

# 300.3.2

## Introduction to Client-Server Models, Middleware, and Databases

# Lesson 1: Introduction to Client, Server, and Client-Server models

## Learning Objectives:

In this lesson, we will explore the ubiquitous terms, which are the client and the server. We are going to start with the definitions, and after that, we will look at the client-server model and architecture.

By the end of this lesson, learners will be able to:

- Define the Client and the Server.
- Describe the Client-Server model and architecture.
- Describe the Web Server and the Application Server.

# Agenda/Topics

3

- ❑ Overview of Client.
- ❑ Overview of Server.
- ❑ Client-Server Model.
- ❑ Client-Server Architecture Overview.
- ❑ Client-Server Architecture Components.
- ❑ Types of Client-Server Architecture.
  - 1 Tier Architecture.
  - 2 Tier Architecture.
  - 3 Tier Architecture.
  - N Tier Architecture.
- ❑ Web Server.
- ❑ Application Server.
- ❑ Most Popular Application Servers.
- ❑ Client-Side Script vs. Server-Side Script.

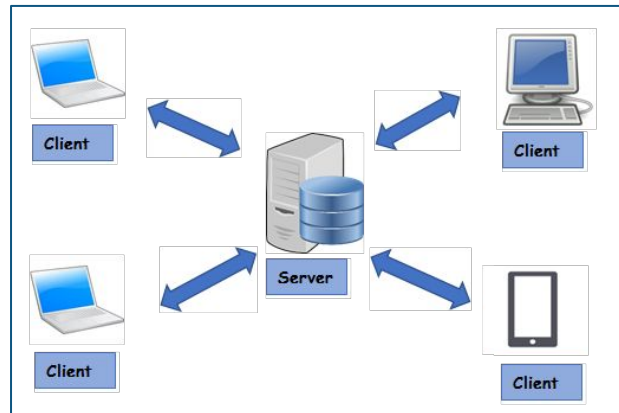
# Overview of Client

4

- ❑ A client is a [computer](#) that connects to and uses the resources of a remote computer, or [server](#). Many corporate networks comprise a client computer for each employee, each of which connects to the corporate server. The server provides resources like files, information, [Internet](#) and [intranet](#) access, and external processing power.
- ❑ **Client** is also another name for a **software program** used to connect to a server.
- ❑ A **Client** can also be another term used to describe a **user**.
- ❑ Any work done on the local client is similarly called "client-side."

## Client Types:

- [Web browsers](#) are clients that connect to [web servers](#).
- [Email clients](#) retrieve [email](#) from [mail servers](#).
- [Online chat](#) uses a variety of clients.
- Apple client - Computer using macOS.
- IoT - Internet of Things device.
- Linux client - Computer running Linux.
- Smartphone or mobile - Mobile device like a smartphone.
- Windows client - Computer running Windows.

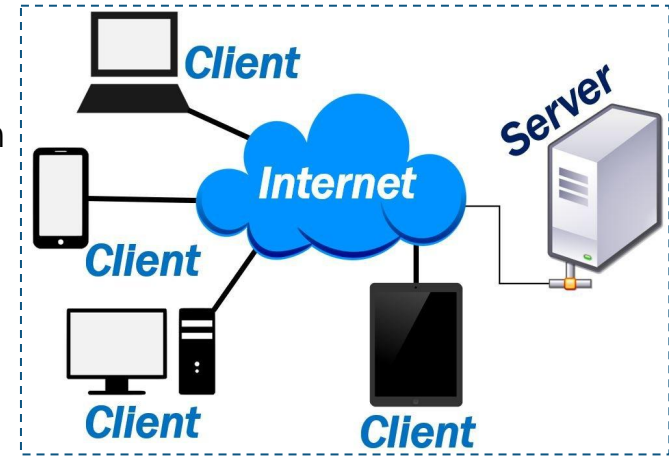


Source: itrelease.com

# Overview of Server

5

- ❑ A Server is a high performance computer that runs a Server Software and is capable of handling user requests, managing networks, data, and resources, and provides some kind of shared services to the connected workstation (clients) over a communication channel (network).
- ❑ Servers can provide various functionalities, often called "services or resources" to its client whenever requested, such as sharing data or [resources](#) among multiple **clients**, or performing [computation](#) for a client. A single server can serve multiple clients, and a single client can use multiple servers.
- ❑ Typical servers are [database servers](#), [file servers](#), [mail servers](#), [print servers](#), [web servers](#), [game servers](#), and [application servers](#).



Source: computerhope.com

# Overview of Server (continued)

6

In the case of processing, any work performed on the server is referred to as **"server-side"** work.

## Server Clusters

- ❑ A server farm or server cluster is a collection of computer servers maintained by an organization to supply server functionality far beyond the capability of a single device. Modern [data centers](#) are now often built of very large clusters of much simpler servers.
- ❑ Servers play an important role in companies' abilities to network and collaborate. They provide a shared recourse that all parties within the network can rely.



