1. What is PHP? [Google – 2022]
PHP (Hypertext Preprocessor) is an open-source, server-side scripting language used mainly for web development. It can be embedded within HTML and works well with databases like MySQL.

- **Key Features: Fast, cross-platform, loosely typed, supports OOP, integrates with HTML and JS.**

2. Differences between PHP 5 and PHP 7 [Facebook – 2022]

- **Performance: PHP 7 is ~2x faster than PHP 5.**
- **Error Handling: PHP 7 introduces Throwable, Error, and Exception handling.**
- **Type Declarations: Scalar type hints and return type declarations in PHP 7.**
- **64-bit Support: Full 64-bit integer and large file support in PHP 7.**
- **Deprecated Features: Old MySQL extension removed in PHP 7.**

3. Common uses of PHP [LinkedIn – 2016]

- **Dynamic website development.**
- **Server-side form handling.**
- **CRUD operations with MySQL.**
- **CMS creation (WordPress, Drupal).**
- **REST APIs and backend services.**
- **Session and cookie management.**

4. How do you declare a variable in PHP? [Google – 2023]
Variables in PHP start with $.

```
$name = "Subham";

$age = 20;
```

5. Scope of a variable in PHP [Microsoft – 2021]

- **Local: Inside a function.**
- **Global: Declared outside functions, accessible via global keyword.**
- **Static: Retains value across multiple calls.**
- **Function Parameters: Scope is limited to the function.**

6. How do you write comments in PHP? [LinkedIn – 2016]

```
// Single-line comment

# Single-line comment

/* Multi-line

  comment */
```

7. Difference between single-quoted and double-quoted strings [Amazon – 2020]

- **Single quotes ('') → No variable interpolation, faster.**
- **Double quotes ("") → Parses variables and escape sequences.**

```
$name = "Subham";

echo 'Hello $name'; // Output: Hello $name

echo "Hello $name"; // Output: Hello Subham
```

8. Use of echo and print statements [Facebook – 2022]

- **echo: Outputs one or more strings, slightly faster, no return value.**
- **print: Outputs one string, returns 1 (can be used in expressions).**

```
echo "Hello", " World!";

print "Hello World!";
```

9. PHP constants and how to define them [IBM – 2019]
Constants are variables with fixed values that cannot change.

```
define("SITE_NAME", "MyWebsite");

const VERSION = "1.0";
```

10. How do you include files in PHP? [Microsoft – 2014]

- **include "file.php"; → Warning on error, script continues.**
- **require "file.php"; → Fatal error on missing file, script stops.**
- **Variants: include_once, require_once.**

11. Different data types supported by PHP [TechInnovate – 2022]

- **Scalar: Integer, Float, String, Boolean.**
- **Compound: Array, Object.**
- **Special: NULL, Resource.**

12. How do you check the type of a variable in PHP? [Google – 2020]

- **gettype($var) → Returns type as string.**
- **var_dump($var) → Prints type + value.**
- **is_int(), is_string(), is_array(), etc.**

13. Explain type juggling in PHP [Microsoft – 2023]
Type juggling = Automatic conversion between data types.

```
$x = "5" + 10; // "5" converted to int → 15
```

14. Different types of loops in PHP

- **for → Known iteration count.**
- **while → Runs until condition is false.**
- **do-while → Runs at least once.**
- **foreach → Iterates over arrays/objects.**

15. Difference between while and do-while [Google – 2019]

- **while: Condition checked first, may run 0 times.**
- **do-while: Executes once before condition check.**

16. How do you use foreach loop in PHP?

```
$fruits = ["apple", "banana", "cherry"];

foreach ($fruits as $fruit) {

   echo $fruit;

}
```

17. Use of switch statement [IBM, Google, TCS]
Switch is used for multi-branching based on a variable's value.

```
$day = "Mon";

switch($day) {

   case "Mon": echo "Start of week"; break;

   case "Fri": echo "Weekend soon"; break;

   default: echo "Midweek";

}
```

18. Difference between break and continue statements

- **break: Exits the loop/switch immediately.**
- **continue: Skips current iteration, continues next.**

19. How do you define and use arrays in PHP? [TechInnovate – 2024]

```
// Indexed array

$colors = ["red", "blue", "green"];

// Associative array

$ages = ["Subham"=>20, "Ravi"=>22];
```

20. Difference between indexed and associative arrays

- **Indexed: Keys are numeric (0,1,2...).**
- **Associative: Keys are user-defined strings.**

21. How do you define a function in PHP? [TechInnovate – 2024]

```
function greet($name) {
   return "Hello, $name!";
}
echo greet("Subham");
```

22. Purpose of return statement in a function
Returns a value to the calling code, exits function execution.

23. Variable scope within a function
Variables inside a function are local. Globals must be imported using global or $GLOBALS[].

