



# ALAGAPPA UNIVERSITY

[Accredited with 'A+' Grade by NAAC (CGPA:3.64) in the Third Cycle  
and Graded as Category-I University by MHRD-UGC]

**KARAIKUDI - 630 003**

**DIRECTORATE OF DISTANCE EDUCATION**



OPEN SOURCE SOFTWARE

## Diploma in Computer Applications

**517 12**

K 4  
0100011  
E  
M 001001

Diploma in Computer Applications

## OPEN SOURCE SOFTWARE

I - Semester



**ALAGAPPA UNIVERSITY**

[Accredited with 'A+' Grade by NAAC (CGPA:3.64) in the Third Cycle and Graded as Category-I University by MHRD-UGC]

(A State University Established by the Government of Tamil Nadu)



**KARAIKUDI – 630 003**

**Directorate of Distance Education**

**Diploma in Computer Applications**

**I - Semester**

**517 12**

**OPEN SOURCE SOFTWARE**

**Edited By**

Dr P.Prabhu

Assistant Professor in Information Technology  
Directorate of Distance Education,  
Alagappa University,  
Karaikudi.

---

# SYLLABI-BOOK MAPPING TABLE

## Open Source Software

---

Syllabi	Mapping in Book
<b>UNIT 1:</b> <b>INTRODUCTION</b> <b>Introduction to Open Sources:</b> Need of Open Sources- Advantages of Open Sources-Applications of Open sources	Introduction (Pages 1-7)
<b>UNIT II:</b> <b>Open Source Operating System: LINUX:</b> Introduction- General Overview-Kernel Mode and user mode-Process <b>Advanced Concepts:</b> Scheduling-Personalities-Cloning- Signals-Development with Linux	Open source Operating System (Pages 8-15) Advanced Concepts (Pages 16-26)
<b>UNIT III:</b> <b>OPEN SOURCE DATABASE</b> <b>MySQL:</b> Introduction -Setting up Account Starting, terminating and writing your Own SQL programs <b>Record selection Technology-</b> Working with strings-Date and Time	MySQL (Pages 27-35) Record selection Technology (Pages 36-46)
<b>UNIT IV:</b> <b>Sorting query Results-</b> Generating Summary-Working with Metadata- Using Sequences- My SQL and Web	Sorting query Results (Pages 47-55)
<b>UNIT V:</b> <b>OPEN SOURCE PROGRAMMING PHP</b> <b>PHP:</b> Introduction- Programming in Web Environment- Variables-Constants-Data types-Operators- Statements <b>Functions:</b> Arrays-OOP- String Manipulation and Regular Expression-File Handling and Data Storage	PHP (Pages 56-71) Functions (Pages 72-87)
<b>UNIT VI:</b> <b>PHP and MySQL database:</b> PHP and LDAP- PHP Connectivity- Sending and Receiving E mails- Debugging and Error Handling-Security-Templates	PHP and MySQL database (Pages 88-105)

---

## CONTENTS

### UNIT I:

INTRODUCTION	1-7
1.0 Introduction	
1.1 Objective	
1.2 Software Terminologies	
1.3 Need of Open Sources	
1.4 Advantages of Open Sources	
1.5 Applications of Open sources	
1.6 Answers to Check Your Progress	
1.7 Summary	
1.8 Keywords	
1.9 Self Assessment Questions and Exercises	
1.10 Further Readings	

### UNIT II

OPEN SOURCE OPERATING SYSTEMS	8-15
2.0 Introduction	
2.1 Objective	
2.2 Open Source Operating System	
2.3 General Overview	
2.4 Kernel Mode	
2.5 User Mode	
2.6 Process	
2.7 Answers to Check Your Progress	
2.8 Summary	
2.9 Keywords	
2.10 Self Assessment Questions and Exercises	
2.11 Further Readings	

## ADVANCED CONCEPTS

16-26

- 3.0 Introduction
- 3.1 Objective
- 3.2 Scheduling Concepts
- 3.3 Signals
- 3.4 Process
- 3.5 Cloning
- 3.6 Personalities
- 3.7 Development with Linux
- 3.8 Answers to Check Your Progress
- 3.9 Summary
- 3.10 Keywords
- 3.11 Self Assessment Questions and Exercises
- 3.12 Further Readings

## UNIT III:

### MySQL

27-35

- 4.0 Introduction
- 4.1 Objective
- 4.2 MySQL
- 4.3 Setting up Account
- 4.5 Answers to Check Your Progress
- 4.6 Summary
- 4.7 Keywords
- 4.8 Self Assessment Questions and Exercises
- 4.9 Further Readings

## RECORD SELECTION TECHNOLOGY

36-46

- 5.0 Introduction
- 5.1 Objective
- 5.2 Record selection Technology
- 5.3 Working with Strings
- 5.4 Date and Time Functions

- 5.5 Answers to Check Your Progress
- 5.6 Summary
- 5.7 Keywords
- 5.8 Self Assessment Questions and Exercises
- 5.9 Further Readings

#### **UNIT IV**

#### **WORKING WITH SQL 47-55**

- 6.0 Introduction
- 6.1 Objective
- 6.2 Sorting query Results
- 6.3 Generating Summary
- 6.4 Working with Metadata
- 6.5 Using Sequences
- 6.6 My SQL and Web
- 6.7 Answers to Check Your Progress
- 6.8 Summary
- 6.9 Keywords
- 6.10 Self Assessment Questions and Exercises
- 6.11 Further Readings

#### **UNIT V:**

#### **PHP 56-71**

- 7.0 Introduction
- 7.1 Objective
- 7.2 PHP
- 7.3 Programming in Web Environment
- 7.4 PHP Variables
- 7.5 Constants
- 7.6 Data Types
- 7.7 Operators
- 7.8 Statement
- 7.9 Answers to Check Your Progress
- 7.10 Summary

- 7.11 Keywords
- 7.12 Self Assessment Questions and Exercises
- 7.13 Further Readings

## **FUNCTIONS**

72-87

- 8.0 Introduction
- 8.1 Objective
- 8.2 Functions
- 8.3 Arrays
- 8.4 Object Oriented Concepts
- 8.5 String Manipulation and Regular Expression
- 8.6 File Handling and Data Storage
- 8.7 Answers to Check Your Progress
- 8.8 Summary
- 8.9 Keywords
- 8.10 Self Assessment Questions and Exercises
- 8.11 Further Readings

## **UNIT VI**

### **PHP AND MySQL DATABASE**

88-105

- 9.0 Introduction
- 9.1 Objective
- 9.2 PHP and MySQL
- 9.3 PHP and LDAP
- 9.4 PHP Connectivity
- 9.5 Sending and Receiving E-mails
- 9.6 Debugging and Error Handling
- 9.7 Templates
- 9.8 Answers to Check Your Progress
- 9.9 Summary
- 9.10 Keywords
- 9.11 Self Assessment Questions and Exercises
- 9.12 Further Readings



---

# BLOCK I

## INTRODUCTION

---

# UNIT 1

## INTRODUCTION

---

### NOTES

### Structure

- 1.0 Introduction
- 1.1 Objective
- 1.2 Software Terminologies
- 1.3 Need of Open Sources
- 1.4 Advantages of Open Sources
- 1.5 Applications of Open sources
- 1.6 Answers to Check Your Progress
- 1.7 Summary
- 1.8 Keywords
- 1.9 Self Assessment Questions and Exercises
- 1.10 Further Readings

---

### 1.0 INTRODUCTION

---

This unit explains the open source software concepts and basics of the system with which we work .The applications and need of this software and how it is freely available is discussed.

---

### 1.1 OBJECTIVE

---

This unit briefs the open source software needs and makes the user to understand and learn the following concepts

- Need of open sources
- Applications
- Advantages

---

### 1.2 SOFTWARE TERMINOLOGIES

---

Some of the common terms used in the field of software.

#### 1.2.1 Public Domain Software

Public domain software refers to any program that is not copy righted. This software is free and can be used without restrictions, that is, the user can copy, distribute, and even modify the software without obtaining permission from the software developer.

### **1.2.2 Freeware**

The term freeware is commonly used for copyrighted software given away free by its author. It is available for free but the author retains the copyright, which means that a user does not have the right to modify anything in the software that is not explicitly allowed by the developer. Thus, freeware software permits re-distribution but not modification.

### **1.2.3 Shareware**

Shareware is the software which comes with permission for people to redistribute copies for a limited period. Anyone who continues to use a copy is required to pay a license fee. Therefore, a free use of the software is usually limited to a period. It is distributed without payment ahead of time.

### **1.2.4 Firmware**

Firmware is a combination of software, permanently stored in the memory. As the name suggests, firmware is a program or data that has been written onto read only memory (ROM).

### **1.2.5 Proprietary Software**

Proprietary describes a technology or product that is owned exclusively by a single company that carefully guards knowledge about the technology or the product's internal working. Proprietary software is also called as Closed Source Software. Its use, redistribution or modification is prohibited or is restricted so much that the user effectively cannot use it freely.

### **1.2.6 Open Source Software**

It is created by generous programmers and released into the public domain for public use. The underlying programming code is available to the users so that they may read it, make changes to it, and build new versions of the software incorporating their changes for software. Usually this software is distributed under an open source license-GPL.

Open source doesn't just mean access to the source code. The distribution terms of open-source software must obey with the following criteria:

#### **1. Free Redistribution**

The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale

#### **2. Source Code**

The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost preferably, downloading via the Internet without charge. The source code

must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.

### **3. Derived Works**

The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

### **4. Integrity of the Author's Source Code**

The license may restrict source-code from being distributed in modified form only if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

### **5. No Discrimination against Persons or Groups**

The license must not discriminate against any person or group of persons.

### **6. No Discrimination against Fields of Endeavor**

The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

### **7. Distribution of License**

The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

### **8. License Must Not Be Specific to a Product**

The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

### **9. License Must Not Restrict Other Software**

The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.

### **10. License Must Be Technology-Neutral**

No provision of the license may be predicated on any individual technology or style of interface.

NOTES

---

### 1.3 NEED OF OPEN SOURCES

---

The name “FOSS” is a recursive acronym for “Free Open Source Software”. Free software is a matter of the users’ freedom to run, copy, distribute, study, change and improve the software. More precisely, it refers to four kinds of freedom, for the users of the software:

1. The freedom to run the program, for any purpose
2. The freedom to study how the program works, and change it to make it do what you wish.
3. The freedom to redistribute copies so you can help your neighbour.
4. The freedom to improve the program, and release your improvements (and modified versions in general) to the public, so that the whole community benefits.

Free software is software that gives you the user the freedom to share study and modify it. We call this free software because the user is free. To use free software is to make a political and ethical choice asserting the right to learn, and share what we learn with others. Free software has become the foundation of a learning society where we share our knowledge in a way that others can build upon and enjoy.

Currently, many people use proprietary software that denies users these freedoms and benefits. If we make a copy and give it to a friend, if we try to figure out how the program works, if we put a copy on more than one of our own computers in our own home, we could be caught and fined or put in jail. That’s what’s in the fine print of the license agreement you accept when using proprietary software.

The corporations behind proprietary software will often spy on your activities and restrict you from sharing with others. And because our computers control much of our personal information and daily activities, proprietary software represents an unacceptable danger to a free society.

---

### 1.4 ADVANTAGES OF OPEN SOURCES

---

- Reliability
- Security
- Combats Piracy
- Total Cost of Ownership
- Non Quantitative Issues
- Freedom from control by another
- Protection from licensing litigation
- Flexibility
- Social/Moral/Ethical Issues
- Innovation

---

## 1.5 APPLICATIONS OF OPEN SOURCES

---

- Active user groups of GNU/Linux in various Indian cities
- Goa Schools Computers Project – GSCP – A collaborative effort
- Localisation of GNU/Linux in Tamil, Hindi, Gujrathi, Bengali and Punjabi
- Contributions to KDE, GNOME
- Anjuta -IDE for C & C++ on GNU/Linux by Naba Kumar
- Mayavi – Scientific Data Visualizer
- KGDB – Kernel patch to debug the Linux Kernel
- National Resource Center for Free Open Source Software-(NRCFOSS) jointly implemented by AU-KBC Centre, MIT Campus.

### NOTES

#### Check your Progress

1. What is Public Domain Software?
2. What is Freeware?
3. What is Firmware?
4. What is FOSS?
5. What are the Advantages of Open Sources?

---

## 1.6. ANSWERS TO CHECK YOUR PROGRESS

---

1. Public domain software refers to any program that is not copy righted. This software is free and can be used without restrictions, that is, the user can copy, distribute, and even modify the software without obtaining permission from the software developer.
2. The term freeware is commonly used for copyrighted software given away free by its author. It is available for free but the author retains the copyright, which means that a user does not have the right to modify anything in the software that is not explicitly allowed by the developer.
3. Firmware is a combination of software, permanently stored in the memory. As the name suggests, firmware is a program or data that has been written onto read only memory (ROM).
4. The name “FOSS” is a recursive acronym for “Free Open Source Software”. Free software is a matter of the users’ freedom to run, copy, distribute, study, change and improve the software.
5. Some of advantages are:
  - Reliability
  - Security
  - Combats Piracy
  - Total Cost of Ownership
  - Non-Quantitative Issues
  - Freedom from control by another

- Protection from licensing litigation
- Flexibility
- Social/Moral/Ethical Issues
- Innovation

NOTES

---

## 1.7. SUMMARY

---

- Firmware is a combination of software, permanently stored in the memory. As the name suggests, firmware is a program or data that has been written onto read only memory (ROM).
- Proprietary describes a technology or product that is owned exclusively by a single company that carefully guards knowledge about the technology or the product's internal working.
- The program must include source code, and must allow distribution in source code as well as compiled form.
- The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

---

## 1.8. KEYWORDS

---

**FOSS:** The name “FOSS” is a recursive acronym for “Free Open Source Software”. Free software is a matter of the users’ freedom to run, copy, distribute, study, change and improve the software.

**Free software:** Free software is software that gives you the user the freedom to share study and modify it.

**Free Redistribution:** The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources.

---

## 1.9. SELF ASSESSMENT QUESTIONS AND EXERCISES

---

**Short Answer questions:**

1. What is Distribution of License?
2. What is Shareware?
3. What is Proprietary Software?
4. What are the needs of open source?
5. What are applications of open source?

**Long Answer questions:**

1. Explain briefly about open sources and its advantages?
2. Explain about Need of open sources?
3. What are Software terminologies?

---

## 1.10. FURTHER READINGS

---

Rémy Card, Eric Dumas, and Franck Mével. The Linux kernel book. John Wiley & Sons, Inc., 2003.

*Introduction*

Steve Suchring. MySQL BBible. John Wiley, 2002.

Rasmus Lerdorf and Levin Tatroe. Programming PHP. " O'Reilly Media, Inc., 2002.

Wesley J. Chun. Core Python Programming. Prentice Hall, 2001.

Martin C. Brown. Perl: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

Steven Holzner. PHP: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

## NOTES

Self-Instructional  
Material

---

# UNIT 2

## OPEN SOURCE OPERATING SYSTEMS

---

### NOTES

#### Structure

- 2.0 Introduction
- 2.1 Objective
- 2.2 Open Source Operating System
- 2.3 General Overview
- 2.4 Kernel Mode
- 2.5 User Mode
- 2.6 Process
- 2.7 Answers to Check Your Progress
- 2.8 Summary
- 2.9 Keywords
- 2.10 Self Assessment Questions and Exercises
- 2.11 Further Readings

---

### 2.0 INTRODUCTION

---

This unit briefs the open source operating system Linux with its general overview of how its available and its usage and working mode are explained,

---

### 2.1 OBJECTIVE

---

This unit helps the users to understand linux by

- Learning the overview of Linux
- User and Kernel Mode
- Process in Linux

---

### 2.2 OPEN SOURCE OPERATING SYSTEM

---

#### Linux

Linus Torvalds [Matt Welsh], a student at the University of Helsinki, created the first version of “Linux” in August 1991. Released as an open-source software under the Free Software Foundation's GNU General Public License (GPL), Linux quickly grew into a complete operating-system package, with contributions from hundreds of programmers. Since the release of version 1.0 in 1994, organizations have been able to download free copies of Linux. One could also purchase commercial distributions of Linux from companies such as Slackware, Red 3 Hat etc who also provide consultancy, services, and maintenance [Carla Schroder]. Many people raise a question about Linux “if it's released under the Free Software



**NOTES**

Foundation's GPL, shouldn't it be free?". The answer is no. A company can charge money for products that include Linux, as long as the source code is made available. The GPL allows people to distribute (and charge for) their own versions of free software [Carla Schroder]. According to the Free Software Foundation, the "free" in free software refers to freedom or liberty, not price. In the foundation's definition, organizations have the freedom to run software for any purpose, study how it works, modify, improve and re-release it.

Another common misconception about Linux is that it's a complete operating system. In reality Linux refers to the "kernel or core" of the operating system. Combining Linux with a set of open-source GNU programs from the Free Software Foundation turns it into what most people know as Linux "forming both the full operating system and the core of most Linux distributions". Distributions are the versions of Linux, GNU programs, and other tools that are offered by different companies, organizations, or individuals. Popular distributions include Red Hat, Debian, SuSE, Caldera, and others. Each distribution might be based on a different version of the Linux kernel, but all migrate forward over time, picking up core changes that are made to the kernel and keeping everything in somewhat loose synchronization.

Eric S. Raymond's famous essay, "The Cathedral and the Bazaar," argues that most commercial software is built like cathedrals by small groups of artisans working in isolation. Open-source software, like Linux, is developed collectively over the Internet, which serves as an electronic bazaar for innovative ideas. The first of the two programming styles is closed source – the traditional factory-production model of proprietary software, in which customers get a sealed block of computer binary that they cannot examine, modify, or evolve. The other style is opensource, the Internet engineering tradition in which software source code is generally available for inspection, independent peer review, and rapid evolution. Linux operating environment is the standard-bearer of the open source approach.

With Open Source products like Linux, new changes come through an open development model, meaning that all new versions are available to the public, regardless of their quality. "Linux' s versioning scheme is designed to let users understand whether they're using a stable version or adevelopment version," says Jim Enright, director of Oracle's Linux program office. "Even decimal-numbered releases [such as 2.0, 2.2, and 2.4] are considered stable versions, while odd numbered releases [such as 2.3 and 2.5] are beta-quality releases intended for developers only." For much of the 1990s, Linux was primarily an experiment: something that developers fiddled with and used on local servers to see how well it worked and how secure it was. Then, with the internet boom of the late 1990s, many companies started using Linux for their Web servers, fueling the first wave of corporate Linux adoption leading to over 30 percent penetration of web server market by 2002.

NOTES

---

## 2.3 GENERAL OVERVIEW

---

Here are some of the benefits and features that Linux provides over single-user operating systems and other versions of UNIX for the PC.