# Client vs. Server

7

**Client** is a piece of software that **uses services**, and is an application.  
A **server** is a piece of software that **provides services**.

# Host

8

- ❑ A host is a computer that is connected to other computers that it provides data or services for over a network. In theory, every computer connected to a network acts as a host to other *peers* on the network. In essence, a host reflects the logical relationship of two or more computers on a network.
- ❑ Suppose you want to download an image from another computer onto your network. That computer is “hosting” the image (the host). On the other hand, if that same computer downloads an image from your computer, then your computer becomes the host.



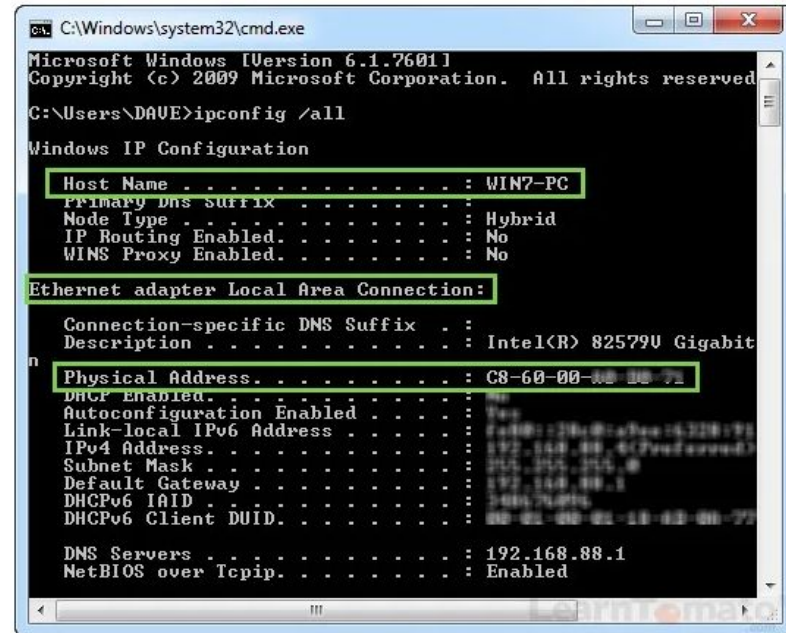
# Hostname and Host ID

9

How do you find the Hostname and the Host ID of a computer?

If you are running Windows OS 

1. Click on the 'Start' button.
2. Type 'cmd' into the search bar.
3. When the cmd prompt opens,
  - ❖ Type "**ipconfig/all**" (without the quotes).
4. Hit Enter on your keyboard.



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved

C:\Users\DAVE>ipconfig /all

Windows IP Configuration

Host Name . . . . . : WIN7-PC
Primary Dns Suffix . . . . . :
Node Type . . . . . : Hybrid
IP Routing Enabled. . . . . : No
WINS Proxy Enabled. . . . . : No

Ethernet adapter Local Area Connection:

Connection-specific DNS Suffix . :
Description . . . . . : Intel(R) 82579U Gigabit Ethernet Controller
Physical Address. . . . . : C8-60-00-A2-00-71
DHCP Enabled. . . . . : Yes
Autoconfiguration Enabled . . . . : Yes
Link-local IPv6 Address . . . . . : fe80::22c4:0a5e:1433:7978
IPv4 Address. . . . . : 192.168.88.1
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.88.1
DHCPv6 IAID . . . . . : 192.168.88.1
DHCPv6 Client DUID. . . . . : 01-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00