24. How do you pass arguments to a function?

- **By Value (default): Copy of value.**
- **By Reference: Using &.**

```
function add(&$x) { $x++; }
```

25. What are default arguments in a function?
If no value is passed, defaults are used.

```
function greet($name = "Guest") {
   echo "Hello, $name";
}
```

26. What is object-oriented programming in PHP?
OOP is a programming paradigm based on objects containing data (properties) and methods (functions). Supports encapsulation, inheritance, and polymorphism.

27. Concept of classes and objects

- **Class: Blueprint with properties and methods.**
- **Object: Instance of a class.**

```
class Car {
  public $brand;
```

```
  function drive() { echo "Driving"; }

}

$car1 = new Car();
```

28. Define and use constructors and destructors

- **Constructor (__construct): Initializes object when created.**
- **Destructor (__destruct): Executes when object is destroyed.**

```
class Test {

  function __construct() { echo "Object created"; }

  function __destruct() { echo "Object destroyed"; }

}
```

29. What is inheritance and how is it implemented in PHP?
Inheritance allows one class to acquire properties/methods of another.

```
class ParentClass {

   function greet() { echo "Hello"; }

}

class ChildClass extends ParentClass {

   function bye() { echo "Bye"; }

}
```

- **Types: Single, Multilevel, Hierarchical (No multiple inheritance, but interfaces/traits can be used).**

30. Explain the concept of interfaces in PHP.

- **An interface defines a contract (set of methods) that a class must implement.**
- **Interfaces cannot have properties or concrete methods.**
- **A class can implement multiple interfaces (workaround for multiple inheritance).**

```
interface Logger {

    public function log($message);

}

class FileLogger implements Logger {
```

```
    public function log($message) { echo "Log: $message"; }
}
```

31. What are the different error types in PHP?

- **Parse Error (E_PARSE): Syntax errors.**
- **Fatal Error (E_ERROR): Execution stops (e.g., undefined function).**
- **Warning (E_WARNING): Non-fatal, script continues.**
- **Notice (E_NOTICE): Minor issues (e.g., undefined variable).**
- **Deprecated (E_DEPRECATED): Features no longer recommended.**

32. How do you handle errors in PHP using try-catch blocks? [TechInnovate – 2024]

```
try {
  throw new Exception("Something went wrong!");
} catch (Exception $e) {
  echo "Error: " . $e->getMessage();
}
```

- **Used for exception handling. Prevents script crashes.**

33. Explain the use of set_error_handler() function.

- **Customizes error handling by defining a user function.**

```
function customHandler($errno, $errstr) {
    echo "Error [$errno]: $errstr";
}
set_error_handler("customHandler");
```

34. Purpose of trigger_error() function [TechInnovate – 2024]

- **Generates a user-defined error message.**

```
if($age < 18){
  trigger_error("Age must be 18+", E_USER_WARNING);
}
```

35. How do you handle fatal errors in PHP?

- **Fatal errors cannot be caught directly.**

- Use register_shutdown_function() to handle cleanup at script end.

```
register_shutdown_function(function(){
  $error = error_get_last();
  if($error) { echo "Fatal Error: ".$error['message']; }
});
```

36. What are sessions in PHP and how do you start a session? [IBM – 2019]

- Sessions store user-specific data on the server.
- Start with:

```
session_start();
```

37. How do you store and retrieve session variables?

```
$_SESSION["user"] = "Subham";   // store
echo $_SESSION["user"];       // retrieve
```

38. What are cookies and how do you set them in PHP?

- Cookies are small pieces of data stored on the client browser.

```
setcookie("username", "Subham", time()+3600, "/");
```

39. How do you retrieve and delete cookies in PHP?

```
echo $_COOKIE["username"];  // retrieve
setcookie("username", "", time()-3600, "/"); // delete
```

40. Differences between sessions and cookies

- Sessions: Stored on server, secure, expire when browser closes.
- Cookies: Stored on client, less secure, persist beyond sessions.

41. How do you open and close files in PHP?

```
$file = fopen("data.txt", "r");
fclose($file);
```

42. Different modes of opening a file

- "r" → Read
- "w" → Write (truncate if exists)
- "a" → Append

- "x" → Create new, fail if exists
- "r+", "w+", "a+" → Read/Write variants

43. How do you read and write files in PHP?

```
$fh = fopen("test.txt", "w");

fwrite($fh, "Hello PHP!");

fclose($fh);



$fh = fopen("test.txt", "r");

echo fread($fh, filesize("test.txt"));

fclose($fh);
```

44. Use of fopen(), fread(), and fwrite()

- fopen() → Opens file.
- fread() → Reads data.
- fwrite() → Writes data.

