}])$$
$$\wedge \exists u \in \text{depositor}(t[\text{customer-name}] = u[\text{customer-name}])\}$$

**Resulting relation:**

CUSTOMER NAME
Saurabh
Mehak

Saurabh

Mehak

**Queries-4:** Find the names of all customers having a loan at the “ABC” branch.

$\{t \mid \exists s \in \text{borrower}(t[\text{customer-name}] = s[\text{customer-name}]$

$\wedge \exists u \in \text{loan}(u[\text{branch-name}] = \text{“ABC”} \wedge u[\text{loan-number}] =$

$s[\text{loan-number}])\}$  Resulting relation:

**CUSTOMER NAME**

Saurabh

## 2. Domain Relational Calculus (DRC)

**Domain Relational Calculus** is a non-procedural query language equivalent in power to Tuple Relational Calculus. Domain Relational Calculus provides only the description of the query but it does not provide the methods to solve it. In Domain Relational Calculus, a query is expressed as,

$\{ < x_1, x_2, x_3, \dots, x_n > \mid P(x_1, x_2, x_3, \dots, x_n) \}$

where,  $< x_1, x_2, x_3, \dots, x_n >$  represents resulting domains variables and  $P(x_1, x_2, x_3, \dots, x_n)$  represents the condition or formula equivalent to the Predicate calculus.

Predicate Calculus Formula:

1. Set of all comparison operators
2. Set of connectives like and, or, not
3. Set of quantifiers

Example:

**Table-1: Customer**

CUSTOMER NAME	STREET	CITY
Debomit	Kadamtala	Alipurduar
Sayantan	Udaypur	Balurghat
Soumya	Nutanchati	Bankura
Ritu	Juhu	Mumbai

**Table-2: Loan**

LOAN NUMBER	BRANCH NAME	AMOUNT
L01	Main	200
L03	Main	150
L10	Sub	90
L08	Main	60

**Table-3: Borrower**

CUSTOMER NAME	LOAN NUMBER
Ritu	L01
Debomit	L08
Soumya	L03

**Query-1:** Find the loan number, branch, amount of loans of greater than or equal to 100 amount.

$\{ \langle l, b, a \rangle \mid \langle l, b, a \rangle \in \text{loan} \wedge (a \geq 100) \}$

## Resulting relation:

LOAN NUMBER	BRANCH NAME	AMOUNT
1234567890	MAIN BRANCH	100000.00

L01 Main 200

L03 Main 150

**Query-2:** Find the loan number for each loan of an amount greater or equal to 150.

$\{ \langle l \rangle \mid \exists b, a \ (\langle l, b, a \rangle \in \text{loan} \wedge (a \geq 150)) \}$

## Resulting relation:

## LOAN NUMBER

L01

L03

**Query-3:** Find the names of all customers having a loan at the “Main” branch and find the loan amount.

$\{ \langle c, a \rangle \mid \exists l (\langle c, l \rangle \in \text{borrower} \wedge \exists b (\langle l, b, a \rangle \in \text{loan} \wedge (b = \text{“Main”}))) \}$

**Resulting relation:**

CUSTOMER NAME	AMOUNT
---------------	--------

Ritu 200

Debomit 60

Soumya 150

**Note:** The domain variables those will be in resulting relation must appear before | within  $\prec$  and  $\succ$  and all the domain variables must appear in which order they are in original relation or table.

### **Difference between Relational Algebra and Relational Calculus:**

S.NO	RELATIONAL ALGEBRA	RELATIONAL CALCULUS
1	It is a Procedural language.	While Relational Calculus is Declarative language.
2	Relational Algebra means how to obtain the result.	While Relational Calculus means what result we have to obtain.
3	In Relational Algebra, The order is specified in which the operations have to be performed.	While in Relational Calculus, the order is not specified.
4	Relational Algebra is independent on domain.	While Relation Calculus can be a domain dependent.
5	Relational Algebra is nearer to a programming language.	While Relational Calculus is not nearer to programming language.

---

## Data Definition language

### Introduction to DDL

- DDL stands for Data Definition Language.
- It is a language used for defining and modifying the data and its structure.
- It is used to build and modify the structure of your tables and other objects in the database.

DDL commands are as follows:

1. CREATE

2. DROP

3. ALTER

4. RENAME

5. TRUNCATE

- These commands can be used to add, remove or modify tables within a database.
- DDL has pre-defined syntax for describing the data.