DNS Servers . . . . . : 192.168.88.1
NetBIOS over Tcpip. . . . . : Enabled
```

# Server vs. Host

10

A server:

- Can be a physical device or software program.
- Is installed on a host computer.
- Provides specific services.
- Serves only clients.

A host:

- Is always a physical computer or device.
- Can run both server and client programs.
- Provides specific services.
- Serves multiple users and devices.

# Client-Server Model

11

- ❑ A Client-Server Model is a [distributed application](#) structure that partitions tasks or workloads between the providers of a resource or service ([servers](#)), and service requesters ([clients](#)). Often, clients and servers communicate over a [computer network](#) on separate hardware, but both client and server may reside in the same system.
- ❑ A server [host](#) runs one or more server programs, which share their resources with clients. A client usually does not share any of its resources, but it requests content or service from a server. Clients, therefore, initiate communication sessions with servers, which await incoming requests. Examples of computer applications that use the client-server model are [email](#), network printing, and the [World Wide Web](#).
- ❑ A client-server network allows people to save their data centrally.

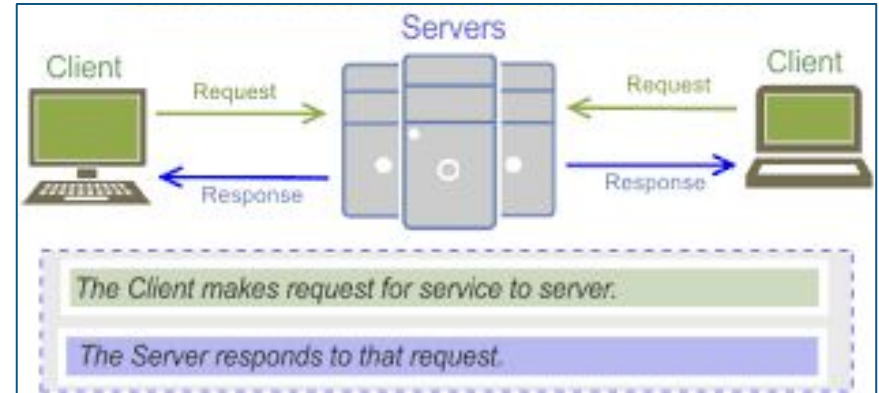
# Client-Server Architecture Overview

12

Client-Server Architecture is **a distributed system architecture where the workload of client servers are separated.**

Client Server architecture is a centralized resource system where the server contains all of the resources. The server is highly secured and scalable to respond to clients. **Client/Server Architecture** is a Service-Oriented Architecture that means client service will never be disrupted.

**Client/Server Architecture** reduces network traffic by responding to the queries of the clients instead of performing a complete file transfer. It replaces the file server with the database server. RDBMS is used by the server to answer the client's request directly.



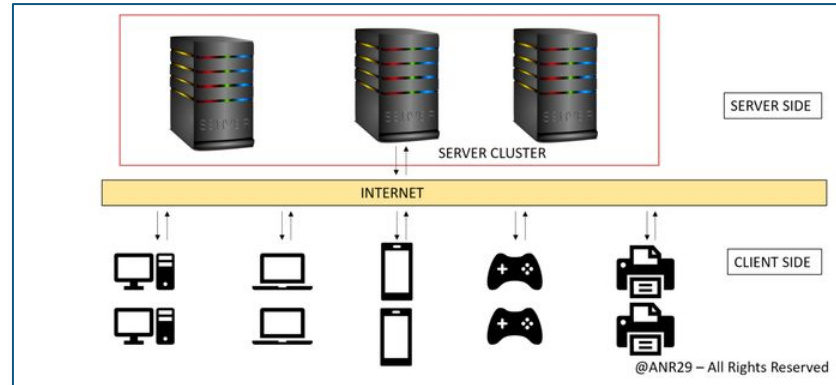
Source: computerhope.com

# Client-Server Architecture Components

13

Working on each architecture requires a few components. Similarly, the client-server architecture is based on four interconnected components:

1. Client (Workstations).
2. Servers.
3. Networking Devices: **Workstations and servers are interconnected with each other by means of a specific medium. Such a medium is called a *network device*.**
4. Other components such as scanners, printers, etc. can also be connected to network architecture.



# Types of Client Server Architecture

14

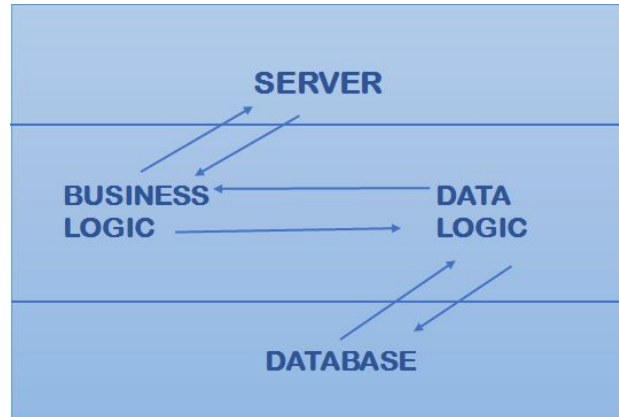
The functionality of client-server architecture is in various tiers. Below are the types where different layers operate in accordance with their type:

1. 1-Tier Architecture.
2. 2-Tier Architecture.
3. 3-Tier Architecture.
4. N-Tier Architecture.

# 1-Tier Architecture

15

In the **1-Tier** Architecture environment, the user interface, business logic, and data logic are present in same system. This kind of service is the cheapest but it is difficult to handle because of data inconsistency that allows for repetition of work.





# 2-Tier Architecture

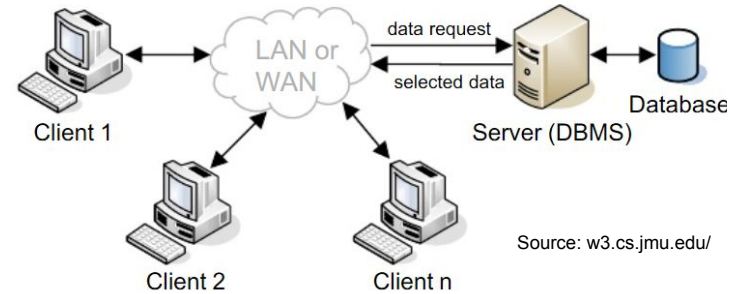
16

In the **2-Tier Architecture** environment, the user interface is stored at the client machine, and the database is stored on the server. Business Logic and Database Logic is stored at either the client or server but it must be unchanged.

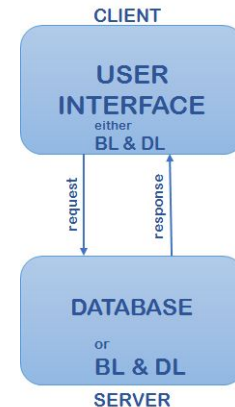
If Business Logic and Data Logic are stored at the client side, it is called *fat client*, *thin server* architecture. If Business Logic and Data Logic are stored on the server, it is called *thin client* *fat server* architecture.

2-Tier Architecture is useful where a client talks directly to a server; there is no intervening server. It is typically used in small environments where the user user interface is placed at the user's desktop environment and the DBMS services are usually placed in a server.