- **Full multitasking and 32-bit support** : Linux, like all other versions of UNIX, is a real multitasking system, allowing multiple users to run many programs on the same system at once. The performance of a 50 MHz 486 system running Linux is comparable to many low- to medium-end workstations, such as those from Sun Microsystems and DEC, running proprietary versions of UNIX. Linux is also a full 32-bit operating system, utilizing the special protected-mode features of the Intel 80386 and 80486 processors.
- **TCP/IP networking support** : TCP/IP (“Transmission Control Protocol/Internet Protocol”) is the set of protocols which links millions of computers into a worldwide network known as the Internet. With an Ethernet connection, one can have access to the Internet or to a local area network from your Linux system. Or, using SLIP (“Serial Line Internet Protocol”), you can access the Internet over the phone lines with a modem.
- **Virtual memory and shared libraries**: Linux can use a portion of the hard drive as virtual memory, expanding the total amount of available RAM. Linux also implements shared libraries, allowing programs which use standard subroutines to find the code for these subroutines in the libraries at runtime. This saves a large amount of space, as each application doesn't store its own copy of these common routines.

### Linux Distributions

Here are some of the more popular distributions of Linux.

- \* Mandrake
- \* Red Hat
- \* SuSE
- \* Caldera
- \* Corel
- \* Debian
- \* Slackware
- \* TurboLinux
- \* Ubuntu

---

## 2.4 KERNEL MODE

---

The system starts in kernel mode when it boots and after the operating system is loaded, it executes applications in user mode. There are some privileged instructions that can only be executed in kernel mode. These are interrupt instructions, input output management etc. If the privileged instructions are executed in user mode, it is illegal and a trap is generated.

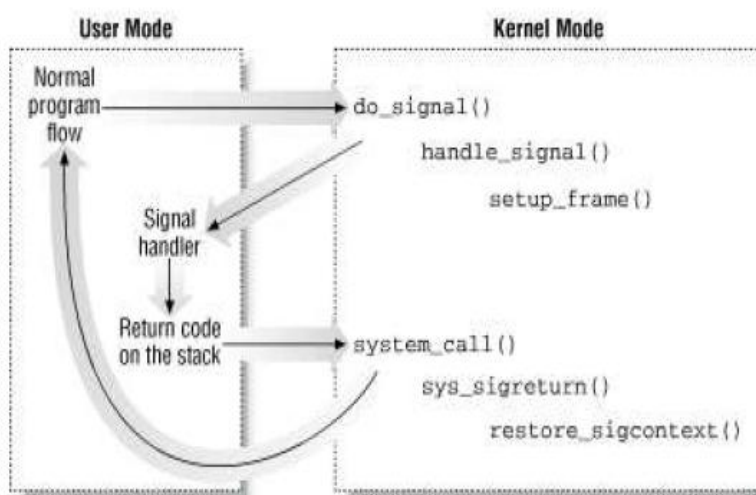
The mode bit is set to 0 in the kernel mode. It is changed from 0 to 1 when switching from kernel mode to user mode

---

## 2.5 USER MODE

---

The system is in user mode when the operating system is running a user application such as handling a text editor. The transition from user mode to kernel mode occurs when the application requests the help of operating system or an interrupt or a system call occurs. The mode bit is set to 1 in the user mode. It is changed from 1 to 0 when switching from user mode to kernel mode.



### Necessity of Dual Mode (User Mode and Kernel Mode) in Operating System

The lack of a dual mode i.e user mode and kernel mode in an operating system can cause serious problems. Some of these are: A running user program can accidentally wipe out the operating system by overwriting it with user data. Multiple processes can write in the same system at the same time, with disastrous results. These problems could have occurred in the MS-DOS operating system which had no mode bit and so no dual mod

---

## 2.6 PROCESS

---

A process is usually defined as an instance of a program in execution; thus, if 16 users are running vi at once, there are 16 separate processes..

### Process Descriptor

To manage processes, the kernel must know the process's priority, whether it is running on the CPU or blocked on some event, what address space has been assigned to it and so on. This is the role of the process descriptor, that is, of a `task_struct` type

### Process State

The following are the possible process states:

## NOTES

## NOTES

**TASK\_RUNNING** The process is either executing on the CPU or waiting to be executed.

**TASK\_INTERRUPTIBLE** The process is suspended (sleeping) until some condition becomes true.

**TASK\_UNINTERRUPTIBLE** like previous state, except that delivering a signal to the sleeping process leaves its state unchanged.

**TASK\_STOPPED** Process execution has been stopped:

The process list To allow an efficient search through processes of a given type the kernel creates several lists of processes. Each list consists of pointers to process descriptors

### 2.6.1 Creating Processes

#### The clone(), fork(), and vfork() System Calls

Lightweight processes are created in Linux by using a function named `__clone()`, which makes use of four parameters:

**fn:** Specifies a function to be executed by the new process; when the function returns, the child terminates.

**Arg:** Pointer to data passed to the `fn()` function.

**Flags:** Miscellaneous information.

**child\_stack** Specifies the User Mode stack pointer to be assigned to the `esp` register of the child

### 2.6.2 Destroying Processes

#### Process Termination

All process terminations are handled by the `do_exit()` function, which removes most references to the terminating process from kernel data structures. The `do_exit()` function executes the following actions:

1. Sets the `PF_EXITING` flag in the flag field of the process descriptor to denote that the process is being eliminated.
2. Examines the process's data structures related to paging, filesystem, open file descriptors, and signal handling, respectively, with the `__exit_mm()`, `__exit_files()`, `__exit_fs()`, and `__exit_sighand()` functions.
3. Sets the state field of the process descriptor to `TASK_ZOMBIE`.
4. Sets the `exit_code` field of the process descriptor to the process termination code.
5. Invokes the `exit_notify()` function to update the parenthood relationships of both the parent process and the children processes.

6. Invokes the `schedule( )` function to select a new process to run. Since a process in a `TASK_ZOMBIE` state is ignored by the scheduler, the process will stop executing right after the `switch_to` macro in `schedule( )` is invoked.

### **2.6.3 Process Removal**

The `release( )` function releases the process descriptor of a zombie process by executing the following steps:

1. Invokes the `free_uid( )` function to decrement by 1 the number of processes created up to now by 3 the user owner of the terminated process.
2. Invokes `add_free_taskslot( )` to free the entry in task that points to the process descriptor to be released.
3. Decrements the value of the `nr_tasks` variable.
4. Invokes `unhash_pid( )` to remove the process descriptor from the `pidhash` hash table.
5. Uses the `REMOVE_LINKS` macro to unlink the process descriptor from the process list.
6. Invokes the `free_task_struct( )` function to release the 8 KB memory area used to contain the process descriptor and the Kernel Mode stack.

#### **Check your Progress**

1. What are the distributions of Linux?
2. What is Kernel Mode?
3. What is User Mode?
4. What are the two different Styles of Programming?
5. State the Necessity of Dual Mode in Operating

---

## **2.7 ANSWERS TO CHECK YOUR PROGRESS**

---

1. Some of the more popular distributions of Linux are:
  - Mandrake
  - Red Hat
  - SuSE
  - Caldera
  - Corel
  - Debian
  - Slackware
  - TurboLinux
  - Ubuntu
2. The system starts in kernel mode when it boots and after the operating system is loaded, it executes applications in user mode. There are some

## **NOTES**

## NOTES

privileged instructions that can only be executed in kernel mode. These are interrupt instructions, input output management etc. If the privileged instructions are executed in user mode, it is illegal and a trap is generated. The mode bit is set to 0 in the kernel mode. It is changed from 0 to 1 when switching from kernel mode to user mode.

3. The system is in user mode when the operating system is running a user application such as handling a text editor. The transition from user mode to kernel mode occurs when the application requests the help of operating system or an interrupt or a system call occurs. The mode bit is set to 1 in the user mode. It is changed from 1 to 0 when switching from user mode to kernel mode.
4. The first of the two programming styles is closed source – the traditional factory-production model of proprietary software, in which customers get a sealed block of computer binary that they cannot examine, modify, or evolve. The other style is opensource, the Internet engineering tradition in which software source code is generally available for inspection, independent peer review, and rapid evolution.
5. The lack of a dual mode i.e. user mode and kernel mode in an operating system can cause serious problems. Some of these are: A running user program can accidentally wipe out the operating system by overwriting it with user data. Multiple processes can write in the same system at the same time, with disastrous results. E.g., These problems could have occurred in the MS-DOS operating system which had no mode bit and so no dual mode.

---

## 2.8 SUMMARY

- Open-source software, like Linux, is developed collectively over the Internet, which serves as an electronic bazaar for innovative ideas.
- Linux, like all other versions of UNIX, is a real multitasking system, allowing multiple users to run many programs on the same system at once.
- The system starts in kernel mode when it boots and after the operating system is loaded, it executes applications in user mode.
- The system is in user mode when the operating system is running a user application such as handling a text editor.

---

## 2.9 KEYWORDS

**GNU software support** : Linux supports a wide range of free software written by the GNU Project, including utilities such as the GNU C and C++ compiler, gawk, groff, and so on.

**X Window System** : The X Window System is the de facto industry standard graphics system for UNIX machines. A free version of The X Window System (known as “Xfree86”) is available for Linux.

**Virtual memory and shared libraries**: Linux also implements shared libraries, allowing programs which use standard subroutines to find the code for these subroutines in the libraries at runtime.

---

## 2.10 SELF ASSESSMENT QUESTIONS AND EXERCISES

---

### **Short Answer questions:**

1. What is Virtual Memory and Shared Libraries? Mention the uses.
2. Write a short note on Process State
3. What is the Role of ProcessDescriptor?
4. What is Creating and Destroying in Process?
5. Write a short note on The X Window System?

### **Long Answer questions:**

1. Explain about Process Creating and steps involved in it.
2. Explain briefly about Process Destroying and Process Removal.
3. Explain and illustrate User Mode and Kernel Mode in Operating system and its necessity.

---

## 2.11 FURTHER READINGS

---

Rémy Card, Eric Dumas, and Franck Mével. The Linux kernel book. John Wiley & Sons, Inc., 2003.

Steve Suchring. MySQL BBible. John Wiley, 2002.

Rasmus Lerdorf and Levin Tatroe. Programming PHP. " O'Reilly Media, Inc., 2002.

Wesley J. Chun. Core Python Programming. Prentice Hall, 2001.

Martin C. Brown. Perl: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

Steven Holzner. PHP: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

Vikram Vaswani. MySQL: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

### **NOTES**

---

## UNIT 3

# ADVANCED CONCEPTS

---

### Structure

- 3.0 Introduction
- 3.1 Objective
- 3.2 Scheduling Concepts
- 3.3 Signals
- 3.4 Process
- 3.5 Cloning
- 3.6 Personalities
- 3.7 Development with Linux
- 3. 8 Answers to Check Your Progress
- 3.9 Summary
- 3.10 Keywords
- 3.11 Self Assessment Questions and Exercises
- 3.12 Further Readings

---

### 3.0 INTRODUCTION

---

This unit explains the advanced concepts of open source software with linux where the scheduling and signals are discussed briefly. The development of the open source software Linux is also discussed

---

### 3.1 OBJECTIVE

---

This unit helps the users to understand the advance concepts such as

- Scheduling
- Cloning
- Personalities
- Signals

---

### 3.2 SCHEDULING CONCEPTS

---

#### 3.2.1 Scheduling Policy

The scheduling algorithm of traditional UNIX operating systems must fulfil several conflicting objectives: fast process response time, good throughput for background jobs, avoidance of process starvation, reconciliation of the needs of low- and high-priority processes, and so on. The set of rules used to determine when and how selecting a new process to run is called scheduling policy. Linux scheduling is based on the time-sharing technique several processes are allowed to run "concurrently,"



### 3.2.2 Process Pre-emption

Linux processes are pre-emptive. If a process enters the TASK\_RUNNING state, the kernel checks whether its dynamic priority is greater than the priority of the currently running the execution of current is interrupted and the scheduler is invoked to select another process to run. Be aware that a pre-empted process is not suspended, since it remains in the TASK\_RUNNING state; it simply no longer uses the CPU. The Linux kernel is not pre-emptive, which means that a process can be pre-empted only while running in User Mode

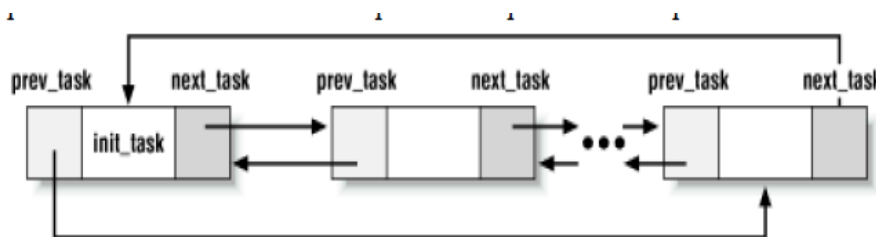
## NOTES

### 3.2.3 The Scheduling Algorithm

The Linux scheduling algorithm works by dividing the CPU time into epochs. In a single epoch, every process has a specified time quantum whose duration is computed when the epoch begins. The time quantum value is the maximum CPU time portion assigned to the process in that epoch. When a process has exhausted its time quantum, it is preempted and replaced by another

**runnable process.** The epoch ends when all runnable processes have exhausted their quantum.

**The schedule( ) Function-** schedule( ) implements the scheduler. Its objective is to find a process in the run queue list and then assign the CPU to it.



#### Direct invocation

The scheduler is invoked directly when the current process must be blocked right away because the resource it needs is not available. In this case, the kernel routine that wants to block it proceeds as follows:

1. Inserts current in the proper wait queue
2. Changes the state of current either to TASK\_INTERRUPTIBLE or to PROCESS TASK\_UNINTERRUPTIBLE 4
3. Invokes schedule( )
4. Checks if the resource is available; if not, goes to step 2
5. Once the resource is available, removes current from the wait queue

#### Lazy invocation

NOTES

The scheduler can also be invoked in a lazy way by setting the need\_resched field of current to 1. Actions performed by schedule( )

1. The schedule( ) function starts by running the functions left by other kernel control paths in various queues. The function invokes run\_task\_queue(&tq\_scheduler); The function then executes all active unmasked bottom halves.

2. Now comes the actual scheduling, and therefore a potential process switch. The value of current is saved in the prev local variable. First, a check is made to determine whether prev is a Round Robin real-time process that has exhausted its quantum. If so, schedule( ) assigns a new quantum to prev and puts it at the bottom of the runqueue list:

3. schedule( ) examines the state of prev. If it has nonblocked pending signals and its state is TASK\_INTERRUPTIBLE, the function wakes up the process as Task\_running.

4. If prev is not in the TASK\_RUNNING state, prev must be removed from the runqueue list:

5. Next, schedule( ) must select the process to be executed in the next time quantum. To that the function scans the runqueue list. The objective is to store in next the process descriptor pointer of the highest priority process. schedule( ) repeatedly invokes the goodness( ) function on the runnable processes to determine the best candidate:

6. A further check must be made at the exit of the loop to determine whether c is 0. This occurs only when all the processes in the runqueue list have exhausted their quantum

7. If a process other than prev has been selected, a process switch must take place. Before performing it, however, the context\_swch field of kstat is increased by 1 to update the statistics maintained by the kernel.

---

### 3.3 SIGNALS

---

A signal is a very short message that may be sent to a process or to a group of processes. The only information given to the process is usually the number identifying the signal. A set of macros whose names start with the prefix SIG is used to identify signals; For example, the SIGCHLD macro yields the identifier of the signal that is sent to a parent process when some child stops or terminates.

Signals serve two main purposes:

- To make a process aware that a specific event has occurred
- To force a process to execute a signal handler function included in its code

The kernel distinguishes two different phases related to signal transmission:

Signal sending The kernel updates the descriptor of the destination process to represent that a new signal has been sent. Signal receiving The kernel

forces the destination process to react to the signal by changing its execution state or by starting the execution of a specified signal handler or both. The following factors must be taken into consideration:

- Signals are usually received only by the currently running process
- Signals of a given type may be selectively blocked by a process; in this case, the process will not receive the signal until it removes the block.

### 3.3.1 Process

Actions Performed upon Receiving a Signal 5 There are three ways in which a process can respond to a signal:

- Explicitly ignore the signal.
- Execute the default action associated with the signal

#### Abort

The process is destroyed. Dump The process is destroyed and a core file containing its execution context is created.

#### Ignore

The signal is ignored. Stop- The process is stopped, that is, put in a TASK\_STOPPED state

Continue If the process is stopped (TASK\_STOPPED), it is put into the TASK\_RUNNING

state. Catch the signal by invoking a corresponding signal-handler function.

### 3.3.2 Sending a Signal

When a signal is sent to a process, the kernel delivers it by invoking the `send_sig_info()`, `send_sig()`,

`force_sig()`, or `force_sig_info()` functions.

The `send_sig_info()` and `send_sig()` Functions

The `send_sig_info()` function acts on three parameters:

Sig The signal number. Info Either the address of a `siginfo_t` table associated with real-time signals or one of two special values. A pointer to the descriptor of the destination process.

1. The `send_sig_info()` function starts by checking whether the parameters are correct:

```
if (sig < 0 || sig > 64) return -EINVAL;
```

2. The function checks then if the signal is being sent by a User Mode process.

## NOTES

3. If the signal is sent by a User Mode process, the function determines whether the operation is allowed. If the sig parameter has the value 0, the function returns immediately without sending any signal: since 0 is not a valid signal number. Some types of signals might nullify other pending signals for the destination process.

4. Next, send\_sig\_info( ) checks whether the new signal can be handled immediately.

### 3.3.3 Receiving Signals

The kernel checks whether there are nonblocked pending signals before allowing a process to resume its execution in User Mode. This check is performed in ret\_from\_intr( ) every time an interrupt or an exception has been handled by the kernel routines. In order to handle the nonblocked pending signals, the kernel invokes the do\_signal( ) function, which receives two parameters: Regs The address of the stack area

---

## 3.4 PROCESS

---

Oldest The address of a variable where the function is supposed to save the bit mask array of 6 blocked signals. The heart of the do\_signal( ) function consists of a loop that repeatedly invokes dequeue\_signal( ) until no more nonblocked pending signals are left. The return code of dequeue\_signal( ) is stored in the signr local variable: if its value is 0, it means that all pending signals have been handled and do\_signal( ) can finish

### Ignoring the Signal

When a received signal is explicitly ignored, the do\_signal( ) function normally just continues with a new execution of the loop and therefore considers another pending signal. If the signal received is SIGCHLD, the sys\_wait4( ) service routine of the wait4( ) system call is invoked to force the process to read information about its children.

### Executing the Default Action for the Signal

If ka->sa.sa\_handler is equal to SIG\_DFL, do\_signal( ) must perform the default action of the signal. The only exception comes when the receiving process is init, in which case the signal is discarded. The signals whose default action is "stop" may stop the current process. In order to do this, do\_signal( ) sets the state of current to TASK\_STOPPED and then invokes the schedule( ) Function.