### 1. CREATE COMMAND

- **CREATE command** is used for creating objects in the database.
- It creates a new table.

Syntax:

CREATE TABLE <table\_name>(column\_name1 datatype,

column\_name2 datatype,

.

.

.

column\_name\_n datatype);

### Example:

```
Create commandCREATE TABLE employee  
(empid INT, ename CHAR, age INT,  
city CHAR(25), phone_no VARCHAR(20));
```

## 2. DROP COMMAND

- **DROP command** allows to remove entire database objects from the database.
- It removes entire data structure from the database.
- It deletes a table, index or view.

Syntax:

```
DROP TABLE <table_name>; OR  
DROP DATABASE <database_name>;
```

### Example:

```
DROP CommandDROP TABLE employee; OR  
DROP DATABASE employees;
```

If you want to remove individual records, then use DELETE command of the DML statement.

## 3. ALTER COMMAND

- An **ALTER command** allows to alter or modify the structure of the database.
- It modifies an existing database object.
- Using this command, you can add additional column, drop existing column and even change the data type of columns.

Syntax:

```
ALTER TABLE <table_name> ADD <column_name datatype>;
```

OR

---

```
ALTER TABLE <table_name>
CHANGE <old_column_name> <new_column_name>;OR
ALTER TABLE <table_name> DROP COLUMN <column_name>;
```

**Example:**

```
Alter CommandALTER TABLE employee
```

```
ADD (address varchar2(50));
```

```
OR
```

```
ALTER TABLE employee CHANGE (phone_no) (contact_no);
```

```
OR
```

```
ALTER TABLE employee
```

```
DROP COLUMN age;
```

To view the changed structure of table, use 'DESCRIBE' command.

**For example:**

```
DESCRIBE TABLE employee;
```

## **4. RENAME COMMAND**

**RENAME command** is used to rename an object.

It renames a database table.

Syntax:

```
RENAME TABLE <old_name> TO <new_name>;
```

**Example:**

```
RENAME TABLE emp TO employee;
```

---

## 5. TRUNCATE COMMAND

- **TRUNCATE command** is used to delete all the rows from the table permanently.
- It removes all the records from a table, including all spaces allocated for the records.
- This command is same as DELETE command, but TRUNCATE command does not generate any rollback data.

Syntax:

```
TRUNCATE TABLE <table_name>;
```

**Example:**

```
TRUNCATE TABLE employee;
```

## Operators: Select, Project, Join, Rename etc

### SELECT ( $\sigma$ )

The SELECT operation is used for selecting a subset of the tuples according to a given selection condition. Sigma( $\sigma$ )Symbol denotes it. It is used as an expression to choose tuples which meet the selection condition. Select operation selects tuples that satisfy a given predicate.

$\sigma p(r)$

$\sigma$  is the predicate

r stands for relation which is the name of the table p is prepositional logic

#### Example 1

$\sigma$  topic = "Database" (Tutorials)

Output - Selects tuples from Tutorials where topic = 'Database'.

#### Example 2

$\sigma$  topic = "Database" and author = "Edunet" ( Tutorials)

Output - Selects tuples from Tutorials where the topic is 'Database' and 'author' is Edunet.

#### Example 3

$\sigma$  sales > 50000 (Customers)

Output - Selects tuples from Customers where sales is greater than 50000

### Projection( $\pi$ )

The projection eliminates all attributes of the input relation but those mentioned in the projection list. The projection method defines a relation that contains a vertical subset of Relation.

This helps to extract the values of specified attributes to eliminate duplicate values. ( $\pi$ ) The symbol used to choose attributes from a relation. This operation helps you to keep specific columns from a relation and discards the other columns.

Example of Projection:

Consider the following table

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive
4	Alibaba	Active

Here, the projection of CustomerName and status will give  $\Pi$  CustomerName, Status (Customers)

CustomerName	Status
Google	Active
Amazon	Active
Apple	Inactive
Alibaba	Active

## JOIN

Join operation is essentially a cartesian product followed by a selection criterion. Join operation denoted by  $\bowtie$ .

JOIN operation also allows joining variously related tuples from different relations.