Information processing is split between the user system interface environment and the database management server environment.



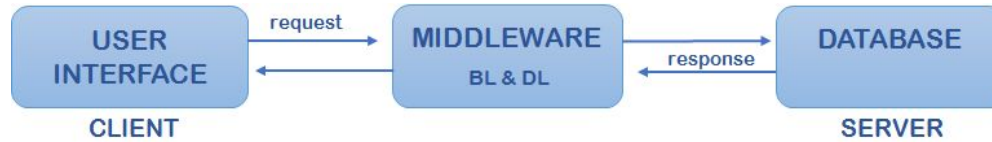
Source: w3.cs.jmu.edu/



Source: computerhope.com

# 3-Tier Architecture

17



Source: computerhope.com

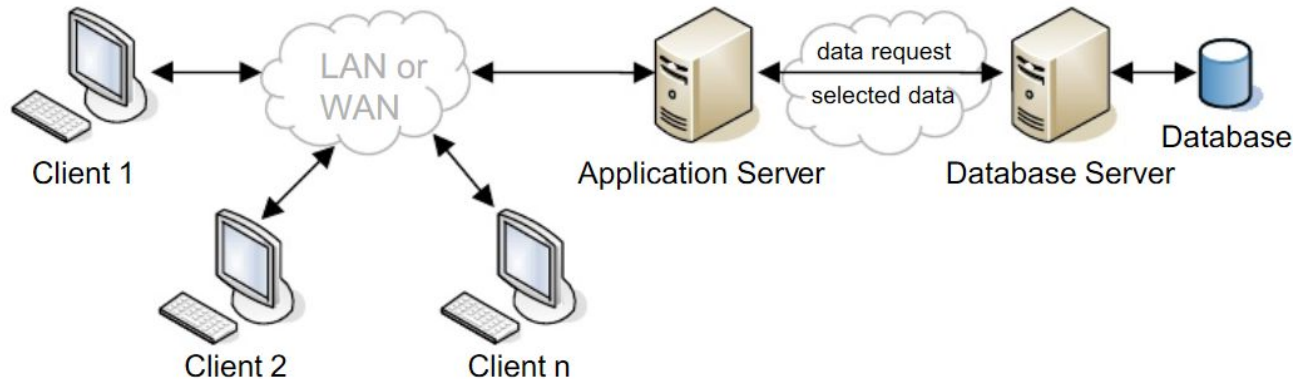
In a **3-Tier** environment, **middleware** is used. This means that the client request goes to the server through a middle layer, and the response of the server is firstly accepted by the middleware, and then to the client.

This architecture overcomes all of the drawbacks of the 2-Tier environment, and gives the best performance. Middleware stores all of the business logic and data access logic. If there are multiple business logic and data logic, it is called N-Tier Architecture. The purpose of middleware is for database staging, queuing, application execution, and scheduling, etc. Middleware can be a file server, message server, application server, transaction processing monitor, etc., and improves flexibility and gives the best performance. Middleware adds scheduling and prioritization for work in progress.

# 3-Tier Architecture (continued)

18

**3-Tier** Architecture is used to improve performance for large numbers of users. It also improves flexibility when compared to the 2-Tier approach, but the development environment is more difficult to use than 2-Tier applications.



# N-Tier Architecture

19

The 3-Tier Architecture can be broken into even more layers, and the broken layers may be able to run in more tiers. For example, an application layer can be broken into a business layer, a persistence layer, or more.

The N-Tier Architecture is also called Multi-Tier Architecture because the software is engineered to have the processing, data management, and presentation functions physically and logically separated. This means that these different functions are hosted on several machines or clusters, ensuring that services are provided without resources being shared; and as such, these services are delivered at top capacity. The “N” in N-Tier Architecture refers to any number from 1.

Not only does your software gain from being able to get services at the best possible rate, but it is also easier to manage. This is because when you work on one section, the changes you make will not affect the other functions. And if there is a problem, you can easily pinpoint where it originates.

# N-Tier Architecture (continued)

20

N-tier Architecture involves dividing an application into three different tiers:

## Presentation tier

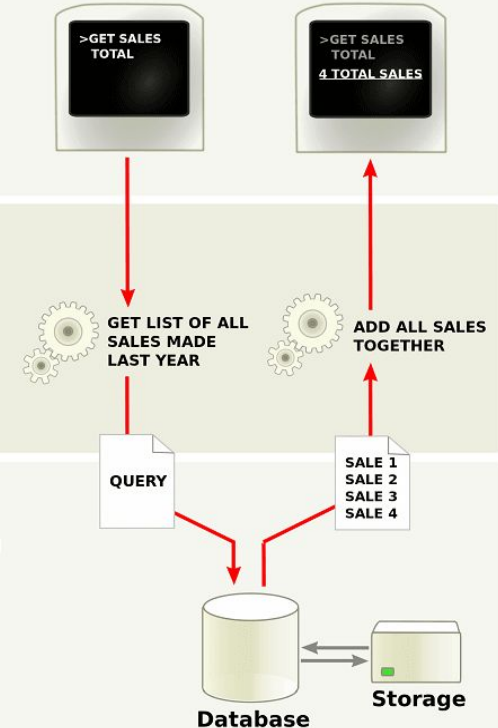
The top-most level of the application is the user interface. The main function of the interface is to translate tasks and results to something the user can understand.

## Logic tier

This layer coordinates the application, processes commands, makes logical decisions and evaluations, and performs calculations. It also moves and processes data between the two surrounding layers.

## Data tier

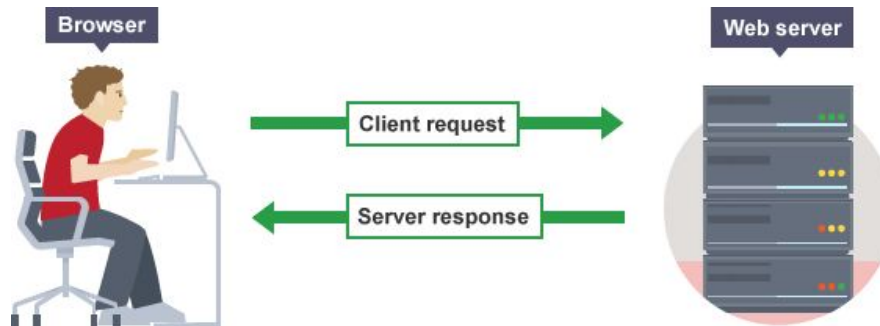
Here information is stored and retrieved from a database or file system. The information is then passed back to the logic tier for processing, and then eventually back to the user.



# Web Server

21

- ❑ A web server is a computer system that stores, processes, and delivers web pages to clients. The client is almost always a web browser or a mobile application. Depending on the setup, a web server can store one or more websites.
- ❑ Web Server only delivers static HTML content, such as documents, images, videos, and fonts. Traditionally, web servers do not deal with dynamic content or server-side programming.
- ❑ Web servers accept and fulfill Hypertext Transfer Protocol (HTTP or HTTPS) requests only. Optionally, you can add components for dealing with dynamic content.



