**We have**

**1) Two tier architecture:**  
**2) Three tier architecture:**  
**3)N- tier architecture:**

**Two tier architecture:**  
Two tier architecture is similar to a basic **client-server** model. The application at the client end directly communicates with the database at the server side. API’s like ODBC,JDBC are used for this interaction. The server side is responsible for providing query processing and transaction management functionalities. On the client side, the user interfaces and application programs are run. The application on the client side establishes a connection with the server side in order to communicate with the DBMS.  
An advantage of this type is that maintenance and understanding is easier, compatible with existing systems. However this model gives poor performance when there are a large number of users.

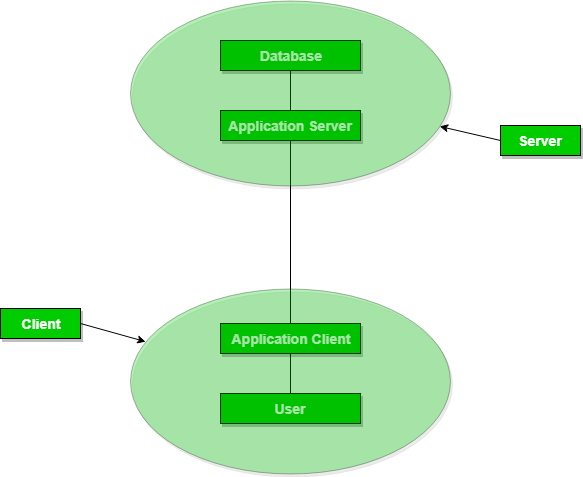


**Three Tier architecture:**  
In this type, there is another layer between the client and the server. The client does not directly communicate with the server. Instead, it interacts with an application server which further communicates with the database system and then the query processing and transaction management takes place. This intermediate layer acts as a medium for exchange of partially processed data between server and client. This type of architecture is used in case of large web applications.  
**Advantages:**

* **Enhanced scalability** due to distributed deployment of application servers. Now, individual connections need not be made between client and server.
* **Data Integrity** is maintained. Since there is a middle layer between client and server, data corruption can be avoided/removed.
* **Security** is improved. This type of model prevents direct interaction of the client with the server thereby reducing access to unauthorized data.

**Disadvantages**:  
Increased complexity of implementation and communication. It becomes difficult for this sort of interaction to take place due to presence of middle layers.

**Three-Tier Architecture**



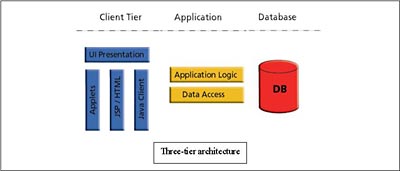
**3-tier application architecture**

A 3-tier application architecture is a modular client-server architecture that consists of a presentation tier, an application tier and a data tier. The data tier stores information, the application tier handles logic and the presentation tier is a graphical user interface ([GUI](https://searchwindevelopment.techtarget.com/definition/GUI)) that communicates with the other two tiers. The three tiers are logical, not physical, and may or may not run on the same physical server.

Presentation tier - This tier, which is built with [HTML5](https://whatis.techtarget.com/definition/HTML5), cascading style sheets ([CSS](https://www.theserverside.com/definition/cascading-style-sheet-CSS)) and [JavaScript](https://www.theserverside.com/definition/JavaScript), is deployed to a computing device through a web browser or a web-based application. The presentation tier communicates with the other tiers through application program interface ([API](https://searchapparchitecture.techtarget.com/definition/application-program-interface-API)) calls.

Application tier - The application tier, which may also be referred to as the logic tier, is written in a programming language such as [Java](https://www.theserverside.com/definition/JavaScript) and contains the [business logic](https://whatis.techtarget.com/definition/business-logic) that supports the application’s core functions. The underlying application tier can either be hosted on distributed servers in the cloud or on a dedicated in-house server, depending on how much processing power the application requires.

Data tier - The data tier consists of a database and a program for managing read and write access to a database. This tier may also be referred to as the storage tier and can be hosted on-premises or in the cloud. Popular database systems for managing read/write access include [MySQL](https://searchoracle.techtarget.com/definition/MySQL), PostgreSQL, Microsoft SQL Server and [MongoDB](https://searchdatamanagement.techtarget.com/definition/MongoDB).



**Benefits of a 3-tier app architecture**

The benefits of using a 3-layer architecture include improved [horizontal scalability](https://searchcio.techtarget.com/definition/horizontal-scalability), performance and availability. With three tiers, each part can be developed concurrently by different team of programmers coding in different languages from the other tier developers. Because the programming for a tier can be changed or relocated without affecting the other tiers, the 3-tier model makes it easier for an enterprise or software packager to continually evolve an application as new needs and opportunities arise. Existing applications or critical parts can be permanently or temporarily retained and encapsulated within the new tier of which it becomes a component.

3-tier Architecture

A 3-tier architecture separates its tiers from each other based on the complexity of the users and how they use the data present in the database. It is the most widely used architecture to design a DBMS.



* **Database (Data) Tier** − At this tier, the database resides along with its query processing languages. We also have the relations that define the data and their constraints at this level.
* **Application (Middle) Tier** − At this tier reside the application server and the programs that access the database. For a user, this application tier presents an abstracted view of the database. End-users are unaware of any existence of the database beyond the application. At the other end, the database tier is not aware of any other user beyond the application tier. Hence, the application layer sits in the middle and acts as a mediator between the end-user and the database.
* **User (Presentation) Tier** − End-users operate on this tier and they know nothing about any existence of the database beyond this layer. At this layer, multiple views of the database can be provided by the application. All views are generated by applications that reside in the application tier.

Multiple-tier database architecture is highly modifiable, as almost all its components are independent and can be changed independently.