---

## Types of JOIN

Various forms of join operation are:

1. Inner Joins:

- a. Theta join
- b. EQUI join
- c. Natural join

2. Outer join:

- a. Left Outer Join
- b. Right Outer Join
- c. Full Outer Join

### Inner Join:

In an inner join, only those tuples that satisfy the matching criteria are included, while the rest are excluded. Let's study various types of Inner Joins:

#### Theta Join:

The general case of JOIN operation is called a Theta join. It is denoted by symbol  $\theta$

**Example:**  $A \bowtie \theta B$

Theta join can use any conditions in the selection criteria.

**For example:**

$A \bowtie A.\text{column 2} > B.\text{column 2} (B)$

**A  $\bowtie A.\text{column 2} > B.\text{column 2} (B)$**

column 1	column 2
1	2

When a theta join uses only equivalence condition, it becomes a equi join.

**For example:**

$A \bowtie A.\text{column 2} = B.\text{column 2} (B)$

**A  $\bowtie A.\text{column 2} = B.\text{column 2} (B)$**

column 1	column 2
1	1

EQUI join is the most difficult operations to implement efficiently in an RDBMS and one reason why RDBMS have essential performance problems.

## NATURAL JOIN ( $\bowtie$ )

Natural join can only be performed if there is a common attribute (column) between the relations. The name and type of the attribute must be same.

### Example

Consider the following two tables

C		
Num	Square	
2	4	
3	9	

D		
Num	Cube	
2	8	
3	27	

$C \bowtie D$

$\bowtie \bowtie D$		
Num	Square	Cube
2	4	4
3	9	27

## OUTER JOIN

In an outer join, along with tuples that satisfy the matching criteria, we also include some or all tuples that do not match the criteria.

Left Outer Join(A  $\bowtie$  B)

In the left outer join, operation allows keeping all tuple in the left relation. However, if there is no matching tuple is found in right relation, then the attributes of right relation in the join result are filled with null values.

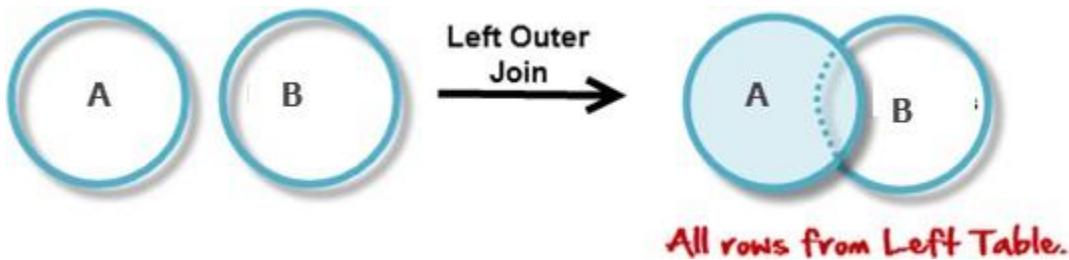


Image 26- Left Outer Join

Reference- <https://www.guru99.com/relational-algebra-dbms.html>

Consider the following 2 Tables

<b>A</b>	
<b>Num</b>	<b>Square</b>
2	4
3	9
4	16

<b>B</b>	
<b>Num</b>	<b>Cube</b>
2	8
3	18
5	75

A  $\bowtie$  B

<b>A <math>\bowtie</math> B</b>		
<b>Num</b>	<b>Square</b>	<b>Cube</b>
2	4	4
3	9	9
4	16	-

Right Outer Join: ( A  $\bowtie$  B )

In the right outer join, operation allows keeping all tuple in the right relation. However, if there is no matching tuple is found in the left relation, then the attributes of the left relation in the join result are filled with null values.

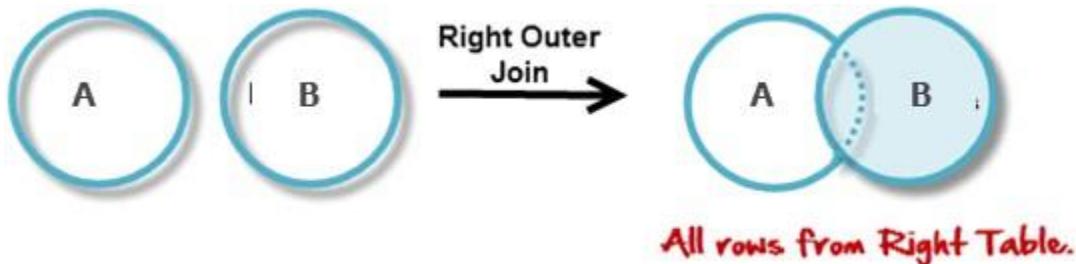


Image 27- Right Outer Join

Reference- <https://www.guru99.com/relational-algebra-dbms.html>

A  $\bowtie$  B

<b>A <math>\bowtie</math> B</b>		
<b>Num</b>	<b>Cube</b>	<b>Square</b>
3	18	9
5	75	-

Full Outer Join: ( A  $\bowtie$  B)

In a full outer join, all tuples from both relations are included in the result, irrespective of the matching condition.