# Web Server (continued)

22

Technically, you could host all files on your own computer, but it is far more convenient to store files all on a dedicated web server because:

- A dedicated web server is always up and running.
- Excluding downtime and systems troubles, a dedicated web server is always connected to the Internet.
- A dedicated web server can have the same IP address all the time. This is known as a dedicated IP address.
- A dedicated web server is typically maintained by a third-party.

For all of these reasons, finding a good hosting provider is key to building a website.



# Most Popular Web Servers

23

- ❖ Nginx.
- ❖ Apache HTTP Server.
- ❖ Microsoft IIS.
- ❖ LiteSpeed.
- ❖ Jetty.

# Application Server

24

An **application server** is a **software framework** that delivers content and assets for a client application. It provides clients with access to business logic. Through business logic, an application server transforms data into dynamic content and enables the functionality of the application. Dynamic content includes:

- A transaction result.
- Decision support.
- Real-time analytics.

This server type is the main link between a client and the server-side code. Typical tasks include:

- Transaction management.
- Security.
- Dependency injection (DI)
- Concurrency.

An application server is designed to install, operate, and host applications and associated services for end-users.

# System Administration

25

System administration of a Application Server environment includes tasks such as:

- Creating Application Server domains.
- Deploying applications.
- Migrating domains from development environments to production environments.
- Monitoring and configuring the performance of the Application Server domains.
- Diagnosing and troubleshooting problems.

# Most Popular Application Servers

26

- ❖ Apache Tomcat.
- ❖ Oracle WebLogic.
- ❖ Glassfish.
- ❖ JBoss.
- ❖ Websphere.

Use a Application Server like **Apache Tomcat** if you have an application that requires JSP and Servlet.

Use a full-blown application server such as JBoss or Oracle WebLogic if you have an application with complex features such as distributed transactions and messaging.

# Server-Side Script vs. Client-Side Script

27

When a client (your computer) makes a request for a web page that information is processed by the web server. If the request is a server side script (e.g. Java, or PHP), before the information is returned to the client, the script is executed on the server and the results of the script is returned to the client.



Server-Side Script

Once the client receives the returned information from the server, if it contains a client-side script (e.g. JavaScript), your computer browser executes that script before displaying the web page.

Client-Side Script



# Knowledge Check

28

1. What is a Client?
2. What is a Server?
3. What are the main differences between a Client and a Server?
4. What is a Web server?
5. What is an Application Server?

# Lesson 2: Middleware

## Learning Objectives:

By the end of this presentation, learners will be able to:

- Describe Middleware.
- Describe Middleware functions.



# Agenda/Topics

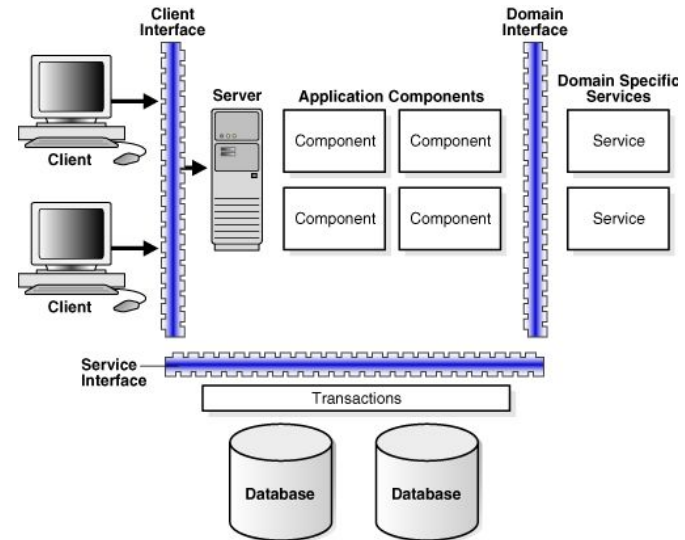
30

- ❑ Middleware.
- ❑ Middleware Functions.
- ❑ Middleware Types.
- ❑ Middleware Deployment Topology.
- ❑ Typical Enterprise Deployment Topology.

# Middleware

31

- ❑ Middleware is the software that connects software components or enterprise applications.
- ❑ Middleware is the software layer that lies between the **operating system** and the **applications** on each side of a distributed computer network. Typically, it supports complex, distributed business software applications.
- ❑ Can be described as "software glue."



Click here to Watch Video:  
<https://youtu.be/IFzdkDnYHac>

# Middleware Functions

32

- ❖ Hides the distributed nature of the application-
  - An application represents a collection of interconnected parts that are operational and running in distributed locations, out of view.
