SNS COLLEGE OF TECHNOLOGY,COIMBATORE-35

(AN AUTONOMOUS INSTITUTION)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

19CSO303-Web Designing(Open Elective)

UNIT-IV

Introduction to PHP – Origin and Uses of PHP – Overview -Characteristics–Primitives, Operations, and Expressions–Output–Control Statements –Arrays–Functions–Pattern Matching –Form Handling – Files–Session Tracking–Database Access Through the Web–Relational Databases–SQL–Architectures for Database Access –MySQL Database System.

Introduction to PHP:

PHP started out as a small open source project that evolved as more and more people found out how useful it was. Rasmus Lerdorf unleashed the first version of PHP way back in 1994.

- PHP is a recursive acronym for "PHP: Hypertext Preprocessor".
- PHP is a server side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking, even build entire e-commerce sites.
- It is integrated with a number of popular databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server.
- PHP is pleasingly zippy in its execution, especially when compiled as an Apache module on the Unix side. The MySQL server, once started, executes even very complex queries with huge result sets in record-setting time.
- PHP supports a large number of major protocols such as POP3, IMAP, and LDAP. PHP4 added support for Java and distributed object architectures (COM and CORBA), making n-tier development a possibility for the first time.
- PHP is forgiving: PHP language tries to be as forgiving as possible.
- PHP Syntax is C-Like.

Database Access Through the Web:

ASP.NET allows the following sources of data to be accessed and used:

- Databases (e.g., Access, SQL Server, Oracle, MySQL)

- XML documents
- Business Objects
- Flat files

ASP.NET hides the complex processes of data access and provides much higher level of classes and objects through which data is accessed easily. These classes hide all complex coding for connection, data retrieving, data querying, and data manipulation.

ADO.NET is the technology that provides the bridge between various ASP.NET control objects and the backend data source. In this tutorial, we will look at data access and working with the data in brief.

Retrieve and display data

It takes two types of data controls to retrieve and display data in ASP.NET:

- **A data source control -** It manages the connection to the data, selection of data, and other jobs such as paging and caching of data etc.
- **A data view control -** It binds and displays the data and allows data manipulation.

We will discuss the data binding and data source controls in detail later. In this section, we will use a SqlDataSource control to access data and a GridView control to display and manipulate data in this chapter.

We will also use an Access database, which contains the details about .Net books available in the market. Name of our database is ASPDotNetStepByStep.mdb and we will use the data table DotNetReferences.

Relational DB:

A relational database is a type of database that stores and provides access to data points that are related to one another. Relational databases are based on the relational model, an intuitive, straightforward way of representing data in tables. In a relational database, each row in the table is a record with a unique ID called the key. The columns of the table hold attributes of the data, and each record usually has a value for each attribute, making it easy to establish the relationships among data points.

SQL:

SQL tutorial provides basic and advanced concepts of SQL. Our SQL tutorial is designed for both beginners and professionals.

SQL (Structured Query Language) is used to perform operations on the records stored in the database, such as updating records, inserting records, deleting records, creating and modifying database tables, views, etc.

SQL is not a database system, but it is a query language.

Suppose you want to perform the queries of SQL language on the stored data in the database. You are required to install any database management system in your systems, for example, Oracle, MySQL, MongoDB, PostgreSQL, SQL Server, DB2, etc.

Architecture of database Access:

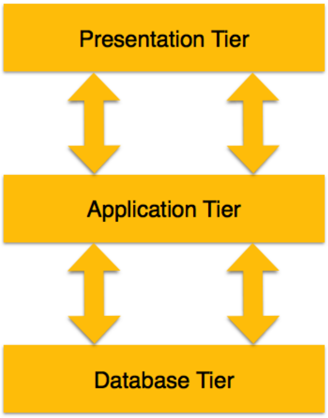
The design of a DBMS depends on its architecture. It can be centralized or decentralized or hierarchical. The architecture of a DBMS can be seen as either single tier or multi-tier. An n-tier architecture divides the whole system into related but independent **n** modules, which can be independently modified, altered, changed, or replaced.

In 1-tier architecture, the DBMS is the only entity where the user directly sits on the DBMS and uses it. Any changes done here will directly be done on the DBMS itself. It does not provide handy tools for end-users. Database designers and programmers normally prefer to use single-tier architecture.

If the architecture of DBMS is 2-tier, then it must have an application through which the DBMS can be accessed. Programmers use 2-tier architecture where they access the DBMS by means of an application. Here the application tier is entirely independent of the database in terms of operation, design, and programming.

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.

MySQL Database System:

A MySQL DB system is a logical container for the MySQL instance. It provides an interface enabling management of tasks such as provisioning, backup and restore, monitoring, and so on. It also provides a read/write endpoint enabling you to connect to the MySQL instance using the standard protocols.

A MySQL Database Service DB System consists of a compute instance, operating system, MySQL Server, Virtual Network Interface Card (VNIC), and block storage.

- A compute instance (with resources defined by the associated shape.

- Oracle Linux Operating System.
- MySQL Server Enterprise Edition 8.0.
- Virtual Network Interface Card (VNIC) that attaches the DB system to a subnet of the Virtual Cloud Network (VCN).
- Network-attached block storage.
- MySQL Database Service uses the **Higher Performance** option for all block storage.