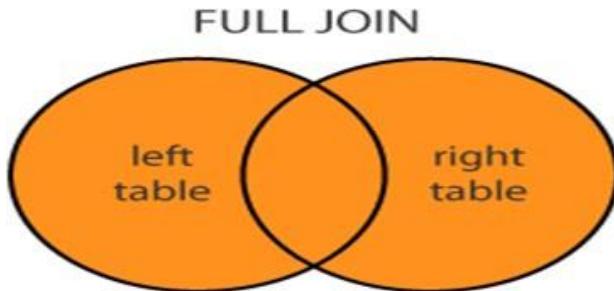


Image 28- Full Outer Join

$A \bowtie B$

$\bowtie$

Num	Cube	Square
2	4	8
3	9	18
4	16	-
5	-	75

Rename( $\rho$ )

The results of relational algebra are also relations but without any name. The rename operation allows us to rename the output relation. 'rename' operation is denoted with small Greek letter rho  $\rho$ .

Notation –  $\rho x (E)$

Where the result of expression E is saved with the name of x.

**Example:**  $\rho(\text{RelationNew}, \text{RelationOld})$

---

**Learning Outcome**  
**Software Development Life Cycle (SDLC)**

## SDLC Overview

### What is SDLC?

Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality softwares. The SDLC aims to produce a high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.

SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.



Image 1: SDLC  
Reference: <https://svitla.com/blog/system-development-life-cycle>

### Why SDLC?

- It offers a basis for project planning, scheduling, and estimating
- Provides a framework for a standard set of activities and deliverables
- It is a mechanism for project tracking and control
- Increases visibility of project planning to all involved stakeholders of the development process
- Increased and enhance development speed
- Improved client relations
- Helps you to decrease project risk and project management plan overhead

### Phases of Software Development Life Cycle (SDLC)

The entire SDLC process divided into the following SDLC steps:

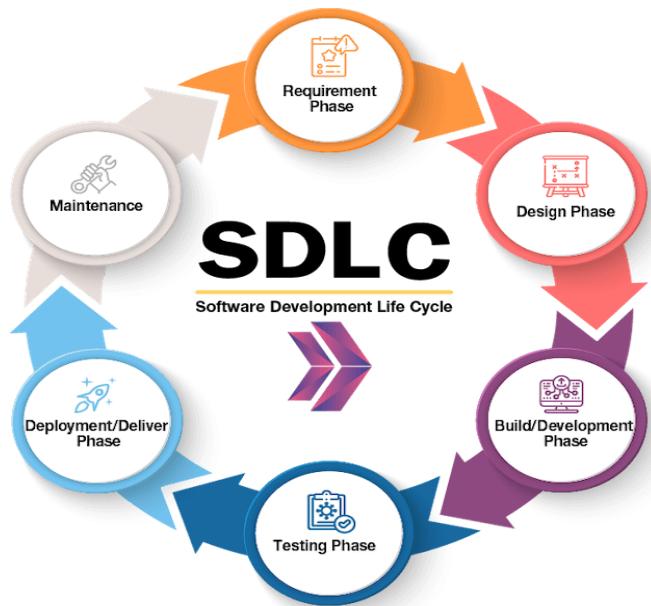


Image 2: SDLC Life Cycle  
Reference: <https://www.slideshare.net/webdevninja/introduction-to-css-13603389>

- Phase 1: Requirement collection and analysis
- Phase 2: Feasibility study
- Phase 3: Design
- Phase 4: Coding
- Phase 5: Testing
- Phase 6: Installation/Deployment
- Phase 7: Maintenance

### Phase 1: Requirement collection and analysis

The requirement is the first stage in the SDLC process. It is conducted by the senior team members with inputs from all the stakeholders and domain experts in the industry. Planning for the quality assurance requirements and recognition of the risks involved is also done at this stage.