---

## 3.5 CLONING

---

A clone process is created, using primitive type clone, by duplicating its parent process. But, unlike classical processes, it may share its context with its parent.

The standard form of the clone function is as follows:

```
int clone(int (*fn)(), void *child stack, int flag, int nargs,...);
```

- The parameter `fn` is the pointer from the child process to the function to be executed.
- The parameter `child stack` is the pointer to the zone of memory allocated for the stack of child process. The parameter `Flags` defines method for cloning.
- The parameter `nargs` defines the number of arguments to be passed to the function pointed by `fn` and is followed by these arguments

However, it is not necessary to have the source code in order to make a clone of a program. In fact, having access to the source code can be undesirable because it could subconsciously influence a programmer and result in imitation of part of the code rather than creating entirely original code. A variety of techniques have been developed for cloning software, including reimplementing from official documentation accompanying the software and from unofficial documentation. Reverse engineering becomes particularly important when the documentation is incomplete, which is often the case. Among the ways in which reverse engineering can be accomplished are through observation of information exchange, disassembly using a disassembler and decompilation using a decompiler. A compiler converts source code into machine language, which can be read directly by a computer's CPU (central processing unit); a decompiler attempts to convert machine language back into source code. An assembler is a program that translates an assembly language (which is a very low level language close to machine language) into machine language.

Clones are generally cheaper than the original software, and in many cases they are free software (i.e., software that is free both in a monetary sense and with regard to use). However, they are not necessarily inferior. In fact, in some cases they are as good as or superior to the originals, as is clearly illustrated by Linux. They also frequently have the benefit of providing competition for the original software, thereby stimulating its developers to improve its performance and features and to lower its price. In contrast to proprietary software, there is no reason to make clones of free software. This is because, by definition, the source code for free software is freely available to anyone to use for any purpose, including modification and redistribution of such modified versions. Occasionally, there is sufficient dissatisfaction with the way in which a free software program is being developed to result in a project fork, which is the starting of a new development branch independent of the the existing project but based on the same source code.

A second meaning of the word clone in a software context is to make an exact copy of a file, directory or disk inclusive of any files and subdirectories within that directory or disk. The term is also used in a hardware context. Most notably, when IBM introduced its revolutionary IBM PC in 1981, other companies decided to develop clones as a legal reimplementing from the PC's documentation and through reverse engineering. Because most of the components were publicly available with the exception of the BIOS (basic input output system), the only major task was reverse engineering the BIOS.

## NOTES

---

### 3.6 PERSONALITIES

---

In order to allow programs coming from other operating system to be run in Linux, Linux supports the idea of personalities. Each process is assigned to an execution domain. This domain specifies the way in which system calls are carried out, and the way in which messages are processed.

**System calls:** Linux uses software interrupts to change into kernel mode, whilst other UNIX system use an inter- segment jump.

**Message number specified by processes:** When a process specifies a message number, for example in calling the primitive sanction or kill, the message number is converted by means of look-up-table

**Number of messages sent to processes:** when a message is to be sent to a process, the message number is converted by means of look-up-table

The system call personality allows a process to modify its execution domain in order that Linux can emulate the behaviour of another operating system. `int personality(int pers);`

---

### 3.7 DEVELOPMENT WITH LINUX

---

It's Free Linux is free. Really and truly free. One can browse to any of the distributors of Linux, find the "download" link and download a complete copy of the entire operating system plus extra software without paying anything. One can also buy a boxed version of course. For a nominal price, the CDs are available, manuals get door-delivered, plus there is telephone or online support. By comparison, "home" versions of popular commercial OS would cost thousands of rupees.

With Linux you also don't have to worry about paying again every time you upgrade the operating system - the upgrades are obviously free too. With commercial OS, upgrades also have to be paid for every time one is announced.

#### **It's Open Source**

This means two things: First, that the CDs (or the download site) contain an entire copy of the source code for Linux. Secondly, the user can legally make modifications to improve it. While this might not mean much to non-programmers, there are thousands of people with programming capability who could improve the code or fix problems quickly. When a problem is found, it is sent off to the coordinating team in charge of the module in question, who will update the software and issue a patch. What all this boils down to is that bugs in Linux get fixed much faster than any other operating system.

#### **It's Modular**

Commercial Operating Systems normally get installed as a complete unit. One cannot, for example, install them without their Graphical User Interface, or without its printing support - install everything or nothing. Linux, on the other hand, is a very modular operating system. One could

install or run exactly the bits and pieces of Linux that are needed. In most cases, the choice is on one of the predefined setups from the installation menu, but is not compulsory. In some cases this makes a lot of sense. For example, while setting up a server, one might want to disable the graphical user interface once it is set up correctly, thus freeing up memory and the processor for the more important task at hand.

It also allows the users to upgrade parts of the operating system without affecting the rest. For example, one could get the latest version of Gnome or KDE without changing the kernel.

### **It's got More Choices**

Also due to its modularity, there is more choice of components to use. One example is the user interface. Many users choose KDE, which is very easy to learn for users with Windows experience. Others choose Gnome, which is more powerful but less similar to Windows. There are also several simple alternatives for less-powerful computers, which make less demand on the hardware available.

### **It's Portable**

Linux runs on practically every piece of equipment which qualifies as a computer. It can be run on huge multiprocessor servers or a PDA. Apart from Pentiums of various flavors, there are versions of Linux (called "ports") on Atari, Amiga, Macintosh, PowerMac, PowerPC, NeXT, Alpha, Motorola, MIPS, HP, PowerPC, Sun Sparc, Silicon Graphics, VAX/MicroVax, VME, Psion 5, Sun UltraSparc, etc.

### **It's got lots of Extras**

Along with the Linux CD, normally quite a lot of software gets thrown in, which is not usually included with operating systems. Using only the applications that come with Linux, one could set up a full web, ftp, database and email server for example. There is a firewall built into the kernel of the operating system, one or more office suites, graphics programs, music players, and lots more. Different distributions of Linux offer different "extra programs". Slackware, for example, is quite simple (though it still provides all the commonly needed programs), while SuSE Linux comes with seven CDs and a DVD-ROM!

### **It is Stable**

All applications can crash, but in many systems, the only recourse is to switch off and reboot (and with some new "soft-switch" PCs, even that doesn't work - you have to pull out the power cable). In comparison, Linux is rock-solid. Every application runs independently of all others - if one crashes, it crashes alone. Most Linux servers run for months on end, never shutting down or rebooting. Even the GUI is independent of the kernel of the operating system.

## Linux Configuration Tool

LinuxConf is a popular utility which allows the configuration of most parts of Linux and its applications from one place

### NOTES

#### Check your Progress

1. What is called scheduling Policy?
2. State the Process of Pre-emption.
3. How the Scheduling Algorithm works?
4. What is Signal? Mention the Properties?
5. Write Short note on Personalities? And where it is used in Linux?

---

### 3.8 ANSWERS TO CHECK YOUR PROGRESS

---

1. The scheduling algorithm of traditional UNIX operating systems must fulfil several conflicting objectives: fast process response time, good throughput for background jobs, avoidance of process starvation, reconciliation of the needs of low- and high-priority processes, and so on. The set of rules used to determine when and how selecting a new process to run is called scheduling policy.
2. Linux processes are pre-emptive. If a process enters the `TASK_RUNNING` state, the kernel checks whether its dynamic priority is greater than the priority of the currently running the execution of current is interrupted and the scheduler is invoked to select another process to run. Be aware that a pre-empted process is not suspended, since it remains in the `TASK_RUNNING` state; it simply no longer uses the CPU. The Linux kernel is not pre-emptive, which means that a process can be pre-empted only while running in User Mode
3. The Linux scheduling algorithm works by dividing the CPU time into epochs. In a single epoch, every process has a specified time quantum whose duration is computed when the epoch begins. The time quantum value is the maximum CPU time portion assigned to the process in that epoch. When a process has exhausted its time quantum, it is pre-empted and replaced by another
4. A signal is a very short message that may be sent to a process or to a group of processes.
  - To make a process aware that a specific event has occurred
  - To force a process to execute a signal handler function included in its code
5. In order to allow programs coming from other operating system to be run in Linux, Linux supports the idea of personalities. Each process is assigned to an execution domain. This domain specifies the way in which system calls are carried out, and the way in which messages are processed.
  - System calls



- Message number specified by processes
- Number of messages sent to processes

---

### 3.9 SUMMARY

---

- The Linux scheduling algorithm works by dividing the CPU time into epochs. In a single epoch, every process has a specified time quantum whose duration is computed when the epoch begins.
- The kernel checks whether there are nonblocked pending signals before allowing a process to resume its execution in User Mode.
- A clone process is created, using primitive type clone, by duplicating its parent process. But, unlike classical processes, it may share its context with its parent.
- Copyright is the power granted by a government to the creator of a creative work (e.g., the source code for a program, a magazine article, a poem, a painting or a musical composition) that gives that creator the exclusive, although transferable, right to copy or perform that work for a defined period of time.

---

### 3.10 KEYWORDS

---

**Process Pre-emption:** Linux processes are pre-emptive. If a process enters the TASK\_RUNNING state, the kernel checks whether its dynamic priority is greater than the priority of the currently running the execution of current is interrupted and the scheduler is invoked to select another process to run.

**runnable process.** The epoch ends when all runnable processes have exhausted their quantum.

**Signal:** A signal is a very short message that may be sent to a process or to a group of processes.

**System calls:** Linux uses software interrupts to change into kernel mode, whilst other UNIX system use an inter- segment jump.

**Message number specified by processes:** When a process specifies a message number, for example in calling the primitive sanction or kill, the message number is converted by means of look-up-table.

---

### 3.11 SELF ASSESSMENT QUESTIONS AND EXERCISES

---

**Short Answer questions:**

1. What is Virtual Memory and Shared Libraries? Mention the uses.
2. What is wine? And why we using it?
3. What are the three ways a process responds to a signal?
4. What is the Reason for the existence of Cloning software?

**Long Answer questions:**

1. Explain about Sending and Receiving Signal.
2. Explain briefly about Scheduling Algorithm
3. Explain about Cloning.

---

**3.12. FURTHER READINGS**

---

Rémy Card, Eric Dumas, and Franck Mével. The Linux kernel book. John Wiley & Sons, Inc., 2003.

Steve Suchring. MySQL BBible. John Wiley, 2002.

Rasmus Lerdorf and Levin Tatroe. Programming PHP. " O'Reilly Media, Inc., 2002.

Wesley J. Chun. Core Python Programming. Prentice Hall, 2001.

Martin C. Brown. Perl: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

---

## BLOCK – II

### OPEN SOURCE DATABASE

---

## UNIT 4

### MySQL

---

**NOTES****Structure**

- 4.0 Introduction
- 4.1 Introduction
- 4.2 MySQL
- 4.3 Setting up Account
- 4.4 Starting, terminating and writing your Own SQL programs
- 4.5 Answers to Check Your Progress
- 4.6 Summary
- 4.7 Keywords
- 4.8 Self Assessment Questions and Exercises
- 4.9 Further Readings

---

**4.0 INTRODUCTION**

---

This unit explains the basic programming of MySQL by establishing and connecting with the server with the root password and how to write own sql programs.

---

**4.1 OBJECTIVE**

---

This unit helps the users to

- Understand and install mysql
- Write own programs

---

**4.2 MySQL**

---

MySQL is the most popular open source SQL database management system (DBMS). A fast, reliable, easy-to-use, multi-user multi-threaded relational database system. It is freely available and released under GPL (GNU General Public License). MySQL is a data storage area. In this storage area, there are small sections called Tables.

**Advantages**

- MySQL is Cross-Platform.
- MySQL is fast.
- MySQL is free.
- Reliable and easy to use.
- Multi-Threaded multi-user and robust SQL Database server.

**Disadvantages**

- Missing Sub-selects.
- MySQL doesn't yet support the Oracle SQL extension.
- Does not support Stored Procedures and Triggers.
- MySQL doesn't support views, but this is on the TODO.

## NOTES

### Following are the tools to manage MySQL server:

- mysqld - MySQL server daemon. It is used to start the mysql server.
- mysqladmin – Used to perform administrative tasks.
- mysql - A command-line interface for end users to manage user data objects.
- mysqlcheck - A command-line interface for administrators to check and repair tables.
- mysqlshow - A command-line interface for end users to see information on tables and columns.
- mysqldump - A command-line interface for administrators or end users to export data from the server to files.
- mysqlimport - A command-line interface for administrators or end users to load data files into tables program tool to load data into tables.

---

## 4.3 SETTING UP ACCOUNT

---

In order to provide access the MySQL database you need to create an account to use for connecting to the MySQL server running on a given host. Use the GRANT statement to set up the MySQL user account. Then use that account's name and password to make connections to the server.

User names, as used by MySQL for authentication purposes, have nothing to do with user names (login names) as used by Windows or UnixMySQL user names can be up to 16 characters long.

MySQL passwords have nothing to do with passwords for logging in to your operating system.

### Account Management Statements

- CREATE USER
- DROUP USER
- RENAME USER
- REVOKE
- SET PASSWORD

### CREATE USER

**Syntax:**

```
CREATE USER user [IDENTIFIED BY [PASSWORD] 'password'] [, user  
[IDENTIFIED BY [PASSWORD] 'password']] ...
```

**Example:**

```
CREATE USER 'monty'@'localhost' IDENTIFIED BY 'some_pass';
```

**DROP USER****Syntax:**

```
DROP USER user [, user] ...
```

**Example:**

```
DROP USER 'jeffrey'@'localhost';
```

**RENAME USER****Syntax**

```
RENAME USER old_user TO new_user [, old_user TO new_user] ...
```

**Example**

```
RENAME USER 'jeffrey'@'localhost' TO 'jeff'@'127.0.0.1';
```

**SET PASSWORD****Syntax**

```
SET PASSWORD [FOR user] =  
{  
    PASSWORD('some password')  
| OLD_PASSWORD('some password') | 'encrypted password'  
}
```

**Example:**

```
SET PASSWORD FOR 'bob'@'%.loc.gov' =  
PASSWORD('newpass');
```

**NOTES**

## 4.4 STARTING, TERMINATING AND WRITING YOUR OWN SQL PROGRAMS

### Invoking MySQL Programs

#### NOTES

To invoke a MySQL program from the command line (that is, from your shell or command prompt), enter the program name followed by any options or other arguments needed to instruct the program what you want it to do. The following commands show some sample program invocations. “shell>” represents the prompt for your command interpreter; it is not part of what you type. The particular prompt you see depends on your command interpreter. Typical prompts are \$ for sh or bash, % for csh or tcsh, and C:\> for the Windows command.com or cmd.exe command interpreters.

```
shell> mysql --user=root test
```

```
shell> mysqladmin extended-status variables
```

```
shell> mysqlshow --help
```

```
shell> mysqldump -u root personnel
```

Arguments that begin with a single or double dash (“-”, “--”) specify program options. Options typically indicate the type of connection a program should make to the server or affect its operational mode. Non-option arguments (arguments with no leading dash) provide additional information to the program. For example, the mysql program interprets the first non-option argument as a database name, so the command `mysql --user=root test` indicates that you want to use the test database.

#### 4.4.1 Connecting to the MySQL Server

For a client program to be able to connect to the MySQL server, it must use the proper connection parameters, such as the name of the host where the server is running and the username and password of your MySQL account. Each connection parameter has a default value, but you can override them as necessary using program options specified either on the command line or in an option file. The examples here use the mysql client program, but the principles apply to other clients such as mysqldump, mysqladmin, or mysqlshow.

This command invokes mysql without specifying any connection parameters explicitly:

```
shell> mysql
```

Because there are no parameter options, the default values apply: The default hostname is localhost. On Unix, this has a special meaning, as described later. The default username is ODBC on Windows or your Unix login name on Unix. No password is sent if neither -p nor --password is given. To specify the hostname and username explicitly, as well as a password, supply appropriate options on the command line:

```
shell> mysql --host=localhost --user=myname --password=mypass
```

```
shell> mysql -h localhost -u myname -pmypass
```

For password options, the password value is optional: If you use a `-p` or `--password` option but do not specify the password value, the client program prompts you to enter the password. The password is not displayed as you enter it. This is more secure than giving the password on the command line. Any user on your system may be able to see a password specified on the command line by executing a command such as `ps auxw`. See Section 5.5.6, “Keeping Your Password Secure”. If you use a `-p` or `--password` option and do specify the password value, there must be no space between `-p` or `--password=` and the password following it.

On Unix, MySQL programs treat the hostname `localhost` specially, in a way that is likely different from what you expect compared to other network-based programs. For connections to `localhost`, MySQL programs attempt to connect to the local server by using a Unix socket file. This occurs even if a `--port` or `-P` option is given to specify a port number. To ensure that the client makes a TCP/IP connection to the local server, use `--host` or `-h` to specify a hostname value of `127.0.0.1`, or the IP address or name of the local server. You can also specify the connection protocol explicitly, even for `localhost`, by using the `--protocol=TCP` option. For example:

```
shell> mysql --host=127.0.0.1
```

```
shell> mysql --protocol=TCP
```

The `--protocol` option enables you to establish a particular type of connection even when the other options would normally default to some other protocol. On Windows, you can force a MySQL client to use a named-pipe connection by specifying the `--pipe` or `--protocol=PIPE` option, or by specifying `.` (period) as the host name. If named-pipe connections are not enabled, an error occurs. Use the `--socket` option to specify the name of the pipe if you do not want to use the default pipe name.

Connections to remote servers always use TCP/IP. This command connects to the server running on `remote.example.com` using the default port number (3306):

```
shell> mysql --host=remote.example.com
```

To specify a port number explicitly, use the `--port` or `-P` option:

```
shell> mysql --host=remote.example.com --port=13306
```

You can specify a port number for connections to a local server, too. However, as indicated previously, connections to `localhost` on Unix will use a socket file by default. You will need to force a TCP/IP connection as already described or any option that specifies a port number will be ignored. For this command, the program uses a socket file on Unix and the `--port` option is ignored:

```
shell> mysql --port=13306 --host=localhost
```

To cause the port number to be used, invoke the program in either of these ways:

```
shell> mysql --port=13306 --host=127.0.0.1
```

```
shell> mysql --port=13306 --protocol=TCP
```

The following options may be used to control how client programs connect to the server:

--host=host\_name, -h host\_name

The host where the server is running. The default value is localhost.

--password[=pass\_val], -p[pass\_val]

The password of the MySQL account. As described earlier, the password value is optional, but if given, there must be no space between -p or --password= and the password following it. The default is to send no password.

--pipe, -W

On Windows, connect to the server via a named pipe. This option applies for connections to a local server only. The server must have been started with the --enable-named-pipe option to enable named-pipe connections.

--port=port\_num, -P port\_num

The port number to use for the connection, for connections made via TCP/IP. The default port number is 3306.