- ❖ Hides the heterogeneity of the enterprise-
  - This includes the hardware components, computer operating systems, and communication protocols.
- ❖ Provides uniform, standard, high-level interfaces-
  - Applications can be easily composed, reused, ported, and made to interoperate.
- ❖ Supplies a set of common services-
  - Performs various general purpose functions to avoid duplicating efforts, and to facilitate collaboration between applications.

Middleware makes application development easier, by providing common programming abstractions by masking application heterogeneity and the distribution of the underlying hardware and operating systems, and by hiding low-level programming details.

# Middleware Types

33

“Middleware” is a broad term covering different types of middleware based on the various functionalities they provide:

- ❖ **Database middleware:** Enables access and interaction with different database gateways (most common).
- ❖ **Message-oriented middleware:** Allows software applications across multiple operating systems and networking protocols to receive and send messages to each other.
- ❖ **Portals:** Used by enterprises to facilitate interactions between client-facing systems and backend systems.
- ❖ **Enterprise application integration:** A virtual layer that connects applications, data, processes, and services, irrespective of where they are located ([cloud or on-premise](#)) and what technology they are based on.
- ❖ **Transactional middleware:** Ensures that transactions between heterogeneous components go through all of the necessary steps to reach completion.

# Middleware Types (continued)

34

- ❖ **Content middleware:** Used to abstract specific content that is similar to publish and subscribe models.
- ❖ **Remote procedure call (RPC) middleware:** Allows one application to trigger a function in another application, whether they reside in the same network or not.
- ❖ **Device middleware:** Contains a specific set of capabilities to integrate with, or build applications on specific devices. It is usually used to build mobile applications.
- ❖ **Platform middleware:** Allows business logic to be seated in any platform (OS or hardware), including web servers, application servers, hosting environments, or containers.
- ❖ **Application server middleware:** Allows the creation, deployment, and maintenance of enterprise applications within a system.
- ❖ **Web middleware:** Allows companies to integrate more easily with individual backend systems, just like the ecommerce example mentioned earlier.

# Middleware Types (continued)

35

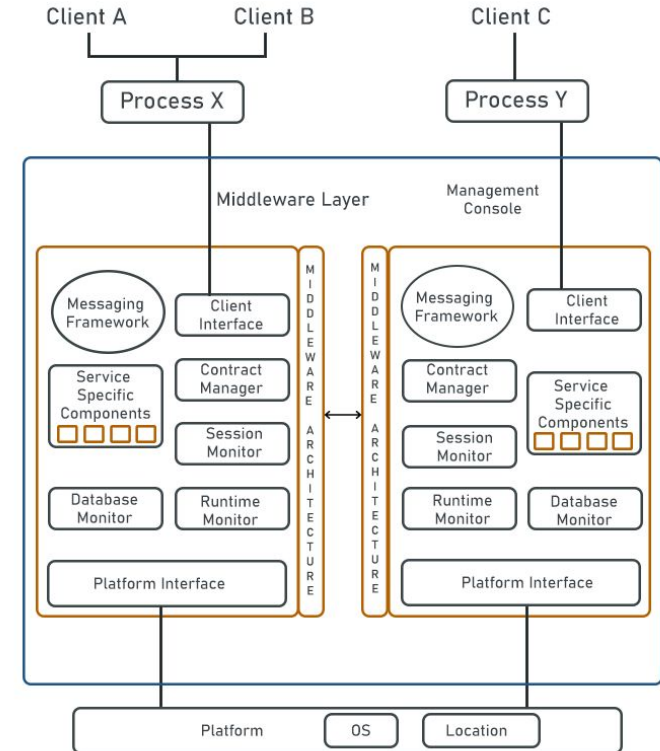
- ❖ **Robotics middleware:** Simplifies the integration of robotic hardware, firmware, and software, irrespective of manufacturer and location.
- ❖ **Cloud middleware:** Sits between the operating systems behind the cloud and the cloud users, providing a remote platform to create, maintain, and communicate with the hosted applications and data.

Most businesses today use at least one of these types of middleware for their everyday operations. Large enterprises often require a combination of these to be financially, operationally, and technologically savvy.

# Middleware Deployment Topology

36

The most common example of Middleware in action is an online shopping website. When users shop, they browse through catalogs, get personalized recommendations, buy desired products, receive intimations about delivery status, and finally, receive their products. This is all done with the help of Middleware. Middleware connects the payment, accounting, production shipping, and notification systems, providing the shopper with a seamless user experience.





# Typical Enterprise Deployment Topology

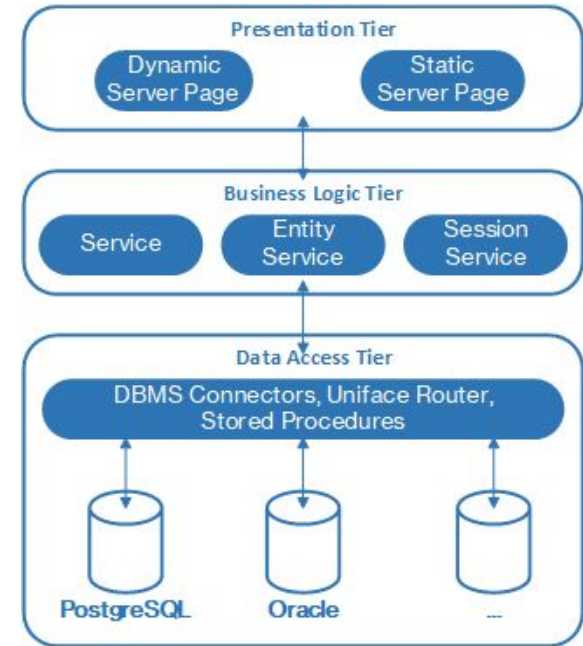
37

The enterprise deployment topology of Middleware consists of the following high-level elements, which is also called **N-tier client-server architecture**.