This stage gives a clearer picture of the scope of the entire project and the anticipated issues, opportunities, and directives which triggered the project.

Requirements Gathering stage need teams to get detailed and precise requirements. This helps companies to finalize the necessary timeline to finish the work of that system.

---

## Phase 2: Feasibility study

Once the requirement analysis phase is completed the next sdlc step is to define and document software needs. This process conducted with the help of ‘Software Requirement Specification’ document also known as ‘SRS’ document. It includes everything which should be designed and developed during the project life cycle.

There are mainly five types of feasibilities checks:

- Economic: Can we complete the project within the budget or not?
- Legal: Can we handle this project as cyber law and other regulatory framework/compliances.
- Operation feasibility: Can we create operations which is expected by the client?
- Technical: Need to check whether the current computer system can support the software
- Schedule: Decide that the project can be completed within the given schedule or not.

## Phase 3: Design

In this third phase, the system and software design documents are prepared as per the requirement specification document. This helps define overall system architecture.

This design phase serves as input for the next phase of the model.

There are two kinds of design documents developed in this phase:

### High-Level Design (HLD)

- Brief description and name of each module
- An outline about the functionality of every module
- Interface relationship and dependencies between modules
- Database tables identified along with their key elements
- Complete architecture diagrams along with technology details

### Low-Level Design (LLD)

- Functional logic of the modules
- Database tables, which include type and size
- Complete detail of the interface
- Addresses all types of dependency issues
- Listing of error messages
- Complete input and outputs for every module

---

#### **Phase 4: Coding**

Once the system design phase is over, the next phase is coding. In this phase, developers start build the entire system by writing code using the chosen programming language. In the coding phase, tasks are divided into units or modules and assigned to the various developers. It is the longest phase of the Software Development Life Cycle process.

In this phase, Developer needs to follow certain predefined coding guidelines. They also need to use programming tools like compiler, interpreters, debugger to generate and implement the code.

#### **Phase 5: Testing**

Once the software is complete, and it is deployed in the testing environment. The testing team starts testing the functionality of the entire system. This is done to verify that the entire application works according to the customer requirement.

During this phase, QA and testing team may find some bugs/defects which they communicate to developers. The development team fixes the bug and send back to QA for a re-test. This process continues until the software is bug-free, stable, and working according to the business needs of that system.

#### **Phase 6: Installation/Deployment**

Once the software testing phase is over and no bugs or errors left in the system then the final deployment process starts. Based on the feedback given by the project manager, the final software is released and checked for deployment issues if any.

---

#### **Phase 7: Maintenance**

Once the system is deployed, and customers start using the developed system, following 3 activities occur

- Bug fixing – bugs are reported because of some scenarios which are not tested at all
- Upgrade – Upgrading the application to the newer versions of the Software
- Enhancement – Adding some new features into the existing software

The main focus of this SDLC phase is to ensure that needs continue to be met and that the system continues to perform as per the specification mentioned in the first phase.

---

## Software Development Life Cycle Models

Software development (SDLC) can be performed according to a variety of models, varying based on project type, requirements given, time constraints, resources, and broader company product development objectives.

We introduce six of the most popular models, including

- waterfall,
- iterative,
- V-Shaped
- Spiral
- Agile
- Big Bang.

### SDLC Model: Waterfall

The waterfall model is seen as one phase “flowing” into the next as any phase in the process cannot commence until the previous is completed. The most commonly accepted of models is documentation-intensive as the earlier documentation tells what will be performed in future phases. Possibly the most straightforward of models, it is easy to manage, although flaws cannot be corrected until the “maintenance” stage.



Image 3: Waterfall model  
Reference: <https://sdlcpartners.com/insights/what-are-sdlc-types/>

#### Uses:

Waterfall is typically made up of seven consecutive phases: requirement gathering, feasibility analysis, design, coding, testing, installation, and maintenance. Often used for simple, small or mid-sized projects, waterfall has traditionally been seen as offering a stricter control mechanism. It can be appropriate for smaller projects where the technology is understood and the requirements are well defined, fixed.