--protocol={TCP|SOCKET|PIPE|MEMORY}

This option explicitly specifies a protocol to use for connecting to the server. It is useful when the other connection parameters normally would cause a protocol to be used other than the one you want. For example, connections on Unix to localhost are made via a Unix socket file by default:

```
shell> mysql --host=localhost
```

To force a TCP/IP connection to be used instead, specify a --protocol option:

```
shell> mysql --host=localhost --protocol=TCP
```

To start the mysql program, try just typing its name at your command-line prompt. If mysql starts up correctly, you'll see a short message, followed by a mysql> prompt that indicates the program is ready to accept queries. To illustrate, here's what the welcome message looks like (to save space, I won't show it in any further examples):

## Starting and Terminating



```
% mysql
```

Welcome to the MySQL monitor. Commands end with ; or \g.

Your MySQL connection id is 18427 to server version: 3.23.51-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

```
mysql>
```

If mysql tries to start but exits immediately with an “access denied” message, you’ll need to specify connection parameters. The most commonly needed parameters are the host to connect to (the host where the MySQL server runs), your MySQL username, and a password. For example:

```
% mysql -h localhost -p -u cbuser
```

Enter password: cbpass

In general, I’ll show mysql commands in examples with no connection parameter options. I assume that you’ll supply any parameters that you need, either on the command line, or in an option file (Recipe 1.5) so that you don’t have to type them each time you invoke mysql. If you don’t have a MySQL username and password, The syntax and default values for the connection parameter options are shown in the following table. These options have both a single-dash short form and a double-dash long form.

```
% mysql -pEnter password: ← enter your password here
```

If you like, you can specify the password directly on the command line by using either -ppassword (note that there is no space after the -p) or --password=password. I don’t recommend doing this on a multiple-user machine, because the password may be visible momentarily to other users who are running tools such as ps that report process information. If you get an error message that mysql cannot be found or is an invalid command when you try to invoke it, To terminate a mysql session, issue a QUIT statement:

```
mysql> QUIT
```

You can also terminate the session by issuing an EXIT statement or (under Unix) by typing Ctrl-D. The way you specify connection parameters for mysql also applies to other MySQL programs such as mysqldump and mysqladmin.

### Check your Progress

1. Write short notes on MySQL.
2. Name the tools used to manage MySQL server.
3. Mention the Account Management Statements.
4. How to invoke a MySQL from command line?
5. What are the Advantages of MySQL?

## 4.5 ANSWERS TO CHECK YOUR PROGRESS

1. MySQL is the most popular open source SQL database management system (DBMS). A fast, reliable, easy-to-use, multi-user multi-threaded relational database system.
2. Some of the tools used are
  - Mysqld
  - Mysqladmin
  - Mysql
  - mysqlcheck
  - mysqlshow
  - mysqldump
  - mysqlimport
3. Some of the account management statements are
  - CREATE USER
  - DROUP USER
  - RENAME USER
  - REVOKE
  - SET PASSWORD
4. To invoke a MySQL program from the command line (that is, from your shell or command prompt), enter the program name followed by any options or other arguments needed to instruct the program what you want it to do. The following commands show some sample program invocations. “shell>” represents the prompt for your command interpreter
5. The advantages of MySQL are
  - MySQL is Cross-Platform.
  - MySQL is fast.
  - MySQL is free.
  - Reliable and easy to use.
  - Multi-Threaded multi-user and robust SQL Database server.

---

## 4.6 SUMMARY

- In order to provide access the MySQL database you need to create an account to use for connecting to the MySQL server running on a given host.
- To invoke a MySQL program from the command line (that is, from your shell or command prompt), enter the program name followed by any options or other arguments needed to instruct the program what you want it to do.
- You can also terminate the session by issuing an EXIT statement or (under Unix) by typing Ctrl-D.
- To start the mysql program, try just typing its name at your command-line prompt. If mysql starts up correctly, you’ll see a short message, followed by a mysql> prompt that indicates the program is ready to accept queries.

---

## 4.7 KEYWORDS

---

**MySQL:** It is the most popular open source SQL database management system (DBMS).

**User names:** It is used by MySQL for authentication purposes, have nothing to do with user names (login names) as used by Windows or UnixMySQL user names can be up to 16 characters long.

---

## 4.8 SELF ASSESSMENT QUESTIONS AND EXERCISES

---

### Short Answer questions:

1. What are the disadvantages of MySQL?
2. Write a syntax with example to Create User and Drop User.
3. Write a syntax with example to Rename User and Revoke.
4. Write a syntax with example to Set a new Password and change the Old Password.

### Long Answer questions:

1. Explain the Process involved in Connecting to the MySQL Server.

---

## 4.9 FURTHER READINGS

---

Rémy Card, Eric Dumas, and Franck Mével. The Linux kernel book. John Wiley & Sons, Inc., 2003.

Steve Suchring. MySQL BBible. John Wiley, 2002.

Rasmus Lerdorf and Levin Tatroe. Programming PHP. " O'Reilly Media, Inc., 2002.

Wesley J. Chun. Core Python Programming. Prentice Hall, 2001.

Martin C. Brown. Perl: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

Steven Holzner. PHP: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

Vikram Vaswani. MySQL: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

---

# UNIT 5

## RECORD SELECTION TECHNOLOGY

---

### Structure

- 5.0 Introduction
- 5.1 Objective
- 5.2 Record selection Technology
- 5.3 Working with Strings
- 5.4 Date and Time Functions
- 5.5 Answers to Check Your Progress
- 5.6 Summary
- 5.7 Keywords
- 5.8 Self Assessment Questions and Exercises
- 5.9 Further Readings

---

### 5.0 INTRODUCTION

---

This unit covers the Mysql commands for selection of records from the tables and how strings are manipulated in Mysql and the date and time functions used in Mysql are described with their syntax and examples

---

### 5.1 Objective

---

This unit helps the users to learn and understand the following

- Record selection
- Working with strings
- Date and time functions

---

### 5.2 Record Selection Technology

---

The SELECT statement is used to select data from a database. The statement begins with the SELECT keyword. The basic SELECT statement has 3 clauses:

SELECT

FROM

WHERE

Select command is used to collect data from the table. We will discuss more on this by adding more commands to this select command. If you have not created the tables then please go to sql introduction page to use the sql commands to create the tables and fill the table with some records ( data ). Now your table with some data is ready. We will apply select command to our table ( name student ) and fetch all the records.

SELECT \* FROM `student`

id	name	mark	class
1	John Deo	Four	75
2	Max Ruin	Three	85
3	Arnold	Three	55
4	Krish Star	Four	60
5	John Mike	Four	60
6	Alex John	Four	55
7	My John Rob	Fifth	78
8	Asruid	Five	5
9	Tes Qry	Six	78
10	Big John	Four	55

**NOTES**

That's all to get all the records from the table student. We have not placed any restriction here and asked for all the fields with all the records. Now if we want to restrict our results and get only name field from the table.

SELECT name FROM `student`

This will return only name field from the table. We can ask for more fields and all field names we have to separate by comma. We can include as many field names we want from a table. SELECT name, class FROM `student`

name	class
John Deo	Four
Max Ruin	Three
Arnold	Three
Krish Star	Four
John Mike	Four
Alex John	Four
My John Rob	Fifth
Asruid	Five
Tes Qry	Six
Big John	Four

Now we will go for bit more and restrict the number of records we are getting. We are interested say in only 3 records. Our command should return 3 records only. We will use SQL limit command. This will take two parameters. One is the starting point and other is number of records

required. Say we are interested in 3 records starting from beginning. Our command will be

## NOTES

```
SELECT * FROM `student` LIMIT 0,3
```

Related Tutorial

Copy data to new table

SQL Left Join

SQL UNION

This command with LIMIT command will return us 3 records starting from 0 ( or first ) record. This is very important when we use this command with ORDER BY command. Now let us try to list all the students who have come in first 3 ranks. We are required to list here 3 records who have mark more than the others. The top 3 students we want to display in order of first, second and third. The order we can display are in by default in ascending order but we require the listing should return in descending order so we will get the highest ranked student at the top. Before that let us start with a simple example of ORDER BY command.

```
SELECT * FROM `student` ORDER BY mark
```

This will display the records of students in the order of lowest mark to highest mark.

id	name	class	mark
19	Tinny	Nine	18
17	Tumyu	Six	54
10	Big John	Four	55
22	Reggid	Seven	55
29	Tess Played	Seven	55
6	Alex John	Four	55
3	Arnold	Three	55
5	John Mike	Four	60
4	Krish Star	Four	60
20	Jackly	Nine	65

We will change it to display in reverse order so it will display highest mark at the top and lowest mark at the bottom.

```
SELECT * FROM `student` ORDER BY `mark` DESC
```

With the addition of command DESC we can change the order of display to keep the highest mark at the top of the list and lowest mark at the bottom of the list. Now let us add the LIMIT command to display only the top 3 records. We already have the list in the order of highest mark to lowest mark so by just limiting the number of records to 3 will give our required top three student records.

```
SELECT * FROM `student` ORDER BY `mark` DESC LIMIT 0,3
```

id	name	class	mark
33	Kenn Rein	Six	96
12	Recky	Six	94
32	Binn Rott	Seven	90

This is the SQL query which will display top three students based on the mark they scored. In the next section we will use sql WHERE clause to restrict or filter the records.

### 5.3 WORKING WITH STRINGS

String functions are used to perform an operation on input string and return an output string. Following are the string functions defined in SQL:

**1. ASCII():** This function is used to find the ASCII value of a character.

**Syntax:** SELECT ascii('t');

**Output:** 116

**2. CHAR\_LENGTH():** This function is used to find the length of a word.

**Syntax:** SELECT char\_length('Hello!');

**Output:** 6

**3. CHARACTER\_LENGTH():** This function is used to find the length of a line.

**Syntax:** SELECT CHARACTER\_LENGTH('geeks for geeks');

**Output:** 15

**4. CONCAT():** This function is used to add two words or strings.

**Syntax:** SELECT 'Geeks' || ' ' || 'forGeeks' FROM dual;

**Output:** 'GeeksforGeeks'

**5. CONCAT\_WS():** This function is used to add two words or strings with a symbol as concatenating symbol.

**Syntax:** SELECT CONCAT\_WS('\_', 'geeks', 'for', 'geeks');

**Output:** geeks\_for\_geeks

**6. FIND\_IN\_SET():** This function is used to find a symbol from a set of symbols.

**Syntax:** SELECT FIND\_IN\_SET('b', 'a, b, c, d, e, f');

**Output:** 2

## NOTES

NOTES

**7. FORMAT():** This function is used to display a number in the given format.

**Syntax:** Format("0.981", "Percent");

**Output:** '98.10%'

**8. INSERT():** This function is used to insert the data into a database.

**Syntax:** INSERT INTO database (geek\_id, geek\_name) VALUES (5000, 'abc');

**Output:** successfully updated

**9. INSTR():** This function is used to find the occurrence of an alphabet.

**Syntax:** INSTR('geeks for geeks', 'e');

**Output:** 2 (the first occurrence of 'e')

**Syntax:** INSTR('geeks for geeks', 'e', 1, 2 );

**Output:** 3 (the second occurrence of 'e')

**10. LCASE():** This function is used to convert the given string into lower case.

**Syntax:** LCASE ("GeeksFor Geeks To Learn");

**Output:** geeksforgeeks to learn

**11. LEFT():** This function is used to SELECT a sub string from the left of given size or characters.

**Syntax:** SELECT LEFT('geeksforgeeks.org', 5);

**Output:** geeks

**12. LENGTH():** This function is used to find the length of a word.

**Syntax:** LENGTH('GeeksForGeeks');

**Output:** 13

**13. LOCATE():** This function is used to find the nth position of the given word in a string.

**Syntax:** SELECT LOCATE('for', 'geeksforgeeks', 1);

**Output:** 6

**14.LOWER():** This function is used to convert the upper case string into lower case.

**Syntax:** SELECT LOWER('GEEKSFORGEEKS.ORG');

**Output:** geeksforgeeks.org



**15.LPAD():** This function is used to make the given string of the given size by adding the given symbol.

**Syntax:** LPAD('geeks', 8, '0');

**Output:** 000geeks

**16.LTRIM():** This function is used to cut the given sub string from the original string.

**Syntax:** LTRIM('123123geeks', '123');

**Output:** geeks

**17. MID():** This function is to find a word from the given position and of the given size.

**Syntax:** Mid ("geeksforgeeks", 6, 2);

**Output:** for

**18. POSITION():** This function is used to find position of the first occurrence of the given alphabet.

**Syntax:** SELECT POSITION('e' IN 'geeksforgeeks');

**Output:** 2

**19.REPEAT():** This function is used to write the given string again and again till the number of times mentioned.

**Syntax:** SELECT REPEAT('geeks', 2);

**Output:** geeksgeeks

**20. REPLACE():** This function is used to cut the given string by removing the given sub string.

**Syntax:** REPLACE('123geeks123', '123');

**Output:** geeks

**21.REVERSE():** This function is used to reverse a string.

**Syntax:** SELECT REVERSE('geeksforgeeks.org');

**Output:** 'gro.skeegrofskeeg'

**22.RIGHT():** This function is used to SELECT a sub string from the right end of the given size.

**Syntax:** SELECT RIGHT('geeksforgeeks.org', 4);

**Output:** '.org'

**23.RPAD():** This function is used to make the given string as long as the given size by adding the given symbol on the right.

## NOTES

NOTES

**Syntax:** RPAD('geeks', 8, '0');

**Output:** 'geeks000'

**24.RTRIM():** This function is used to cut the given sub string from the original string.

**Syntax:** RTRIM('geeksxyxzyyy', 'xyz');

**Output:** 'geeks'

**25.SPACE():** This function is used to write the given number of spaces.

**Syntax:** SELECT SPACE(7);

**Output:** '      '

**26.STRCMP():** This function is used to compare 2 strings.

If string1 and string2 are the same, the STRCMP function will return 0.

If string1 is smaller than string2, the STRCMP function will return -1.

If string1 is larger than string2, the STRCMP function will return 1.

**Syntax:** SELECT STRCMP('google.com', 'geeksforgeeks.com');

**Output:** -1

**27. SUBSTR():** This function is used to find a sub string from the a string from the given position.

**Syntax:**SUBSTR('geeksforgeeks', 1, 5);

**Output:** 'geeks'

**28. SUBSTRING():** This function is used to find an alphabet from the mentioned size and the given string.

**Syntax:** SELECT SUBSTRING('GeeksForGeeks.org', 9, 1);

**Output:** 'G'

**29. SUBSTRING\_INDEX():** This function is used to find a sub string before the given symbol.

**Syntax:** SELECT SUBSTRING\_INDEX('www.geeksforgeeks.org', '.', 1);

**Output:** 'www'

**30.TRIM():** This function is used to cut the given symbol from the string.

**Syntax:** TRIM(LEADING '0' FROM '000123');

**Output:** 123

**31.UCASE():** This function is used to make the string in upper case.

**Syntax:** UCASE ("GeeksForGeeks");

**Output:**GEEKSFORGEEKS

---

## 5.4 DATE AND TIME FUNCTIONS

---

In SQL, dates are complicated for newbies, since while working with database, the format of the date in table must be matched with the input date in order to insert. In various scenarios instead of date, datetime (time is also involved with date) is used.

In MySql the default date functions are:

**NOW():** Returns the current date and time. Example:

**SELECT NOW();**

**Output:**

2017-01-13 08:03:52

**CURDATE():** Returns the current date. Example:

**SELECT CURDATE();**

**Output:**

2017-01-13

**CURTIME():** Returns the current time. Example:

**SELECT CURTIME();**

**Output:**

08:05:15

**DATE():** Extracts the date part of a date or date/time expression. Example:For the below table named 'Test'

Id	Name	BirthTime
4120	Pratik	1996-09-26 16:44:15.581

**SELECT Name, DATE(BirthTime) AS BirthDate FROM Test;**

**Output:**

Name BirthDate

Pratik 1996-09-26

**EXTRACT():** Returns a single part of a date/time. Syntax:

**EXTRACT(unit FORM date);**

## NOTES

NOTES

**Check your Progress**

1. Mention the use of SELECT statement and also Mention it clauses.
2. Name any 6 String Operations in SQL.
3. Explain the syntax with example of MID() and POSITION()
4. Write short notes on Date and Time Functions.
5. What is the difference between SUBSTR() and SUBSTRING()

---

**5.5 ANSWERS TO CHECK YOUR PROGRESS**

---

1. MySQL is the most popular open source SQL database management system (DBMS). A fast, reliable, easy-to-use, multi-user multi-threaded relational database system.
2. The string operations in SQL are
  - i. ASCII()
  - ii. CHAR\_LENGTH()
  - iii. CHARACTER\_LENGTH()
  - iv. CONCAT()
  - v. FIND\_IN\_SET()
  - vi. REPLACE()
3. Different syntaxes are
  - i. MID(): This function is to find a word from the given position and of the given size.
    - Syntax: Mid ("geeksforgeeks", 6, 2);
    - Output: for
  - ii. POSITION(): This function is used to find position of the first occurrence of the given alphabet.
    - Syntax: SELECT POSITION('e' IN 'geeksforgeeks');
    - Output: 2
4. The Date and Time Functions are given below.
  - i. NOW(): Returns the current date and time. Example:  
SELECT NOW();  
  
Output:  
  
2017-01-13 08:03:52
  - ii. CURDATE(): Returns the current date. Example:  
SELECT CURDATE();  
  
Output:  
  
2017-01-13
  - iii. CURTIME(): Returns the current time. Example:  
SELECT CURTIME();  
  
Output:  
  
08:05:15
5. Two of them are defined below.

- i. SUBSTR(): This function is used to find a sub string from the a string from the given position.  
Syntax:SUBSTR('geeksforgeeks', 1, 5);  
Output: 'geeks'
- ii. SUBSTRING(): This function is used to find an alphabet from the mentioned size and the given string.

## NOTES

---

### 5.6 SUMMARY

---

- The SELECT statement is used to select data from a database. The statement begins with the SELECT keyword.
- String functions are used to perform an operation on input string and return an output string.
- In SQL, dates are complicated for newbies, since while working with database, the format of the date in table must be matched with the input date in order to insert.
- With the addition of command DESC we can change the order of display to keep the highest mark at the top of the list and lowest mark at the bottom of the list.

---

### 5.7 KEYWORDS

---

**SUBSTR():** This function is used to find a sub string from the a string from the given position.

**RTRIM():** This function is used to cut the given sub string from the original string.

**POSITION():** This function is used to find position of the first occurrence of the given alphabet

---

### 5.8 SELF ASSESSMENT QUESTIONS AND EXERCISES

---

**Short Answer questions:**

1. Write Appropriate syntax to select first 3 record from table.
2. Write Appropriate syntax to sort the table (student mark) with Rank
3. What is difference between TRIM() and RTRIM()
4. What are the units Considered in Date and Time?

**Long Answer questions:**

1. Briefly Explain Strings and its Functions with example.
2. Explain About Record Selection Technology.

---

### 5.9 FURTHER READINGS

---

Rémy Card, Eric Dumas, and Franck Mével. The Linux kernel book. John Wiley & Sons, Inc., 2003.