- **Presentation Tier** (also called **Web Tier**), consists of two or more physical computers that are hosting web server instances.
- **Business Logic Tier** (also called **Application Tier**) consists of two or more physical computers that are hosting a cluster of Application Servers, and the Administration Server for the domain. The Application Servers are configured to run the various Middleware products, (e.g., Oracle SOA Suite and Oracle Service Bus).
- **Data Access Tier** consists of two or more physical hosts. The data access tier includes the data persistence mechanisms (database servers, file shares, etc.).

As with the separation of any tier, there are costs for implementation and costs of performance in exchange for improved scalability and maintainability.

[https://en.wikipedia.org/wiki/Multitier\\_architecture](https://en.wikipedia.org/wiki/Multitier_architecture)



# Knowledge Check

38

- ❑ What is Middleware?
- ❑ What is Typical Enterprise Deployment Topology?

# Lesson 3: Database

## Learning Objectives:

By the end of this presentation, learners will be able to:

- Describe the Database.
- Define the Database Management System (DBMS).
- Explain the challenges of a Database.

# Agenda/Topics

40

- ❑ A Database.
- ❑ Database Management System (DBMS).
- ❑ Database Components.
- ❑ Database Types.
- ❑ Database Challenges.

# A Database

41

A database is a systematic collection of data, and supports electronic storage and manipulation of data. Databases make data management easy.



- ❑ An example: An online telephone directory uses a database to store data of people, phone numbers, and other contact details. Your electricity service provider uses a database to manage billing and client-related issues, as well as to handle fault data, etc.
- ❑ Consider Facebook. Facebook needs to store, manipulate, and present data, which is related to members, their friends, activities, messages, advertisements, and a lot more. We can provide a countless number of examples for the usage of databases.

# Database Management System

42

A Database Management System (DBMS) features:

- ❑ Is software that enables users to create and manage a database.
- ❑ Provides physical and logical independence from data. Users and applications do not need to know either the physical or logical locations of data. A DBMS can limit and control access to the database and provide different views of the same database schema to multiple users.
- ❑ Offers data integrity and security.
- ❑ Implies integrity constraints to get a high level of protection against prohibited access to data.

# Database Components

43

- ❑ **Hardware:** The hardware consists of physical, electronic devices like computers, I/O devices, storage devices, etc. This offers the interface between computers and real-world systems.
- ❑ **Software:** Software is a set of programs used to manage and control the overall database. This includes the database software itself, the Operating System (OS), the network software used to share the data among users, and the application programs for accessing data in the database.
- ❑ **Data:** Data is a raw and unorganized that is required to be processed to make it meaningful. Data can be simple and unorganized at the same time, unless it is organized. Generally, data comprises facts, observations, perceptions, numbers, characters, symbols, images, etc.
- ❑ **Procedure:** The procedure is a set of instructions and rules that help you to use the DBMS. It is designing and running the database using documented methods, which allow you to guide the users who operate and manage it.
- ❑ **Database Access Language:** Database Access language is used to access the data to and from the database, enter new data, update already existing data, or retrieve required data from DBMS. The user writes some specific commands in a database access language and submits these to the database.

# Database Types

44