---

**Pros:**

- Simple to use and understand
- Easy to classify and prioritize tasks
- Well understood milestones and checkpoints
- Each phase has specific deliverables

**Cons:**

- High risks and uncertainty
- Assumes the requirements of a system can be frozen
- Difficult to go back to any stage once it is finished
- Difficult to measure progress within stages

**SDLC Model: Iterative**

The iterative model is a realization of the sequential approximation method, meaning it is a series of waterfall models where the requirements are divided into groups at the beginning of the project. The main difference from the previous is that the SDLC can start without all of the requirements gathered. Developers implement a set of software requirements, test, evaluate and pinpoint further requirements. This model provides a working version earlier in the cycle.

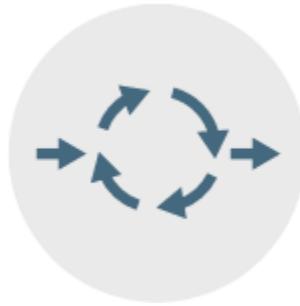


Image 4: Iterative model  
Reference: <https://sdlcpartners.com/insights/what-are-sdlc-types/>

**Uses:**

Applied to large-scale projects where the final product and main task are predefined, but the details evolve over time. This model is well-suited for projects that have a time constraint as well as large, mission-critical projects that consist of loosely coupled parts like microservices or web services.

**Pros:**

- Risk analysis is more thorough
- Initial operating time is faster
- More focused on customer value than linear approaches
- Encourages flexibility and readiness to change to evolving requirements

### Cons:

- More resources may be needed
- Complicated to manage
- End of project may not be known, which is a risk
- Partitioning the functions and features may be problematic

### SDLC Model: V-Shaped

The V-shaped model addresses validation and verification through parallel processes where coding is connected to both concurrently. This model grew out of the waterfall model where each stage begins only after the previous one has ended.



Image 5: V-shaped model  
Reference: <https://sdlcpartners.com/insights/what-are-sdlc-types/>

### Uses:

Most appropriate when failures and downtimes are unacceptable (e.g., medical software, aviation fleet management software) or where there are no unknown requirements because this model is restrictive to making changes mid-cycle.

### Pros:

- Testing and verification takes place in early stages
- Easy to control
- Highly disciplined
- Good when requirements are static and clear

### Cons:

- Lack of flexibility
- Meant only for bigger projects
- Not suitable for projects where requirements are likely to change
- Once in the testing phase, it is difficult to go back and change aspects like functionality

### SDLC Model: Spiral

---

Iterative in nature, the spiral is a combination of prototype and waterfall models. This is a risk-driven model that is more flexible and allows for building a highly customized product where user feedback can be incorporated early in the project.



Image 6: Spiral model  
Reference: <https://sdlcpartners.com/insights/what-are-sdlc-types/>

### Uses:

In the spiral model, the project passes through four phases over and over in a “spiral” until complete, allowing for multiple rounds of refinement. It is best used in large projects and systems that contain small phases or identifiable segments.

### Pros:

- Development process is well-documented and scalable
- Progress is easily measured
- High-risk tasks are completed first
- Early involvement of developers

### Cons:

- Can be expensive, creating a high cost and longer time to reach a final product
- Can be ineffective for smaller projects
- Highly customized, which limits reusability
- Needs special skills to evaluate the risks and assumptions

### SDLC Model: Agile

Agile has gained a great deal of popularity. The model breaks down big projects into smaller, more manageable chunks, which can lead to a software product that represents a culmination of multiple, smaller projects. The model produces ongoing releases where each iteration includes small, incremental changes and improvements from the previous release.

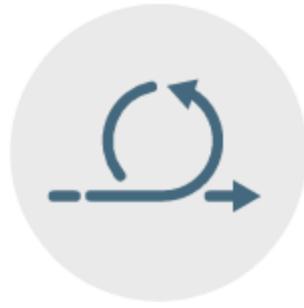


Image 7: Agile model  
Reference: <https://sdlcpartners.com/insights/what-are-sdlc-types/>

#### Uses:

Agile is particularly well-suited for large and complex projects; those that can be easily divided into smaller parts. It's also useful for mid-size custom software development projects where business requirements cannot be easily translated into detailed requirements.