Steve Suchring. MySQL Bible. John Wiley, 2002.

Rasmus Lerdorf and Levin Tatroe. Programming PHP. " O'Reilly Media, Inc., 2002.

Wesley J. Chun. Core Python Programming. Prentice Hall, 2001.

Martin C. Brown. Perl: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

Steven Holzner. PHP: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

Vikram Vaswani. MySQL: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

---

# UNIT 6

## WORKING WITH SQL

---

### Structure

- 6.0 Introduction
- 6.1 Objective
- 6.2 Sorting query Results
- 6.3 Generating Summary
- 6.4 Working with Metadata
- 6.5 Using Sequences
- 6.6 My SQL and Web
  - 6.6.1 Basic Web Page Generation
- 6.7 Answers to Check Your Progress
- 6.8 Summary
- 6.9 Keywords
- 6.10 Self Assessment Questions and Exercises
- 6.11 Further Readings

### NOTES

---

## 6.0 INTRODUCTION

---

This unit helps the user to use and query with MySQL by sorting the queries using the sorting commands and the commands used with the web environment for development. It also explains the metadata and how to generate the summary from the tables.

---

## 6.1 OBJECTIVE

---

This unit explains to users the following concepts

- Learn the sorting queries
- To understand metadata
- Work with sequences
- Working with web environment

---

## 6.2 SORTING QUERY RESULTS

---

This MySQL tutorial explains how to use the MySQL ORDER BY clause with syntax and examples.

### Description

The MySQL ORDER BY clause is used to sort the records in your result set.

### Syntax

The syntax for the ORDER BY clause in MySQL is:

SELECT expressions

FROM tables

[WHERE conditions]

ORDER BY expression [ ASC | DESC ];

### **Parameters or Arguments**

#### **Expressions**

The columns or calculations that you wish to retrieve.

#### **Tables**

The tables that you wish to retrieve records from. There must be at least one table listed in the FROM clause.

#### **WHERE conditions**

Optional. The conditions that must be met for the records to be selected.

#### **ASC**

Optional. It sorts the result set in ascending order by expression (default, if no modifier is provided).

#### **DESC**

Optional. It sorts the result set in descending order by expression.

If the ASC or DESC modifier is not provided in the ORDER BY clause, the results will be sorted by expression in ascending order. This is equivalent to ORDER BY expression ASC. The ORDER BY clause can be used in a SELECT statement, SELECT LIMIT statement, and DELETE LIMIT statement in MySQL.

Example - Sorting without using ASC/DESC attribute

The MySQL ORDER BY clause can be used without specifying the ASC or DESC modifier. When this attribute is omitted from the ORDER BY clause, the sort order is defaulted to ASC or ascending order.

For example:

SELECT city

FROM customers

WHERE customer\_name = 'Apple'

ORDER BY city;



This MySQL ORDER BY example would return all records sorted by the city field in ascending order and would be equivalent to the following ORDER BY clause:

```
SELECT city
FROM customers
WHERE customer_name = 'Apple'
ORDER BY city ASC;
```

### **Sorting Subsets of a Table**

You don't want to sort an entire table, just part of it. Add a WHERE clause that selects only the records you want to see.

```
mysql> SELECT trav_date, miles FROM driver_log WHERE name =
'Henry' ORDER BY trav_date
```

### **Sorting and NULL Values**

To sort columns that may contain NULL values

```
mysql> SELECT NULL = NULL;
```

### **Sorting by Calendar Day**

To sort by day of the calendar year. Sort using the month and day of a date, ignoring the year. Sorting in calendar order differs from sorting by date.

```
mysql> SELECT date, description FROM event ORDER BY date;
```

---

## **6.3 GENERATING SUMMARY**

---

Database systems are useful for data storage and retrieval, but can also summarize your data in more concise forms. Summaries are useful when you want the overall picture, not the details. They're more readily understood than a long list of records. They enable you to answer questions such as "How many?" or "What is the total?" or "What is the range of values?" If you run a business, you may want to know how many customers you have in each state, or how much sales volume you generate each month.

The preceding examples include two common summary types: counting summaries and content summaries. The first (the number of customer records per state) is a counting summary. The content of each record is important only for purposes of placing it into the proper group or category for counting. Such summaries are essentially histograms, where you sort items into a set of bins and count the number of items in each bin. The second example (sales volume per month) is a content summary, in which sales totals are based on sales values in order records.

Another summary type produces neither counts nor sums, but simply a list of unique values. This is useful if you care which values are present rather

than how many of each there are. To determine the states in which you have customers, you need a list of the distinct state names contained in the records, not a list consisting of the state value from every record.

## NOTES

Using aggregate function we can achieve summary of values. Aggregate functions are COUNT (), MIN (), MAX (), SUM. (), AVG () and GROUP BY clause to group the rows into subsets and obtain an aggregate value for each one. To getting a list of unique values, use SELECT DISTINCT rather than SELECT. Using COUNT (DISTINCT) - To count how many distinct values there are

### Summarizing with COUNT()

To count the number of rows in an entire table or that match particular conditions, use the

COUNT() function.

For example,

```
mysql> SELECT COUNT(*) FROM emp_tab;
```

### Summarizing with MIN() and MAX()

Finding smallest or largest values in an entire table, use the MIN () and MAX () function. For example,

```
mysql> SELECT MIN(Sal) AS low, MAX(sal) AS  
high FROM emp_tab;
```

### Summarizing with SUM() and AVG()

SUM( ) and AVG( ) produce the total and average (mean) of a set of values:

For Example,

```
mysql> SELECT SUM(rno) AS 'No-Of Emp', AVG(sal) AS 'Avg Sal'  
FROM
```

### Sorting Query Results

SQL SELECT command to fetch data from MySQL table. When you select rows, the MySQL server is free to return them in any order, unless you instruct it otherwise by saying how to sort the result. But a query doesn't come out in the order you want

---

## 6.4 WORKING WITH METADATA

---

Metadata is data about data.

For example - Consider a file with a picture. The picture or the pixels inside the file are data. A description of the picture, like "JPEG format, 300x400 pixels, 72dpi", is metadata, because it describes the content of the file, although it's not the actual data

### 6.4.1 Types of Metadata

- Information about the result of queries-This includes the number of records affected by any SELECT, UPDATE or DELETE statement.
- Information about the tables and databases – This includes information pertaining to the structure of the tables and the databases.
- Information about the MySQL server – This includes the status of the database server, version number, etc.

### 6.4.2 Obtaining Metadata with SHOW

MySQL provides a SHOW statement that displays many types of database metadata. SHOW is helpful for keeping track of the contents of your databases and reminding yourself about the structure of your tables. The following examples demonstrate a few uses for SHOW statements.

List the databases you can access:

```
SHOW DATABASES;
```

Display the CREATE DATABASE statement for a database:

```
SHOW CREATE DATABASE db_name;
```

List the tables in the default database or a given database:

```
SHOW TABLES;
```

```
SHOW TABLES FROM db_name;
```

### 6.4.3 Obtaining Metadata with INFORMATION\_SCHEMA

Another way to obtain information about databases is to access the INFORMATION\_SCHEMA database. INFORMATION\_SCHEMA is based on the SQL standard. That is, the access mechanism is standard, even though some of the content is MySQL-specific. This makes INFORMATION\_SCHEMA more portable than the various SHOW statements, which are entirely MySQL-specific.

You can think of INFORMATION\_SCHEMA as a virtual database in which the tables are views for different kinds of database metadata. To see what tables INFORMATION\_SCHEMA contains, use SHOW TABLES.

---

## 6.5 USING SEQUENCES

---

A sequence is a database object that generates numbers in sequential order. Applications most often use these numbers when they require a unique value in a table such as primary key values. The following list describes the characteristics of sequences.

### 6.5.1 Creating a Sequence

```
CREATE SEQUENCE sequence_name
```

## NOTES

[INCREMENT BY #]

[START WITH #]

[MAXVALUE # | NOMAXVALUE]

[MINVALUE # | NOMINVALUE]

[CYCLE | NOCYCLE]

### 6.5.2 Dropping a Sequence

DROP SEQUENCE my\_sequence

Use sequences when an application requires a unique identifier. INSERT statements, and occasionally UPDATE statements, are the most common places to use sequences. Two "functions" are available on sequence

**NEXTVAL:** Returns the next value from the sequence.

**CURVAL:** Returns the value from the last call to NEXTVAL by the current user during the current connection

#### Examples

To create the sequence:  
CREATE SEQUENCE customer\_seq INCREMENT BY 1 START WITH 100

To use the sequence to enter a record into the database:  
INSERT INTO customer (cust\_num, name, address)  
VALUES (customer\_seq.NEXTVAL, 'Kalam', '123 Gandhi Nagar.')

---

## 6.6 MY SQL AND WEB

---

MySQL makes it easier to provide dynamic rather than static content. Static content exists as pages in the web server's document that are served exactly as is. Visitors can access only the documents that you place in the tree, and changes occur only when you add, modify, or delete those documents. By contrast, dynamic content is created on demand

### 6.6.1 Basic Web Page Generation

Using HTML we can generate your own Web site. HTML is a language for describing web pages. HTML stands for Hyper Text Markup Language. HTML is not a programming language, it is a markup language. A markup language is a set of markup tags. HTML uses markup tags to describe web pages

#### HTML

- HTML documents describe web pages
- HTML documents contain HTML tags and plain text
- HTML documents are also called web pages

#### Static Web Page

A static web page shows the required information to the viewer, but do not accept any information from the viewer

### **Dynamic Web Page**

A dynamic web page displays the information to the viewer and also accepts the information from the user Railway reservation, Online shopping etc. are examples of dynamic web page.

### **Client side scripting**

It is a script, (ex. Javascript, VB script), that is executed by the browser (i.e. Firefox, Internet Explorer, Safari, Opera, etc.) that resides at the user computer

### **Server side scripting**

It is a script (ex. ASP .NET, ASP, JSP, PHP, Ruby, or others), is executed by the server (Web Server), and the page that is sent to the browser is produced by the serve-side scripting.

### **Using Apache to Run Web Scripts**

Open-Source Web server originally based on NCSA server(National Center for Supercomputing Applications).

Apache is the most widely used web server software package in the world.

Apache is highly configurable and can be setup to support technologies such as, password protection, virtual hosting (name based and IP based), and SSL encryption.

NOTES

#### **Check your Progress**

1. Which clause is used to Sort the Records? What is the Use of where Condition?
2. Write short note on Client-Side scripting.
3. Where SUMMARY Generating is used?
4. What are the types of Metadata?
5. What is the use of SHOW in SQL?

---

## **6.7 ANSWERS TO CHECK YOUR PROGRESS**

---

1. The MySQL ORDER BY clause is used to sort the records in your result set.  
The conditions that must be met for the records to be selected.
2. It is a script, (ex. Javascript, VB script), that is executed by the browser (i.e. Firefox, Internet Explorer, Safari, Opera, etc.) that resides at the user computer
3. Summaries are useful when you want the overall picture, not the details. They're more readily understood than a long list of records. They enable you to answer questions such as "How many?" or "What is the total?" or "What is the range of values?"

Self-Instructional  
Material

4. The types of metadata are
  - Information about the result of queries
  - Information about the tables and databases
  - Information about the MySQL server
5. SHOW statement that displays many types of database metadata. SHOW is helpful for keeping track of the contents of your databases and reminding yourself about the structure of your tables.
- 6.

---

## 6.8 SUMMARY

---

- Database systems are useful for data storage and retrieval, but can also summarize your data in more concise forms.
- Information about the result of queries-This includes the number of records affected by any SELECT, UPDATE or DELETE statement.
- A sequence is a database object that generates numbers in sequential order. Applications most often use these numbers when they require a unique value in a table such as primary key values.
- MySQL makes it easier to provide dynamic rather than static content.

---

## 6.9 KEYWORDS

---

**Sorting Query Results:** SQL SELECT command to fetch data from MySQL table. When you select rows, the MySQL server is free to return them in any order, unless you instruct it otherwise by saying how to sort the result.

**Static Web Page:** A static web page shows the required information to the viewer, but do not accept any information from the viewer

**Dynamic Web Page:** A dynamic web page displays the information to the viewer and also accepts the information from the user Railway reservation, Online shopping etc. are examples of dynamic web page.

---

## 6.10 SELF ASSESSMENT QUESTIONS AND EXERCISES

---

### Short Answer questions:

1. How to create a sequence?
2. What is Dropping a Sequence and what are their Functions.
3. Write short notes on HTML.
4. What is the Difference Between Static and Dynamic web Pages?
5. Write short nodes on Server-side Scripting.

### Long Answer questions:

1. Explain the Process of Basic Web Page Generation.
2. Explain briefly about Generating Summary with Example.

## 6.11 FURTHER READINGS

Rémy Card, Eric Dumas, and Franck Mével. The Linux kernel book. John Wiley & Sons, Inc., 2003.

Steve Suchring. MySQL BBible. John Wiley, 2002.

Rasmus Lerdorf and Levin Tatroe. Programming PHP. " O'Reilly Media, Inc., 2002.

Wesley J. Chun. Core Python Programming. Prentice Hall, 2001.

Martin C. Brown. Perl: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

Steven Holzner. PHP: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

Vikram Vaswani. MySQL: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

## NOTES

Self-Instructional  
Material

---

**BLOCK – III**  
**OPEN SOURCE PROGRAMMING LANGUAGE PHP**

---

**UNIT 7**  
**PHP**

---

**NOTES****Structure**

- 7.0 Introduction
- 7.1 Objective
- 7.2 PHP
- 7.3 Programming in Web Environment
- 7.4 PHP Variables
- 7.5 Constants
- 7.6 Data Types
- 7.7 Operators
- 7.8 Statement
- 7.9 Answers to Check Your Progress
- 7.10 Summary
- 7.11 Keywords
- 7.12 Self Assessment Questions and Exercises
- 7.13 Further Readings

---

**7.0 INTRODUCTION**

---

Open source programming languages plays an pivotal role in creation of real time applications as the software is freely available and it can be used with good packages. The worked modules can be shared for later use of developers. In this unit, the open source programming language PHP is explained with the basics of programming

---

**7.1 OBJECTIVE**

---

After reading this unit you will be able to

- Understand PHP
- Learn the web environment
- Basics of PHP

---

**7.2 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 Pre-processor". 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.

### **7.2.1 Common uses of PHP**

PHP performs system functions, i.e. from files on a system it can create, open, read, write, and close them. PHP can handle forms, i.e. gather data from files, save data to a file, through email you can send data, return data to the user. You add, delete, modify elements within your database through PHP. Using PHP, you can restrict users to access some pages of your website. It can encrypt data.

### **7.2.2 Characteristics of PHP**

Five important characteristics make PHP's practical nature possible

- Simplicity
- Efficiency
- Security
- Flexibility
- Familiarity

---

## **7.3 PROGRAMMING IN WEB ENVIRONMENT**

---

PHP programs are written using a text editor, such as Notepad, Simple Text, or vi, just like HTML pages. However, unlike HTML, PHP files end with a .php extension. This extension signifies to the server that it needs to parse the PHP code before sending the resulting HTML code to the viewer's web browser. In PHP, on the fly method is adopted to publish the document. Hence, the PHP developer can generate not only web pages, but also other web embedding documents like PDF, PNG, GIF, etc. The PHP web environment is usually set, with AMP (Apache, MySQL, and PHP/Perl/Python), which are linked together.

PHP not only allows HTML pages to be created on the fly, but it is invisible to your web site visitors. The only thing they see when they view the source of your code is the resulting HTML output. In this respect, PHP gives you a bit more security by hiding your programming logic. HTML can also be written inside the PHP code of your page, which allows you to format text while keeping blocks of code together. This will also help you write organized, efficient code, and the browser (and, more importantly, the person viewing the site) won't know the difference.

**NOTES**

Self-Instructional  
Material

PHP can also be written as a standalone program with no HTML at all. This is helpful for storing your connection variables, redirecting your visitors to another page of your site, or performing other functions

---

## 7.4 PHP VARIABLES

---

### NOTES

The main way to store information in the middle of a PHP program is by using a variable. Here are the most important things to know about variables in PHP. All variables in PHP are denoted with a leading dollar sign (\$). The value of a variable is the value of its most recent assignment. Variables are assigned with the = operator, with the variable on the left-hand side and the expression to be evaluated on the right. Variables can, but do not need, to be declared before assignment. Variables in PHP do not have intrinsic types - a variable does not know in advance whether it will be used to store a number or a string of characters.

Variables used before they are assigned have default values. PHP does a good job of automatically converting types from one to another when necessary. PHP variables are Perl-like.

### 7.4.1 Variable Scope

Scope can be defined as the range of availability a variable has to the program in which it is declared. PHP variables can be one of four scope types –

- Local variables
- Function parameters
- Global variables
- Static variables
- Variable Naming

Rules for naming a variable is:

- Variable names must begin with a letter or underscore character.
- A variable name can consist of numbers, letters, underscores but you cannot use characters like + , - , % , ( , ) . & , etc
- There is no size limit for variables.

---

## 7.5 CONSTANTS

---

A constant is a name or an identifier for a simple value. A constant value cannot change during the execution of the script. By default, a constant is case-sensitive. By convention, constant identifiers are always uppercase. A constant name starts with a letter or underscore, followed by any number of letters, numbers, or underscores. If you have defined a constant, it can never be changed or undefined.

To define a constant you have to use define() function and to retrieve the value of a constant, you have to simply specifying its name. Unlike with variables, you do not need to have a constant with a \$. You can also use the

function `constant()` to read a constant's value if you wish to obtain the constant's name dynamically.

`constant()` function

As indicated by the name, this function will return the value of the constant. This is useful when you want to retrieve value of a constant, but you do not know its name, i.e. It is stored in a variable or returned by a function.

`constant()` example

```
<?php
    define("MINSIZE", 50);

    echo MINSIZE;

    echo constant("MINSIZE"); // same thing as the previous line
?>
```

Only scalar data (boolean, integer, float and string) can be contained in constants.

Differences between constants and variables are There is no need to write a dollar sign (\$) before a constant, where as in Variable one has to write a dollar sign. Constants cannot be defined by simple assignment, they may only be defined using the `define ()` function. Constants may be defined and accessed anywhere without regard to variable scoping rules.

Once the Constants have been set, may not be redefined or undefined.  
Valid and invalid constant names

// Valid constant names

```
define("ONE", "first thing");
define("TWO2", "second thing");
define("THREE_3", "third thing");
```

// Invalid constant names

```
define("2TWO", "second thing");
define("__THREE__", "third value");
```

### 7.5.1 PHP Magic constants

PHP provides a large number of predefined constants to any script which it runs. There are five magical constants that change depending on where they are used. For example, the value of `LINE` depends on the line that it's used on in your script. These special constants are case-insensitive and are as follows.