1. **Hierarchical databases:** support “parent-child” relationships of storing data. Its structure is like a tree.
2. **Network databases:** support many-to-many relationships. It usually results in complex database structures.
3. **Object-oriented databases:** data is stored in the form of objects.
4. **Relational databases:** defines database relationships in the form of tables.
5. **NoSQL databases:** data is stored in documents rather than relational tables.
6. **Graph databases:** uses graph theory to store, map, and query relationships.
7. **Distributed databases:** data is not in one place and is distributed to various organizations.



# Database Challenges

45

Setting up, operating, and maintaining a database includes challenges; the most common challenges include:

- **Data security** is required because data is a valuable business asset. Protecting data stores requires a skilled **cybersecurity** staff; this can be costly.
- **Data integrity** ensures data is trustworthy. It is not always easy to achieve **data integrity** because it means restricting access to databases to only those qualified to handle it.
- **Database performance** requires regular database updates and maintenance. Without the proper support, database functionality can decline as the technology, which supports the database can change, or the data itself can contain changes.
- **Database integration** can also be difficult because it can involve **integrating data sources** from varying types of databases and structures into a single database or data warehouse.

# Knowledge Check

46

- What is Data?
- Describe a Database?

# Summary

47

- ❑ The Client is a computer that requests service from the server. The type of services that the client requests is printing, making a call, or fetching data from a website.
- ❑ A server is a computer that provides services to the client computer. A server might be a web server of Google, a print server, a database server, a file server, or fax server.
- ❑ Client-server Architecture is the architecture of a computer network, which many clients (remote processors) request and receive service from a centralized server (host computer).
- ❑ Middleware is the software layer that lies between the operating system and the applications on each side of a distributed computer network. Typically, it supports complex, distributed business software applications.
- ❑ A database is a systematic collection of data, which supports electronic storage and the manipulation of data. Databases make data management easy.

# Glossary/Acronyms

48

- ❑ OS: Operating System.
- ❑ DB: Database.
- ❑ HAL: Hardware Abstraction Layer.
- ❑ Servers: Fast processing devices.
- ❑ Workstations: Workstations are also called client computers.
- ❑ Networking Devices: Workstations and servers that are interconnected.
- ❑ DBMS: Database Management System.
- ❑ RDBMS: Relational Database Management System.

# References

49

[https://en.wikipedia.org/wiki/Client%E2%80%93server\\_model](https://en.wikipedia.org/wiki/Client%E2%80%93server_model)

<https://www.techtarget.com/searchdatamanagement/definition/database>

Apache®, Apache Tomcat®, Apache Kafka®, Apache Cassandra™, and Apache Geode™ are trademarks or registered trademarks of the Apache Software Foundation in the United States and/or other countries. Java™, Java™ SE, Java™ EE, and OpenJDK™ are trademarks of Oracle and/or its affiliates. Kubernetes® is a registered trademark of the Linux Foundation in the United States and other countries. Linux® is the registered trademark of Linus Torvalds in the United States and other countries. Windows® and Microsoft® Azure are registered trademarks of Microsoft Corporation. “AWS” and “Amazon Web Services” are trademarks or registered trademarks of Amazon.com Inc. or its affiliates. All other trademarks and copyrights are property of their respective owners and are only mentioned for informative purposes. Other names may be trademarks of their respective owners.