#### Pros:

- Face-to-face communication and continuous inputs from customer representative leave no space for guesswork
- End result is the highest quality software in the least possible time
- Very realistic approach to development
- Suitable for fixed and evolving requirements

#### Cons:

- Documentation is created in later stages
- Reduce the usability of components
- Requires special skills and ongoing oversight
- Transfer of technology to new team members may be difficult due to lack of documentation

#### SDLC Model: Big Bang

Done with little-to-no planning, the Big Bang model focuses on all types of coding and development types, implementing requirements as they are discovered. Because it does not follow a set process and is a high-risk model, the Big Bang is best for small projects with only one or two engineers.



Image 8: Big Bang model  
Reference: <https://sdlcpartners.com/insights/what-are-sdlc-types/>

### Uses:

It is mainly used for academic software development projects or smaller projects where the development team is small and working in tight collaboration. It is helpful when requirements are unknown and a release date is very flexible.

### Pros:

- Allows for more resources to be used in development
- More straightforward to manage
- Allows developer flexibility
- Good learning aid for students or newcomers

### Cons:

- Harder to attain a firm grasp on requirements
- High-risk with high level of uncertainty
- Poor model type for long or ongoing projects
- Can be expensive if requirements are misunderstood and timelines are vague

## Software Test Levels

### What is Software Testing?

Software testing is a process, to evaluate the functionality of a software application with an intent to find whether the developed software met the specified requirements or not and to identify the defects to ensure that the product is defect free in order to produce the quality product.

---

## Types of Software Testing

### **Manual Testing:**

Manual testing is the process of testing the software by hand to learn more about it, to find what is and isn't working.

This usually includes verifying all the features specified in requirements documents, but often also includes the testers trying the software with the perspective of their end user's in mind.

Manual test plans vary from fully scripted test cases, giving testers detailed steps and expected results, through to high-level guides that steer exploratory testing sessions.

There are lots of sophisticated tools on the market to help with manual testing, but if you want a simple and flexible place to start, take a look at Testpad.

### **Automation Testing:**

Automation testing is the process of testing the software using an automation tool to find the defects.

In this process, testers execute the test scripts and generate the test results automatically by using automation tools.

Some of the famous automation testing tools for functional testing include Selenium and Katalon Studio.

Selenium is no longer a strange name for web application testers. It offers powerful capabilities like cross-browser testing but is difficult to learn for those new to automation or with limited programming experience. That's why most QAs freshers and manual testers start out with Katalon Studio. While still providing essential features from Selenium, users can utilize its simple UI, built-in keywords, and record & playback to create test cases easier and pick up programming skills along the way with the scripting mode (Java and Groovy supported).

## Levels of Testing

There are mainly four Levels of Testing in software testing:

- **Unit Testing:** checks if software components are fulfilling functionalities or not.
- **Integration Testing:** checks the data flow from one module to other modules.
- **System Testing:** evaluates both functional and non-functional needs for the testing.
- **Acceptance Testing:** checks the requirements of a specification or contract are met as per its delivery.

Unit Testing	Done by Developers
Integration Testing	Done by Testers
System Testing	Done by Testers
Acceptance Testing	Done by End Users

Image 9: Levels of Testing

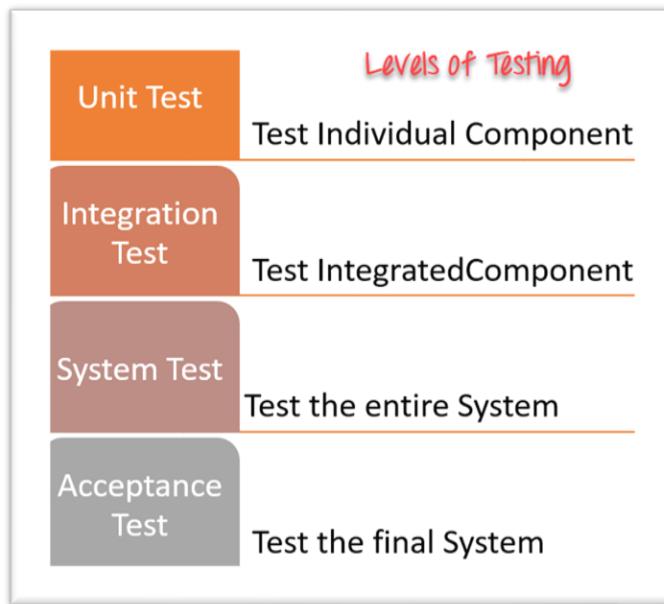


Image 10: Levels of Testing  
Reference: <https://www.guru99.com/levels-of-testing.html>

## Unit Testing:

Unit Testing is done to check whether the individual modules of the source code are working properly. i.e. testing each and every unit of the application separately by the developer in the developer's environment. It is AKA Module Testing or Component Testing.