A few "magical" PHP constants are given below

NOTES

Self-Instructional  
Material

## NOTES

**LINE**- The current line number of the file.

**FILE**- The full path and filename of the file. If used inside an include, the name of the included file is returned. Since PHP 4.0.2, `__FILE__` always contains an absolute path whereas in older versions it contained relative path under some circumstances.

**FUNCTION**- The function name. (Added in PHP 4.3.0) As of PHP 5 this constant returns the function name as it was declared (case-sensitive). In PHP 4 its value is always lowercased.

**CLASS**-The class name. (Added in PHP 4.3.0) As of PHP 5 this constant returns the class name as it was declared (case-sensitive). In PHP 4 its value is always lowercased.

**METHOD**- The class method name. (Added in PHP 5.0.0) The method name is returned as it was declared (case-sensitive).

---

## 7.6 DATA TYPES

---

PHP has a total of eight data types which we use to construct our variables

**Integers** – are whole numbers, without a decimal point, like 4195.

**Doubles** – are floating-point numbers, like 3.14159 or 49.1.

**Booleans** – have only two possible values either true or false.

**NULL** – is a special type that only has one value: NULL.

**Strings** – are sequences of characters, like 'PHP supports string operations.'

**Arrays** – are named and indexed collections of other values.

**Objects** – are instances of programmer-defined classes, which can package up both other kinds of values and functions that are specific to the class.

**Resources** – are special variables that hold references to resources external to PHP (such as database connections).

The first five are simple types, and the next two (arrays and objects) are compound - the compound types can package up other arbitrary values of arbitrary type, whereas the simple types cannot. We will explain only simple data type in this chapters. Array and Objects will be explained separately.

### 7.6.1 Integers

They are whole numbers, without a decimal point, like 4195. They are the simplest type .they correspond to simple whole numbers, both positive and negative. Integers can be assigned to variables, or they can be used in expressions, like so –

```
$int_var = 12345;
```

```
$another_int = -12345 + 12345;
```

Integer can be in decimal (base 10), octal (base 8), and hexadecimal (base 16) format. Decimal format is the default, octal integers are specified with a leading 0, and hexadecimals have a leading 0x. For most common platforms, the largest integer is  $(2^{31} - 1)$  (or 2,147,483,647), and the smallest (most negative) integer is  $-(2^{31} - 1)$  (or -2,147,483,647).

### 7.6.2 Doubles

They like 3.14159 or 49.1. By default, doubles print with the minimum number of decimal places needed. For example, the code –

Live Demo

```
<?php
    $many = 2.2888800;
    $many_2 = 2.2111200;
    $few = $many + $many_2;
    print("$many + $many_2 = $few <br>");
?>
```

It produces the following browser output –

```
2.28888 + 2.21112 = 4.5
```

### 7.6.3 Boolean

They have only two possible values either true or false. PHP provides a couple of constants especially for use as Booleans: TRUE and FALSE, which can be used like so –

```
if (TRUE)
    print("This will always print<br>");
else
    print("This will never print<br>");
```

Interpreting other types as Booleans. Here are the rules for determine the "truth" of any value not already of the Boolean type – If the value is a number, it is false if exactly equal to zero and true otherwise. If the value is a string, it is false if the string is empty (has zero characters) or is the string "0", and is true otherwise. Values of type NULL are always false.

If the value is an array, it is false if it contains no other values, and it is true otherwise. For an object, containing a value means having a member variable that has been assigned a value. Valid resources are true (although some functions that return resources when they are successful will return FALSE when unsuccessful). Don't use double as Booleans.

## NOTES

Each of the following variables has the truth value embedded in its name when it is used in a Boolean context.

```
$true_num = 3 + 0.14159;  
$true_str = "Tried and true"  
$true_array[49] = "An array element";  
$false_array = array();  
$false_null = NULL;  
$false_num = 999 - 999;  
$false_str = "";
```

#### 7.6.4 NULL

NULL is a special type that only has one value: NULL. To give a variable the NULL value, simply assign it like this –

```
$my_var = NULL;
```

The special constant NULL is capitalized by convention, but actually it is case insensitive; you could just as well have typed –

```
$my_var = null;
```

A variable that has been assigned NULL has the following properties –It evaluates to FALSE in a Boolean context. It returns FALSE when tested with IsSet() function.

#### 7.6.5 Strings

They are sequences of characters, like "PHP supports string operations". Following are valid examples of string

```
$string_1 = "This is a string in double quotes";  
$string_2 = 'This is a somewhat longer, singly quoted string';  
$string_39 = "This string has thirty-nine characters";  
$string_0 = ""; // a string with zero characters
```

Singly quoted strings are treated almost literally, whereas doubly quoted strings replace variables with their values as well as specially interpreting certain character sequences.

Live Demo

```
<?php  
$variable = "name";  
$literally = 'My $variable will not print!';
```

```
print($literally);  
print "<br>";  
$literally = "My $variable will print!";  
print($literally);  
?>
```

This will produce following result –

My \$variable will not print!

My name will print

### 7.6.6 Arrays

An array is a data structure that stores one or more similar type of values in a single value. For example if you want to store 100 numbers then instead of defining 100 variables its easy to define an array of 100 length.

There are three different kind of arrays and each array value is accessed using an ID c which is called array index.

- **Numeric array** – An array with a numeric index. Values are stored and accessed in linear fashion.
- **Associative array** – An array with strings as index. This stores element values in association with key values rather than in a strict linear index order.
- **Multidimensional array** – An array containing one or more arrays and values are accessed using multiple indices

---

## 7.7 OPERATORS

---

PHP language supports following type of operators.

- Arithmetic Operators
- Comparison Operators
- Logical (or Relational) Operators
- Assignment Operators
- Conditional (or ternary) Operators

## NOTES

Self-Instructional  
Material

### 7.7.1 Arithmetic Operators

There are following arithmetic operators supported by PHP language .Assume variable A holds 10 and variable B holds 20 then

Show Examples

Operator	Description	Example
+	Adds two operands	A + B will give 30
-	Subtracts second operand from the first	A - B will give -10
*	Multiply both operands	A * B will give 200
/	Divide numerator by de-numerator	B / A will give 2
%	Modulus Operator and remainder of after an integer division	B % A will give 0
++	Increment operator, increases integer value by one	A++ will give 11
--	Decrement operator, decreases integer value by one	A-- will give 9

### 7.7.2 Comparison Operators

There are following comparison operators supported by PHP language Assume variable A holds 10 and variable B holds 20 then

Show Examples

Operator	Description	Example
==	Checks if the value of two operands are equal or not, if yes then condition becomes true.	(A == B) is not true.
!=	Checks if the value of two operands are equal or not, if values are not equal then condition becomes true.	(A != B) is true.
>	Checks if the value of left operand is greater than the value of right operand, if yes then	(A > B) is not true.



	condition becomes true.	
<	Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true.	(A < B) is true.
>=	Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true.	(A >= B) is not true.
<=	Checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true.	(A <= B) is true.

### 7.7.3 Logical Operators

There are following logical operators supported by PHP language Assume variable A holds 10 and variable B holds 20 then –Show Examples

NOTES

Operator	Description	Example
and	Called Logical AND operator. If both the operands are true then condition becomes true.	(A and B) is true.
or	Called Logical OR Operator. If any of the two operands are non zero then condition becomes true.	(A or B) is true.
&&	Called Logical AND operator. If both the operands are non zero then condition becomes true.	(A && B) is true.
	Called Logical OR Operator. If any of the two operands are non zero then condition becomes true.	(A    B) is true.
!	Called Logical NOT Operator. Use to reverses the logical state of its operand. If a condition is true then Logical NOT operator will make false.	!(A && B) is false.

### 7.7.4 Assignment Operators

There are following assignment operators supported by PHP language  
 –Show Examples

Operator	Description	Example
=	Simple assignment operator, Assigns values from right side operands to left side operand	C = A + B will assign value of A + B into C
+=	Add AND assignment operator, It adds right operand to the left operand and assign the	C += A is equivalent to C = C + A

	result to left operand	
<code>-=</code>	Subtract AND assignment operator, It subtracts right operand from the left operand and assign the result to left operand	<code>C -= A</code> is equivalent to <code>C = C - A</code>
<code>*=</code>	Multiply AND assignment operator, It multiplies right operand with the left operand and assign the result to left operand	<code>C *= A</code> is equivalent to <code>C = C * A</code>
<code>/=</code>	Divide AND assignment operator, It divides left operand with the right operand and assign the result to left operand	<code>C /= A</code> is equivalent to <code>C = C / A</code>
<code>%=</code>	Modulus AND assignment operator, It takes modulus using two operands and assign the result to left operand	<code>C %= A</code> is equivalent to <code>C = C % A</code>

**NOTES****7.7.5 Conditional Operator**

There is one more operator called conditional operator. This first evaluates an expression for a true or false value and then execute one of the two given statements depending upon the result of the evaluation. The conditional operator has this syntax –

Show Examples

Operator	Description	Example
<code>?:</code>	Conditional Expression	If Condition is true ? Then value X : Otherwise value Y

**7.7.6 Operators Categories**

All the operators we have discussed above can be categorised into following categories

**NOTES**

- Unary prefix operators, which precede a single operand.
- Binary operators, which take two operands and perform a variety of arithmetic and logical operations.
- The conditional operator (a ternary operator), which takes three operands and evaluates either the second or third expression, depending on the evaluation of the first expression.
- Assignment operators, which assign a value to a variable.

**7.7.7 Precedence of PHP Operators**

Operator precedence determines the grouping of terms in an expression. This affects how an expression is evaluated. Certain operators have higher precedence than others; for example, the multiplication operator has higher precedence than the addition operator

For example  $x = 7 + 3 * 2$ ; Here x is assigned 13, not 20 because operator \* has higher precedence than + so it first get multiplied with 3\*2 and then adds into 7. Here operators with the highest precedence appear at the top of the table, those with the lowest appear at the bottom. Within an expression, higher precedence operators will be evaluated first.

Category	Operator	Associativity
Unary	! ++ --	Right to left
Multiplicative	* / %	Left to right
Additive	+ -	Left to right
Relational	< <= > >=	Left to right
Equality	== !=	Left to right
Logical AND	&&	Left to right
Logical OR		Left to right
Conditional	?:	Right to left
Assignment	= += -= *= /= %=	Right to left

**7.8 STATEMENT**

A prepared statement is a feature used to execute the same (or similar) SQL statements repeatedly with high efficiency. Prepared statements basically work like this:

**Prepare:** An SQL statement template is created and sent to the database. Certain values are left unspecified, called parameters (labeled "?"). Example: `INSERT INTO MyGuests VALUES(?, ?, ?)` The database parses,

compiles, and performs query optimization on the SQL statement template, and stores the result without executing it

Execute: At a later time, the application binds the values to the parameters, and the database executes the statement. The application may execute the statement as many times as it wants with different values. Compared to executing SQL statements directly, prepared statements have three main advantages:

- Prepared statements reduce parsing time as the preparation on the query is done only once (although the statement is executed multiple times).
- Bound parameters minimize bandwidth to the server as you need send only the parameters each time, and not the whole query. Prepared statements are very useful against SQL injections, because parameter values, which are transmitted later using a different protocol, need not be correctly escaped.
- If the original statement template is not derived from external input, SQL injection cannot occur.

## NOTES

### Check your Progress

1. Write short notes on PHP?
2. What are the Characteristics of PHP?
3. What is Variable Scope? Mention in types.
4. What are Constants?

---

## 7.9. ANSWERS TO CHECK YOUR PROGRESS

---

1. PHP started out as a small open source project that evolved as more and more people found out how useful it was. 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.
2. The characteristics of PHP are
  - Simplicity
  - Efficiency
  - Security
  - Flexibility
  - Familiarity
3. Scope can be defined as the range of availability a variable has to the program in which it is declared. PHP variables can be one of four scope types:

- Local variables
- Function parameters
- Global variables
- Static variables
- Variable Naming

## NOTES

4. A constant is a name or an identifier for a simple value. A constant value cannot change during the execution of the script. By default, a constant is case-sensitive.

---

## 7.10 SUMMARY

---

- An array is a data structure that stores one or more similar type of values in a single value. For example if you want to store 100 numbers then instead of defining 100 variables its easy to define an array of 100 length.
- Resources are special variables that hold references to resources external to PHP (such as database connections).
- To define a constant you have to use define() function and to retrieve the value of a constant, you have to simply specifying its name.

---

## 7.11 KEYWORDS

---

**Numeric array** – An array with a numeric index. Values are stored and accessed in linear fashion

**Boolean:** They have only two possible values either true or false. PHP provides a couple of constants especially for use as Booleans: TRUE and FALSE,

**Resources:** They are special variables that hold references to resources external to PHP (such as database connections).

**Constants:** A constant is a name or an identifier for a simple value. A constant value cannot change during the execution of the script.

---

## 7.12 SELF ASSESSMENT QUESTIONS AND EXERCISES

---

### Short Answer questions:

1. List some of the PHP variables?
2. Write short notes on constants.
3. What are the different data types?
4. Write short notes on PHP Magical constants.

**Long Answer questions:**

1. Explain the type of operators supported in PHP.
2. Explain briefly about the eight data types.

---

**7.13. FURTHER READINGS**

---

Rémy Card, Eric Dumas, and Franck Mével. The Linux kernel book. John Wiley & Sons, Inc., 2003.

Steve Suchring. MySQL BBible. John Wiley, 2002.

Rasmus Lerdorf and Levin Tatroe. Programming PHP. " O'Reilly Media, Inc., 2002.

Wesley J. Chun. Core Python Programming. Prentice Hall, 2001.

Martin C. Brown. Perl: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

Steven Holzner. PHP: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

Vikram Vaswani. MySQL: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

**NOTES**

---

# UNIT 8

## FUNCTIONS

---

**NOTES****Structure**

- 8.0 Introduction
- 8.1 Objective
- 8.2 Functions
- 8.3 Arrays
- 8.4 Object Oriented Concepts
- 8.5. String Manipulation and Regular Expression
- 8.6 File Handling and Data Storage
- 8.7 Answers to Check Your Progress
- 8.8 Summary
- 8.9 Keywords
- 8.10 Self Assessment Questions and Exercises
- 8.11 Further Readings

---

### 8.0 INTRODUCTION

---

The programming of PHP with functions helps for the manipulation of data in a much easier manner. The file handling functions and the storage of data is helpful for the usage of large databases. IN this unit, the functions of PHP and its usage with files and strings are explained.

---

### 8.1 OBJECTIVE

---

This unit helps the user to understand the following concepts

- Function
- Strings
- Arrays
- OOP
- Regular Expression
- File Handling

---

### 8.2 FUNCTIONS

---

PHP functions are similar to other programming languages. A function is a piece of code which takes one more input in the form of parameter and does some processing and returns a value. You already have seen many functions like fopen() and fread() etc. They are built-in functions but PHP gives you option to create your own functions as well. There are two parts which should be clear to you

- Creating a PHP Function
- Calling a PHP Function



In fact you hardly need to create your own PHP function because there are already more than 1000 of built-in library functions created for different area and you just need to call them according to your requirement. Please refer to PHP Function Reference for a complete set of useful functions.

### 8.2.1 Creating PHP Function

It's very easy to create your own PHP function. Suppose you want to create a PHP function which will simply write a simple message on your browser when you will call it. Following example creates a function called writeMessage() and then calls it just after creating it. Note that while creating a function its name should start with keyword function and all the PHP code should be put inside { and } braces as shown in the following example below –

Live Demo

```
<html>
  <head>
    <title>Writing PHP Function</title>
  </head>
  <body>
    <?php
      /* Defining a PHP Function */
      function writeMessage() {
        echo "You are really a nice person, Have a nice time!";
      }
      /* Calling a PHP Function */
      writeMessage();
    ?>
  </body>
</html>
```

This will display following result –

You are really a nice person, Have a nice time!

### 8.2.2 PHP Functions with Parameters

PHP gives you option to pass your parameters inside a function. You can pass as many as parameters your like. These parameters work like variables inside your function. Following example takes two integer parameters and add them together and then print them.

Live Demo

```
<html>
  <head>
    <title>Writing PHP Function with Parameters</title>
  </head>
  <body>
    <?php
      function addFunction($num1, $num2) {
        $sum = $num1 + $num2;
        echo "Sum of the two numbers is : $sum";
      }
      addFunction(10, 20);
    ?>
  </body>
</html>
```

This will display following result –

Sum of the two numbers is : 30

### 8.2.3 Passing Arguments by Reference

It is possible to pass arguments to functions by reference. This means that a reference to the variable is manipulated by the function rather than a copy of the variable's value. Any changes made to an argument in these cases will change the value of the original variable. You can pass an argument by reference by adding an ampersand to the variable name in either the function call or the function definition.

Following example depicts both the cases.

Live Demo

```
<html>
  <head>
    <title>Passing Argument by Reference</title>
  </head>
  <body>
    <?php
```

```

function addFive($num) {
    $num += 5;
}

function addSix(&$num) {
    $num += 6;
}

$orignum = 10;
addFive( $orignum );
echo "Original Value is $orignum<br />";
addSix( $orignum );
echo "Original Value is $orignum<br />";
?>
</body>
</html>

```

This will display following result –

Original Value is 10

Original Value is 16

## 8.2.4 PHP Functions Returning Value

A function can return a value using the return statement in conjunction with a value or object. return stops the execution of the function and sends the value back to the calling code. You can return more than one value from a function using return array(1,2,3,4). Following example takes two integer parameters and add them together and then returns their sum to the calling program. Note that return keyword is used to return a value from a function.

Live Demo

```

<html>
  <head>
    <title>Writing PHP Function which returns value</title>
  </head>
  <body>
    <?php

```

```

function addFunction($num1, $num2) {
    $sum = $num1 + $num2;
    return $sum;
}
$return_value = addFunction(10, 20);
echo "Returned value from the function : $return_value";
?>
</body>
</html>

```

This will display following result –Ret  
urned value from the function : 30

### 8.2.5 Setting Default Values for Function Parameters

You can set a parameter to have a default value if the function's caller doesn't pass it. Following function prints NULL in case use does not pass any value to this function.

Live Demo

```

<html>
    <head>
        <title>Writing PHP Function which returns value</title>
    </head>
    <body>
        <?php
            function printMe($param = NULL) {
                print $param;
            }
            printMe("This is test");
            printMe();
        ?>
    </body>
</html>

```

This will produce following result  
This is test

## 8.2.6 Dynamic Function Calls

It is possible to assign function names as strings to variables and then treat these variables exactly as you would the function name itself. Following example depicts this behaviour.

Live Demo

```
<html>
  <head>
    <title>Dynamic Function Calls</title>
  </head>
  <body>
    <?php
      function sayHello() {
        echo "Hello<br />";
      }
      $function_holder = "sayHello";
      $function_holder();
    ?>
  </body>
</html>
```

This will display following result

Hello

---

## 8.3 ARRAYS

---

An array is a data structure that stores one or more similar type of values in a single value. For example if you want to store 100 numbers then instead of defining 100 variables its easy to define an array of 100 length. is called array index.

- Numeric array – An array with a numeric index. Values are stored and accessed in linear fashion.
- Associative array – An array with strings as index. This stores element values in association with key values rather than in a strict linear index order.
- Multidimensional array – An array containing one or more arrays and values are accessed using multiple indices

### 8.3.1 Numeric Array

These arrays can store numbers, strings and any object but their index will be represented by numbers. By default array index starts from zero.

Example

#### NOTES

Following is the example showing how to create and access numeric arrays. Here we have used array() function to create array. This function is explained in function reference.

Live Demo

```
<html>
  <body>
    <?php
      /* First method to create array. */
      $numbers = array( 1, 2, 3, 4, 5);
      foreach( $numbers as $value ) {
        echo "Value is $value <br />";
      }
      /* Second method to create array. */
      $numbers[0] = "one";
      $numbers[1] = "two";
      $numbers[2] = "three";
      $numbers[3] = "four";
      $numbers[4] = "five";
      foreach( $numbers as $value ) {
        echo "Value is $value <br />";
      }
    ?>
  </body>
</html>
```

This will produce the following result –

Value is 1

Value is 2

Value is 3

Value is 4

Value is 5

Value is one

Value is two

Value is three

Value is four

Value is five

### 8.3.2 Associative Arrays

The associative arrays are very similar to numeric arrays in term of functionality but they are different in terms of their index. Associative array will have their index as string so that you can establish a strong association between key and values. To store the salaries of employees in an array, a numerically indexed array would not be the best choice. Instead, we could use the employees names as the keys in our associative array, and the value would be their respective salary.

Example

Live Demo

```
<html>
<body>
  <?php
    /* First method to associate create array. */
    $salaries = array("mohammad"=>2000,"qadir"=>1000,"zara"=>500);
    echo "Salary of mohammad is ". $salaries['mohammad'] . "<br />";
    echo "Salary of qadir is ". $salaries['qadir']. "<br />";
    echo "Salary of zara is ". $salaries['zara']. "<br />";
    /* Second method to create array. */
    $salaries['mohammad'] = "high";
    $salaries['qadir'] = "medium";
    $salaries['zara'] = "low";
    echo "Salary of mohammad is ". $salaries['mohammad'] . "<br />";
    echo "Salary of qadir is ". $salaries['qadir']. "<br />";
    echo "Salary of zara is ". $salaries['zara']. "<br />";
  ?>
</body>
</html>
```

This will produce the following result –

Salary of mohammad is 2000

Salary of qadir is 1000

Salary of zara is 500

Salary of mohammad is high

Salary of qadir is medium

Salary of zara is low

---

## 8.4 OBJECT ORIENTED CONCEPTS

---

Before we go in detail, let's define important terms related to Object Oriented Programming.

**Class** – this is a programmer-defined data type, which includes local functions as well as local data. You can think of a class as a template for making many instances of the same kind (or class) of object.

**Object** – an individual instance of the data structure defined by a class. You define a class once and then make many objects that belong to it. Objects are also known as instance.

**Member Variable** – these are the variables defined inside a class. This data will be invisible to the outside of the class and can be accessed via member functions. These variables are called attribute of the object once an object is created.

**Member function** – these are the function defined inside a class and are used to access object data.

**Inheritance** – When a class is defined by inheriting existing function of a parent class then it is called inheritance. Here child class will inherit all or few member functions and variables of a parent class.

**Parent class** – A class that is inherited from by another class. This is also called a base class or super class.

**Child Class** – A class that inherits from another class. This is also called a subclass or derived class.

**Polymorphism** – this is an object oriented concept where same function can be used for different purposes. For example function name will remain same but it takes different number of arguments and can do different task.

**Overloading** – a type of polymorphism in which some or all of operators have different implementations depending on the types of their arguments. Similarly functions can also be overloaded with different implementation.

**Data Abstraction** – any representation of data in which the implementation details are hidden (abstracted).

**Encapsulation** – refers to a concept where we encapsulate all the data and member functions together to form an object.

**Constructor** – refers to a special type of function which will be called automatically whenever there is an object formation from a class.

**Destructor** – refers to a special type of function which will be called automatically whenever an object is deleted or goes out of scope.



---

## 8.5 STRING MANIPULATION AND REGULAR EXPRESSION

---

Strings that are delimited by double quotes (as in "this") are pre-processed in both the following two ways by PHP. Certain character sequences beginning with backslash (\) are replaced with special characters. Variable names (starting with \$) are replaced with string representations of their values. The escape-sequence replacements are

\n is replaced by the newline character

\r is replaced by the carriage-return character

\t is replaced by the tab character

\\$ is replaced by the dollar sign itself (\$)

\" is replaced by a single double-quote (")

\\ is replaced by a single backslash (\)

### 8.5.1 String Concatenation Operator

To concatenate two string variables together, use the dot (.) operator

Live Demo

```
<?php
    $string1="Hello World";
    $string2="1234";
    echo $string1 . " " . $string2;
?>
```

This will produce the following result –

Hello World 1234

If we look at the code above you see that we used the concatenation operator two times. This is because we had to insert a third string. Between the two string variables we added a string with a single character, an empty space, to separate the two variables.

#### Using the strlen() function

The strlen() function is used to find the length of a string. Let's find the length of our string "Hello world!"

Live Demo

```
<?php
    echo strlen("Hello world!");
```

?>

This will produce the following result –

12

The length of a string is often used in loops or other functions, when it is important to know when the string ends. (i.e. in a loop, we would want to stop the loop after the last character in the string)

### Using the strpos() function

The strpos() function is used to search for a string or character within a string. If a match is found in the string, this function will return the position of the first match. If no match is found, it will return FALSE. Let's see if we can find the string "world" in our string

Live Demo

```
<?php
```

```
    echo strpos("Hello world!","world");
```

```
?>
```

This will produce the following result –

6

As you see the position of the string "world" in our string is position 6. The reason that it is 6, and not 7, is that the first position in the string is 0, and not 1

### Regular Expressions

Regular expressions are nothing more than a sequence or pattern of characters. They provide the foundation for pattern-matching functionality. Using regular expression you can search a particular string inside another string, you can replace one string by another string and you can split a string into many chunks. PHP offers functions specific to two sets of regular expression functions, each corresponding to a certain type of regular expression. You can use any of them based on your comfort.

- POSIX Regular Expressions
- PERL Style Regular Expressions
- POSIX Regular Expressions

The structure of a POSIX regular expression is not dissimilar to that of a typical arithmetic expression: various elements (operators) are combined to form more complex expressions.

The simplest regular expression is one that matches a single character, such as g, inside strings such as g, haggie, or bag. Let's give explanation for few

concepts being used in POSIX regular expression. After that we will introduce you with regular expression related functions.

## Brackets

Brackets ([]) have a special meaning when used in the context of regular expressions. They are used to find a range of characters.

[0-9]

It matches any decimal digit from 0 through 9.

## Quantifiers

The frequency or position of bracketed character sequences and single characters can be denoted by a special character. Each special character having a specific connotation. The +, \*, ?, {int. range}, and \$ flags all follow a character sequence.

Sr.No Expression & Description

1 p+ It matches any string containing at least one p

.

a p followed by zero or more instances of the sequence php.

## Predefined Character Ranges

For your programming convenience several predefined character ranges, also known as character classes, are available. Character classes specify an entire range of characters, for example, the alphabet or an integer set

Sr.No Expression & Description

1 [[:alpha:]] It matches any string containing alphabetic characters aA through zZ.

## PHP's Regexp POSIX Functions

PHP currently offers seven functions for searching strings using POSIX-style regular expressions

Sr.No Function & Description

1 ereg() The ereg() function searches a string specified by string for a string specified by pattern, returning true if the pattern is found, and false otherwise.

---

## 8.6 FILE HANDLING AND DATA STORAGE

---

It can include the content of a PHP file into another PHP file before the server executes it. There are two PHP functions which can be used to included one PHP file into another PHP file.

- The include() Function

- The require() Function

This is a strong point of PHP which helps in creating functions, headers, footers, or elements that can be reused on multiple pages. This will help developers to make it easy to change the layout of complete website with minimal effort. If there is any change required then instead of changing thousands of files just change included file.

### 8.6.1 The include() Function

The include() function takes all the text in a specified file and copies it into the file that uses the include function. If there is any problem in loading a file then, the include() function generates a warning but the script will continue execution. Assume you want to create a common menu for your website. Then create a file menu.php with the following content.

```
<a href="http://www.tutorialspoint.com/index.htm">Home</a> -  
<a href="http://www.tutorialspoint.com/ebxml">ebXML</a> -  
<a href="http://www.tutorialspoint.com/ajax">AJAX</a> -  
<a href="http://www.tutorialspoint.com/perl">PERL</a> <br />
```

Now create as many pages as you like and include this file to create header. For example now your test.php file can have following content.

```
<html>  
  <body>  
    <?php include("menu.php"); ?>  
    <p>This is an example to show how to include PHP file!</p>  
  </body>  
</html>
```

It will produce the following result

Include

### 8.6.2 The require() Function

The require() function takes all the text in a specified file and copies it into the file that uses the include function. If there is any problem in loading a file then the require() function generates a fatal error and halt the execution of the script. So there is no difference in require() and include() except they handle error conditions. It is recommended to use the require() function instead of include(), because scripts should not continue executing if files are missing or misnamed.

You can try using above example with require() function and it will generate same result. But if you will try following two examples where file does not exist then you will get different results.

```
<html>
  <body>
    <?php include("xxmenu.php"); ?>
    <p>This is an example to show how to include wrong PHP file!</p>
  </body>
</html>
```

This will produce the following result –

This is an example to show how to include wrong PHP file!

Now lets try same example with require() function.

```
<html>
  <body>
    <?php require("xxmenu.php"); ?>
    <p>This is an example to show how to include wrong PHP file!</p>
  </body>
</html>
```

This time file execution halts and nothing is displayed.

### **Check your Progress**

1. What are the Concepts of PHP?
2. What are Arrays?
3. What is Multidimensional Array?
4. What are Constructor Functions?
5. What are Function Overriding?

---

## **8.7 ANSWERS TO CHECK YOUR PROGRESS**

---

1. The Concepts of PHP are:
  - Function
  - Strings
  - Arrays
  - OOP
  - Regular Expression
  - File Handling
2. An array is a data structure that stores one or more similar type of values in a single value. For example if you want to store 100 numbers then instead of defining 100 variables it's easy to define an array of 100 length. is called array index.
3. A multi-dimensional array each element in the main array can also be an array. And each element in the sub-array can be an array, and

so on. Values in the multi-dimensional array are accessed using multiple indexes.

4. Constructor Functions are special type of functions which are called automatically whenever an object is created. So, we take full advantage of this behaviour, by initializing many things through constructor functions. PHP provides a special function called `construct ()` to define a constructor.
5. Function definitions in child classes override definitions with the same name in parent classes. In a child class, we can modify the definition of a function inherited from parent class.

---

## 8.8 SUMMARY

---

- A multi-dimensional array each element in the main array can also be an array.
- Object is an individual instance of the data structure defined by a class. You define a class once and then make many objects that belong to it. Objects are also known as instance.
- Polymorphism is an object oriented concept where same function can be used for different purposes. For example function name will remain same but it takes different number of arguments and can do different task.
- Overloading is a type of polymorphism in which some or all of operators have different implementations depending on the types of their arguments. Similarly functions can also be overloaded with different implementation.

---

## 8.9 KEYWORDS

---

**Function:** A function is a piece of code which takes one more input in the form of parameter and does some processing and returns a value.

**Associative array:** An array with strings as index. This stores element values in association with key values rather than in a strict linear index order.

**Multidimensional array:** An array containing one or more arrays and values are accessed using multiple indices

**Inheritance:** When a class is defined by inheriting existing function of a parent class then it is called inheritance. Here child class will inherit all or few member functions and variables of a parent class.

**Parent class:** A class that is inherited from by another class. This is also called a base class or super class.

---

## 8.10 SELF ASSESSMENT QUESTIONS AND EXERCISES

---

### Short Answer questions:

1. What are Interfaces?
2. What is String Manipulation?
3. Explain about Regular Expression?
4. What is the `include ()` function?
5. What is the `require ()` function?

**Long Answer questions:**

1. Explain briefly about Object Oriented Concepts?
2. Explain about Arrays and its Types?
3. Explain about Functions?

---

**8.11. FURTHER READINGS**

---

Rémy Card, Eric Dumas, and Franck Mével. The Linux kernel book. John Wiley & Sons, Inc., 2003.

Steve Suchring. MySQL BBible. John Wiley, 2002.

Rasmus Lerdorf and Levin Tatroe. Programming PHP. " O'Reilly Media, Inc., 2002.

Wesley J. Chun. Core Python Programming. Prentice Hall, 2001.

Martin C. Brown. Perl: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

Steven Holzner. PHP: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

Vikram Vaswani. MySQL: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

---

# UNIT 9

## PHP AND MySQL DATABASE

---

### Structure

- 9.0 Introduction
- 9.1 Objective
- 9.2 PHP and MySQL
- 9.3 PHP and LDAP
- 9.4 PHP Connectivity
- 9.5 Sending and Receiving E-mails
- 9.6 Debugging and Error Handling
- 9.7 Templates
- 9.8 Answers to Check Your Progress
- 9.9 Summary
- 9.10 Keywords
- 9.11 Self Assessment Questions and Exercises
- 9.12 Further Readings

---

### 9.0 INTRODUCTION

---

The open source software PHP programming is combined with other backend open source available software's for creation of real time applications. This unit covers the advance concepts of PHP for better programming skills in connection of PHP with other languages.

---

### 9.1 OBJECTIVE

---

This unit helps you to

- Understand PHP and SQL
- Combine PHP with PDAP
- PHP connectivity and error handling with debugging

---

### 9.2 PHP AND MYSQL

---

MySQL easily fits into server-side programming languages, through a domain. Before your MySQL functions will be recognizable, make sure to enable MySQL in your php.ini file. You can use MySQL commands within PHP code almost as seamlessly as you do with HTML. Numerous PHP functions work specifically with MySQL to make your life easier.

Some of the more commonly used functions are:

- `mysql_connect([$host[, $username[, $password]])`: Connects to the MySQL server and returns a resource which is used to reference the connection.



- `mysql_select_db($database[, $resource])`: Equivalent to the MySQL command USE and sets the active database.
- `mysql_query($query[, $resource])`: Used to send any MySQL command to the database server. In the case of SELECT queries, a reference to the result set will be returned.
- `mysql_fetch_array($result)`: Return a row of data from the query's result set as an associative array, numeric array or both.
  - `mysql_fetch_assoc($result)`: Return a row of data from the query's result set as an associative array.
- `mysql_error([$resource])`: Shows the error message generated by the previous query.

## NOTES

### 9.2.1 Connecting to the MySQL Server

Before you can do anything with MySQL, you must first connect to the MySQL server using your specific connection values. Connection variables consist of the following parameters: Hostname: In our case, this is localhost because everything has been installed locally. You will need to change this to whatever host is acting as your MySQL server, if MySQL is not on the same server.

Username and password: This is to authenticate securely over server end. You issue this connection command with the PHP function called `mysql_connect()`. As with all of your PHP/MySQL statements, you can either put the information into variables or leave it as text in your MySQL query.

Here's how you would do it with variables:

```
$host = 'localhost';
```

```
$user = 'user_name';
```

```
$pass = 'password';
```

```
$db = mysql_connect($host, $user, $pass);
```

The following statement has the same effect:

```
$db = mysql_connect('localhost', 'user_name', 'password');
```

For the most part, your specific needs and the way you are designing your table will dictate what piece of code you use. Most people use the first method for security 'sake and put them variables in a different file. Then they include them wherever they need to make a connection to the database.

## 9.2.2 Web Interface with Apache, MySQL, and PHP

phpMyAdmin is a tool written in PHP intended to handle the administration of MySQL over the Web. Currently it can create and drop databases, create/drop/alter tables, delete/edit/add fields, execute any SQL statement, manage keys on fields. Features provided by the program include:

1. Web interface
2. MySQL database management
3. Import data from CSV and SQL
4. Export data to various formats: CSV, SQL, XML, PDF (via the TCPDF library), ISO/IEC 26300 - OpenDocument Text and Spreadsheet, Word, Excel, LaTeX and others PHP in the Web Environment
5. Administering multiple servers
6. Creating PDF graphics of the database layout
7. Creating complex queries using Query-by-example (QBE)
8. Searching globally in a database or a subset of it
9. Transforming stored data into any format using a set of predefined functions, like displaying BLOB-data as image or download-link
10. Active query monitor (Processes)

### AMP Bundles

There are many AMP bundles such as LAMP, XAMP, PHPTriad, WAMP, MAMP, FoxServ, Etc. consists of AMP build in itself. So, there is no need to configure AMPs manually. Some of the AMPs which are available for various operating systems are LAMP (for Linux); WAMP (for Windows); MAMP (for Macintosh); SAMP (for Solaris); and FAMP (for FreeBSD)

---

## 9.3 PHP AND LDAP

---

LDAP is the Lightweight Directory Access Protocol, and is a protocol used to access "Directory Servers". The Directory is a special kind of database that holds information in a tree structure. The concept is similar to your hard disk directory structure, except that in this context, the root directory is "The world" and the first level subdirectories are "countries". Lower levels of the directory structure contain entries for companies, organisations or places, while yet lower still we find directory entries for people, and perhaps equipment or documents. To refer to a file in a subdirectory on your hard disk, you might use something like:

```
/usr/local/myapp/docs
```

The forward slash marks each division in the reference, and the sequence is read from left to right. The equivalent to the fully qualified file reference

in LDAP is the "distinguished name", referred to simply as "dn". An example dn might be:

```
cn=John Smith,ou=Accounts,o=My Company,c=US
```

The comma marks each division in the reference, and the sequence is read from right to left. You would read this dn as:

```
country = US
```

```
organization = My Company
```

```
organizationalUnit = Accounts
```

```
commonName = John Smith
```

In the same way as there are no hard rules about how you organise the directory structure of a hard disk, a directory server manager can set up any structure that is meaningful for the purpose. However, there are some conventions that are used. The message is that you cannot write code to access a directory server unless you know something about its structure, any more than you can use a database without some knowledge of what is available.

---

## 9.4 PHP CONNECTIVITY

---

In order to develop and run PHP Web pages three vital components need to be installed on our computer system. Web Server – PHP will work with virtually all Web Server software, including Microsoft's Internet Information Server (IIS) but then most often used is freely available Apache Server. Download Apache for free here <https://httpd.apache.org/download.cgi>

**Database** – PHP will work with virtually all database software, including Oracle and Sybase but most commonly used is freely available MySQL database. Download MySQL for free here – <https://www.mysql.com/downloads/>.

**PHP Parser** – In order to process PHP script instructions a parser must be installed to generate HTML output that can be sent to the Web Browser. This tutorial will guide you how to install PHP parser on your computer.

**PHP Parser Installation**-Before you precede it is important to make sure that you have proper environment setup on your machine to develop your web programs using PHP.