---

## **Integration Testing:**

Integration Testing is the process of testing the connectivity or data transfer between a couple of unit tested modules. It is AKA I&T Testing or String Testing

It is subdivided into the Top-Down Approach, Bottom-Up Approach and Sandwich Approach (Combination of Top Down and Bottom Up). This process is carried out by using dummy programs called Stubs and Drivers. Stubs and Drivers do not implement the entire programming logic of the software module but just simulate data communication with the calling module.

- **Big Bang Integration Testing:**

In Big Bang Integration Testing, the individual modules are not integrated until all the modules are ready. Then they will run to check whether it is performing well. In this type of testing, some disadvantages might occur like, defects can be found at the later stage. It would be difficult to find out whether the defect arises in an interface or in a module.

- **Top-Down Integration Testing**

In Top-Down Integration Testing, the high-level modules are integrated and tested first. i.e Testing from the main module to the submodule. In this type of testing, Stubs are used as a temporary module if a module is not ready for integration testing.

- **Bottom-Up Integration Testing**

In Bottom-Up Integration Testing, the low-level modules are integrated and tested first i.e. Testing from sub-module to the main module. Same like Stubs, here drivers are used as a temporary module for integration testing.

### **Stub:**

It is called by the Module under Test.

### **Driver:**

It calls the Module to be tested.

## **System Testing (End TO End Testing):**

It's a black box testing. Testing the fully integrated application this is also called as an end-to-end scenario testing. To ensure that the software works in all intended target systems. Verify thorough testing of every input in the application to check for desired outputs. Testing of the users' experiences with the application.

---

### **Acceptance Testing:**

To obtain customer sign-off so that software can be delivered and payments received.

Types of Acceptance Testing are

- Alpha Testing
- Beta Testing
- Gamma Testing.

### **Alpha Testing:**

Alpha testing is mostly like performing usability testing which is done by the in-house developers who developed the software. Sometimes this alpha testing is done by the client or outsiders with the presence of developers or testers.

### **Beta Testing:**

Beta testing is done by a limited number of end users before delivery, the change request would be fixed if the user gives feedback or reports defect.

### **Gamma Testing:**

Gamma testing is done when the software is ready for release with specified requirements; this testing is done directly by skipping all the in-house testing activities.

---

## References

1. <https://www.w3schools.in/css3/introduction-to-css/>
2. <https://www.javatpoint.com/javascript-tutorial>
3. <https://www.guru99.com/introduction-to-javascript.html>
4. <https://www.tutorialrepublic.com/javascript-tutorial/javascript-variables.php>
5. <https://www.tutorialsteacher.com/javascript/javascript-data-types>
6. <https://www.section.io/engineering-education/exception-handling-in-javascript/>
7. <https://itwebtutorials.mga.edu/js/chp1/browser-object-model.aspx>
8. <https://riptutorial.com/javascript>
9. <https://www.javatpoint.com/javascript-tutorial>
10. <https://getbootstrap.com/>
11. <https://www.w3schools.com/bootstrap/>
12. <https://www.freecodecamp.org/news/learn-bootstrap-4-in-30-minute-by-building-a-landing-page-website-guide-for-beginners-f64e03833f33/>
13. <https://dev.to/bootstrap/ui-components-5a9k>
14. <https://webpixels.io/blog/bootstrap-5-components-new-website-and-open-source-css-design-system>
15. <https://shuffle.dev/blog/2021/06/new-library-to-quickly-create-bootstrap-5-templates/>
16. <https://icons.getbootstrap.com/>
17. <https://www.codegrepper.com/code-examples/whatever/glyphicons+bootstrap+5>
18. [https://www.w3schools.com/jquery/jquery\\_intro.asp](https://www.w3schools.com/jquery/jquery_intro.asp)
19. <https://www.geeksforgeeks.org/jquery-introduction/>
20. <https://www.javatpoint.com/jquery-selectors>
21. <https://www.tutorialrepublic.com/jquery-reference/jquery-selectors.php>
22. <https://www.guru99.com/software-development-life-cycle-tutorial.html>
23. <https://www.guru99.com/levels-of-testing.html>