Type the following address into your browser's address box.

```
http://127.0.0.1/info.php
```

If this displays a page showing your PHP installation related information then it means you have PHP and Webserver installed properly. Otherwise you have to follow given procedure to install PHP on your computer. This section will guide you to install and configure PHP over the following four platforms

## Apache Configuration

If you are using Apache as a Web Server then this section will guide you to edit Apache Configuration Files. Just Check it here – PHP Configuration in Apache Server

### NOTES

#### PHP.INI File Configuration

The PHP configuration file, php.ini, is the final and most immediate way to affect PHP's functionality.

---

### 9.5 SENDING AND RECEIVING E-MAILS

---

Windows users should ensure that two directives are supplied. The first is called SMTP that defines your email server address. The second is called send mail from which defines your own email address. The configuration for Windows should look something like this –

```
[mail function]
```

```
; For Win32 only.
```

```
SMTP = smtp.secureserver.net
```

```
; For win32 only
```

```
sendmail_from = webmaster@tutorialspoint.com
```

Linux users simply need to let PHP know the location of their send mail application. The path and any desired switches should be specified to the send mail\_path directive. The configuration for Linux should look something like this –

```
[mail function]
```

```
; For Win32 only.
```

```
SMTP =
```

```
; For win32 only
```

```
sendmail_from =
```

```
; For Unix only
```

```
sendmail_path = /usr/sbin/sendmail -t -i
```

Now you are ready to go –

#### Sending plain text email

PHP makes use of mail() function to send an email. This function requires three mandatory arguments that specify the recipient's email address, the subject of the message and the actual message additionally there are other two optional parameters.

```
mail( to, subject, message, headers, parameters );
```

Here is the description for each parameters.

Sr.No Parameter & Description

- 1 to Required. Specifies the receiver / receivers of the email
- 2 subject Required. Specifies the subject of the email. This parameter cannot contain any newline characters
- 3 message Required. Defines the message to be sent. Each line should be separated with a LF (\n). Lines should not exceed 70 characters
- 4 headers Optional. Specifies additional headers, like From, Cc, and Bcc. The additional headers should be separated with a CRLF (\r\n)
- 5 parameters Optional. Specifies an additional parameter to the send mail program

As soon as the mail function is called PHP will attempt to send the email then it will return true if successful or false if it is failed. Multiple recipients can be specified as the first argument to the mail() function in a comma separated list.

### 9.5.1 Sending HTML email

When you send a text message using PHP then all the content will be treated as simple text. Even if you will include HTML tags in a text message, it will be displayed as simple text and HTML tags will not be formatted according to HTML syntax. But PHP provides option to send an HTML message as actual HTML message. While sending an email message you can specify a Mime version, content type and character set to send an HTML email.

#### Example

Following example will send an HTML email message to xyz@somedomain.com copying it to afgh@somedomain.com. You can code this program in such a way that it should receive all content from the user and then it should send an email.

```
<html>
  <head>
    <title>Sending HTML email using PHP</title>
  </head>
  <body>
    <?php
      $to = "xyz@somedomain.com";
      $subject = "This is subject";

      $message = "<b>This is HTML message.</b>";
```

#### NOTES

*PHP and MySQL  
Database*

Self-Instructional  
Material

```

$message .= "<h1>This is headline.</h1>";
$header = "From:abc@somedomain.com \r\n";
$header .= "Cc:afgh@somedomain.com \r\n";
$header .= "MIME-Version: 1.0\r\n";
$header .= "Content-type: text/html\r\n";
$retval = mail ($to,$subject,$message,$header);
if( $retval == true ) {
    echo "Message sent successfully...";
}else {
    echo "Message could not be sent...";
}
?>
/body>
</html>

```

### 9.5.2 Sending Attachments with Email

To send an email with mixed content requires to set Content-type header to multipart/mixed. Then text and attachment sections can be specified within boundaries. A boundary is started with two hyphens followed by a unique number which can not appear in the message part of the email. A PHP function md5() is used to create a 32 digit hexadecimal number to create unique number. A final boundary denoting the email's final section must also end with two hyphens.

```

<?php
// request variables // important
$from = $_REQUEST["from"];
$emaila = $_REQUEST["emaila"];
$filea = $_REQUEST["filea"];
if ($filea) {
    function mail_attachment ($from , $to, $subject, $message,
    $attachment){
        $fileatt = $attachment; // Path to the file
        $fileatt_type = "application/octet-stream"; // File Type
        $start = strpos($attachment, '/') == -1 ?
        strpos($attachment, '/') : strpos($attachment, '/')+1;
        $fileatt_name = substr($attachment, $start,
        strlen($attachment)); // Filename that will be used for the

```

```

file as the attachment

Semail_from = $from; // Who the email is from
$subject = "New Attachment Message";
Semail_subject = $subject; // The Subject of the email
Semail_txt = $message; // Message that the email has in it
Semail_to = $to; // Who the email is to
$headers = "From: ".$semail_from;
$file = fopen($fileatt,'rb');
$data = fread($file,filesize($fileatt));
fclose($file);

$msg_txt="\n\n You have recieved a new attachment message from
$from";

$semi_rand = md5(time());
$mime_boundary = "=="Multipart_Boundary_x{$semi_rand}x";

$headers .= "\nMIME-Version: 1.0\n" . "Content-Type:
multipart/mixed;\n" . "
boundary=\"{$mime_boundary}\"";
$semail_txt .= $msg_txt;

$semail_message .= "This is a multi-part message in MIME
format.\n\n" .

"--{$mime_boundary}\n" . "Content-Type:text/html;
charset = \"iso-8859-1\"\n" . "Content-Transfer-Encoding: 7bit\n\n"

.

$semail_txt . "\n\n";

$data = chunk_split(base64_encode($data));
$semail_message .= "--{$mime_boundary}\n" . "Content-Type:
{$fileatt_type};\n" .

" name = \"{$fileatt_name}\".\n" . //"Content-Disposition:
attachment;\n" .

//" filename = \"{$fileatt_name}\".\n" . "Content-Transfer-
Encoding:

base64\n\n" . $data . "\n\n" . "--{$mime_boundary}--\n";
$ok = mail($semail_to, $semail_subject, $semail_message,
$headers);

if($ok) {
echo "File Sent Successfully.";
unlink($attachment); // delete a file after attachment sent.

```

```

    }else {
        die("Sorry but the email could not be sent. Please go back and try
again!");
    }
}

move_uploaded_file($_FILES["filea"]["tmp_name"],
'temp/'.basename($_FILES['filea']['name']));

mail_attachment("$from", "youremailaddress@gmail.com",
'subject', "message", ("temp/" . $_FILES["filea"]["name"]));
}
?>
<html>
<head>
<script language = "javascript" type = "text/javascript">
function CheckData45() {
with(document.filepost) {
if(filea.value != "") {
document.getElementById('one').innerText =
'Attaching File ... Please Wait';
}
}
}
</script>
</head>
<body>
<table width = "100%" height = "100%" border = "0"
cellpadding = "0" cellspacing = "0">
<tr>
<td align = "center">
<form name = "filepost" method = "post"
action = "file.php" enctype = "multipart/form-data" id = "file">
<table width = "300" border = "0" cellspacing = "0"
cellpadding = "0">
<tr valign = "bottom">
<td height = "20">Your Name:</td>

```



```

</tr>
<tr>
  <td><input name = "from" type = "text"
    id = "from" size = "30"></td>
</tr>
<tr valign = "bottom">
  <td height = "20">Your Email Address:</td>
</tr>
<tr>
  <td class = "frmtxt2"><input name = "emaila"
    type = "text" id = "emaila" size = "30"></td>
</tr>
<tr>
  <td height = "20" valign = "bottom">Attach File:</td>
</tr>
<tr valign = "bottom">
  <td valign = "bottom"><input name = "filea"
    type = "file" id = "filea" size = "16"></td>
</tr>
<tr>
  <td height = "40" valign = "middle"><input
    name="Reset2" type="reset" id ="Reset2" value="Reset">
    <input name = "Submit2" type = "submit"
      value = "Submit" onClick = "return
CheckData45()"></td>
</tr>
</table>
</form>
<center>
<table width = "400">
  <tr>
    <td id = "one">
    </td>
  </tr>
</table>
</center>

```

```
</td>
</tr>
</table>
</body>
</html>
```

---

## 9.6 DEBUGGING AND ERROR HANDLING

---

Programs rarely work correctly the first time. Many things can go wrong in your programs that cause the PHP interpreter to generate an error message. You have a choice about where those error messages go. The messages can be sent along with other program output to the web browser. They can also be included in the web server error log. To make error messages display in the browser, set the display errors configuration directive to on. To send errors to the web server error log, set log errors to on. You can set them both to on if you want error messages in both places.

PHP defines some constants you can use to set the value of error\_reporting such that only errors of certain types get reported: E\_ALL (for all errors except strict notices), E\_PARSE (parse errors), E\_ERROR (fatal errors), E\_WARNING (warnings), E\_NOTICE (notices), and E\_STRICT (strict notices). While writing your PHP program, it is a good idea to use PHP-aware editors like BBEdit or Emacs. One of the special special features of these editors is syntax highlighting. It changes the color of different parts of your program based on what those parts are. For example, strings are pink, keywords such as if and while are blue, comments are grey, and variables are black.

Another feature is quote and bracket matching, which helps to make sure that your quotes and brackets are balanced. When you type a closing delimiter such as }, the editor highlights the opening { that it matches. There are following points which need to be verified while debugging your program.

**Missing Semicolons** – Every PHP statement ends with a semicolon (;). PHP doesn't stop reading a statement until it reaches a semicolon. If you leave out the semicolon at the end of a line, PHP continues reading the statement on the following line.

**Not Enough Equal Signs** – When you ask whether two values are equal in a comparison statement, you need two equal signs (==). Using one equal sign is a common mistake.

**Misspelled Variable Names** – If you misspelled a variable then PHP understands it as a new variable. Remember: To PHP, \$test is not the same variable as \$Test.

**Missing Dollar Signs** – A missing dollar sign in a variable name is really hard to see, but at least it usually results in an error message so that you know where to look for the problem.

**Troubling Quotes** – You can have too many, too few, or the wrong kind of quotes. So check for a balanced number of quotes.

**Missing Parentheses and curly brackets** – They should always be in pairs.

**Array Index** – All the arrays should start from zero instead of 1.

Moreover, handle all the errors properly and direct all trace messages into system log file so that if any problem happens then it will be logged into system log file and you will be able to debug that problem.

Error handling is the process of catching errors raised by your program and then taking appropriate action. If you would handle errors properly then it may lead to many unforeseen consequences. It's very simple in PHP to handle an errors.

### **Using die() function**

While writing your PHP program you should check all possible error condition before going ahead and take appropriate action when required. Try following example without having /tmp/test.txt file and with this file.

```
<?php
    if(!file_exists("/tmp/test.txt")) {
        die("File not found");
    }else {
        $file = fopen("/tmp/test.txt","r");
        print "Opend file sucessfully";
    }

    // Test of the code here.

?>
```

This way you can write an efficient code. Using above technique you can stop your program whenever it errors out and display more meaningful and user friendly message.

### **9.6.1 Defining Custom Error Handling Function**

You can write your own function to handling any error. PHP provides you a framework to define error handling function. This function must be able to handle a minimum of two parameters (error level and error message) but can accept up to five parameters (optionally: file, line-number, and the error context)

## Syntax

```
error_function(error_level,error_message,  
error_file,error_line,error_context);
```

Sr.No	Parameter & Description
1	error_level Required - Specifies the error report level for the user defined error. Must be a value number
2	error_message Required - Specifies the error message for the user-defined error
3	error_file Optional - Specifies the file name in which the error occurred
4	error_line Optional - Specifies the line number in which the error occurred
5	error_context Optional - Specifies an array containing every variable and their values in use when the error occurred

## 9.6.2 Possible Error levels

These error report levels are the different types of error the user-defined error handler can be used for. These values can be used in combination using | operator

Sr.No	Constant & Description
1	E_ERROR Fatal run-time errors. Execution of the script is halted
2	E_WARNING Non-fatal run-time errors. Execution of the script is not halted
3	E_PARSE Compile-time parse errors. Parse errors should only be generated by the parser
4	E_NOTICE Run-time notices. The script found something that might be an error, but could also happen when running a script normally
5	E_CORE_ERROR Fatal errors that occur during PHP's initial start-up.

```
int error_reporting ( [int $level] )
```

Following is the way you can create one error handling function:

```
<?php  
  
function handleError($errno, $errstr,$error_file,$error_line) {  
  
    echo "<b>Error:</b> [$errno] $errstr - $error_file:$error_line";
```

```

    echo "<br />";

    echo "Terminating PHP Script";

    die();

}

?>

```

### 9.6.3 Exceptions Handling

PHP 5 has an exception model similar to that of other programming languages. Exceptions are important and provide a better control over error handling. Let's explain their new keyword related to exceptions.

**Try** – A function using an exception should be in a "try" block. If the exception does not trigger, the code will continue as normal. However if the exception triggers, an exception is "thrown".

**Throw** – This is how you trigger an exception. Each "throw" must have at least one "catch".

**Catch** – A "catch" block retrieves an exception and creates an object containing the exception information.

When an exception is thrown, code following the statement will not be executed, and PHP will attempt to find the first matching catch block. If an exception is not caught, a PHP Fatal Error will be issued with an "Uncaught Exception. Exception can be thrown, and caught ("caught") within PHP. Code may be surrounded in a try block. Each try must have at least one corresponding catch block. Multiple catch blocks can be used to catch different classes of exceptions. Exceptions can be thrown (or re-thrown) within a catch block.

Example

Following is the piece of code, copy and paste this code into a file and verify the result.

```

<?php

try {

    $error = 'Always throw this error';

    throw new Exception($error);

    // Code following an exception is not executed.

    echo 'Never executed';
}

```

*PHP and MySQL  
Database*

```

}catch (Exception $e) {
    echo 'Caught exception: ', $e->getMessage(), "\n";
}

// Continue execution

echo 'Hello World';

?>

```

In the above example `$e->getMessage` function is used to get error message. There are following functions which can be used from Exception class.

`getMessage()` – message of exception

`getCode()` – code of exception

`getFile()` – source filename

`getLine()` – source line

`getTrace()` – n array of the backtrace()

`getTraceAsString()` – formatted string of trace

---

## 9.7 TEMPLATES

---

Many programming instructors emphasize separating content, layout, and data. However, the way PHP is often written, it combines all three elements. As a response, web developers often use some form of templating system to try to separate the content from the view. The simplest form of a template is something like the following code:

```

<?php include_once("vars.php") ?>

<!doctype html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title><?=$title?></title>

</head>

<body>

    <h1><?=$heading?></h1>

    <div id = "main">

        <?=$content?>

```

```
</div>
<div id = "footer">
  <?=$footer?>
</div>
</body>
</html>
```

The page holds the structure, but none of the actual contents. The contents are stored in PHP variables, which are stored in another file, called (in this example) vars.php. Here's what vars.php might look like:

```
<?php
$title = "template demo";
$heading = "Template Demo";
$content = <<<HERE
```

This is a very simple template example, which allows the user to load content as variables from an external file. Of course, template systems get much more complex than this.

```
HERE;
```

```
$footer = "from HTML5 / CSS All in One for Dummies";
?>
```

In this extremely simple example, the second PHP file simply defines variables containing the various values, achieving separation of data from view. Often, the secondary PHP file is more sophisticated, grabbing contents from a database or other storage medium

### Check your Progress

1. What is PHP?
2. What is LDAP?
3. What are Missing Semicolons?
4. What are Missing Dollar Signs?
5. What are templates?

---

## 9.8 ANSWERS TO CHECK YOUR PROGRESS

---

1. Python is the open source programming language which is very easy and effective among programmers. This unit helps the user to start from the basic of python in defining their syntax, control and

## NOTES

*PHP and MySQL  
Database*

Self-Instructional  
Material

loops which makes the beginners to learn and understand more easily.

2. There are three numeric types in Python:
  - int
  - float
  - complex
3. Immutable Objects are of in-built types like int, float, bool, string, unicode, tuple. In simple words, an immutable object can't be changed after it is created.
4. A tuple is a sequence of immutable Python objects. Tuples are sequences, just like lists. The differences between tuples and lists are, the tuples cannot be changed unlike lists and tuples use parentheses, whereas lists use square brackets.
5. Python conditional statements are also known as selection statements. Among multiple options only one statement is selected and executed is selection statements. Conditional statements in python are if statement, if-else statements and if elif in python.

---

## 9.9 SUMMARY

---

- The open source software PHP programming is combined with other backend open source available software's for creation of real time applications.
- phpMyAdmin is a tool written in PHP intended to handle the administration of MySQL over the Web.
- LDAP is the Lightweight Directory Access Protocol, and is a protocol used to access "Directory Servers".
- In order to develop and run PHP Web pages three vital components need to be installed on our computer system.
- Windows users should ensure that two directives are supplied. The first is called SMTP that defines your email server address. The second is called send mail from which defines your own email address.

---

## 9.10 KEYWORDS

---

**Error handling:** It is the process of catching errors raised by your program and then taking appropriate action.

**Missing Semicolons** – Every PHP statement ends with a semicolon (;). PHP doesn't stop reading a statement until it reaches a semicolon.

**Catch:** A "catch" block retrieves an exception and creates an object containing the exception information.

---

## 9.11 SELF ASSESSMENT QUESTIONS AND EXERCISES

---

**Short Answer questions:**

1. What are Possible Error levels?



2. Define Custom Error Handling Function?
3. Define Debugging and Error Handling?
4. What are AMP Bundles?
5. Explain about combine PHP with PDAP?

**Long Answer questions:**

1. Explain briefly PHP and LDAP?
2. Explain about Sending and Receiving E-Mails?
3. Explain about Debugging and Error Handling?
4. Explain about Exceptions Handling?

---

## **9.12 FURTHER READINGS**

---

Rémy Card, Eric Dumas, and Franck Mével. The Linux kernel book. John Wiley & Sons, Inc., 2003.

Steve Suchring. MySQL BBible. John Wiley, 2002.

Rasmus Lerdorf and Levin Tatroe. Programming PHP. " O'Reilly Media, Inc., 2002.

Wesley J. Chun. Core Python Programming. Prentice Hall, 2001.

Martin C. Brown. Perl: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

Steven Holzner. PHP: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

Vikram Vaswani. MySQL: The complete reference. 2nd Edition, Tata McGraw-Hill Publishing Company Limited, Indian Reprint, 2009.

# Diploma in Computer Applications 517 12

## OPEN SOURCE SOFTWARE

I - Semester



**ALAGAPPA UNIVERSITY**

[Accredited with A+ Grade by NAAC (CGPA:3.64) in the Third Cycle  
and Graded as Category-I University by MHRD-UGC]

**KARAIKUDI – 630 003**



**DIRECTORATE OF DISTANCE EDUCATION**