45. How do you check if a file exists in PHP?

```
if(file_exists("test.txt")) echo "File exists";
```

46. Common string functions in PHP

- strlen($str) → Length
- strtoupper($str) / strtolower($str)
- strpos($str, "sub") → Find substring
- substr($str, start, length) → Extract substring
- str_replace("old", "new", $str)

47. How do you concatenate strings in PHP?

```
$a = "Hello"; $b = "World";

echo $a . " " . $b;
```

48. Use of strlen() and strpos()

- strlen("Hello") → 5
- strpos("Hello World", "World") → 6

49. How do you replace part of a string using str_replace()?

```
echo str_replace("world", "PHP", "Hello world");
```

// Output: Hello PHP

50. Purpose of substr() function
Extracts a substring.

echo substr("Subham", 0, 3); // Sub

51. Common array functions in PHP

- count($arr) → Length
- array_merge($a, $b) → Merge
- array_push($arr, "x") / array_pop($arr)
- in_array("x", $arr)
- array_search("x", $arr)

52. How do you merge two arrays in PHP?

$a = [1,2]; $b = [3,4];

$c = array_merge($a, $b);

53. Use of array_push() and array_pop()

- array_push($arr, "x") → Add element at end.
- array_pop($arr) → Remove last element.

54. How do you find the length of an array?

echo count($arr);

55. Purpose of array_search() function
Finds the key/index of a value.

echo array_search("apple", ["apple","banana"]); // 0

56. How do you get the current date and time in PHP?

echo date("Y-m-d H:i:s");

57. Use of date() function

- Formats timestamps into human-readable strings.

echo date("l, d M Y"); // Monday, 08 Sep 2025

58. How do you format a date in PHP?

$date = date("d-m-Y H:i:s");

59. Purpose of strtotime() function
Converts a human-readable date string into a Unix timestamp.

```
echo strtotime("next Monday");
```

60. How do you calculate the difference between two dates?

```
$d1 = new DateTime("2025-09-01");

$d2 = new DateTime("2025-09-08");

$diff = $d1->diff($d2);

echo $diff->days; // 7
```

61. Purpose of $_GET and $_POST arrays

- **$_GET → Retrieves data from URL query parameters.**
- **$_POST → Retrieves data from submitted forms (hidden from URL).**

```
$name = $_GET['name'];

$password = $_POST['pass'];
```

Q62: "How do you send an email using PHP?"
Ans: "By using the mail() function → mail(to, subject, message, headers);"

Q63: "Explain the use of header() function."
Ans: "It is used to send raw HTTP headers to the browser (e.g., redirection, content type)."

Q64: "How do you handle file uploads in PHP?"
Ans: "Use an HTML <form enctype='multipart/form-data'> and access files via $_FILES array."

Q65: "What is the purpose of $_SESSION and $_COOKIE arrays?"
Ans: "$_SESSION stores data on server-side; $_COOKIE stores data on client-side."

Q66: "What are some common security threats in PHP?"
Ans: "SQL Injection, XSS, CSRF, Session Hijacking, Remote File Inclusion."

Q67: "How do you prevent SQL injection in PHP?"
Ans: "By using prepared statements (PDO or mysqli) and input sanitization."

Q68: "What is MySQL?"
Ans: "MySQL is an open-source relational database management system (RDBMS)."

Q69: "Explain the differences between MySQL and SQL."
Ans: "SQL is a query language; MySQL is a DBMS that uses SQL for managing databases."

Q70: "What is a database?"
Ans: "A structured collection of data stored electronically for easy access and management."

Q71: "What are tables in a database?"
Ans: "Tables are database objects that store data in rows and columns."

Q72: "What is a primary key?"
Ans: "A unique identifier for each record in a table, cannot be NULL or duplicate."

Q73: "What are the different data types supported by MySQL?"
Ans: "Numeric (INT, FLOAT, DECIMAL), String (CHAR, VARCHAR, TEXT), Date/Time (DATE, DATETIME, TIMESTAMP), Boolean."

Q74: "Explain the difference between CHAR and VARCHAR."
Ans: "CHAR is fixed-length storage; VARCHAR is variable-length storage."

Q75: "What is the TEXT data type used for?"
Ans: "For storing large text data (up to 65,535 characters)."

Q76: "What are the different numeric data types in MySQL?"
Ans: "INT, TINYINT, SMALLINT, MEDIUMINT, BIGINT, FLOAT, DOUBLE, DECIMAL."

Q77: "What is the DATE data type used for?"
Ans: "To store calendar dates (YYYY-MM-DD)."

Q78: "How do you create a database in MySQL?"
Ans: "CREATE DATABASE dbname;"

Q79: "How do you create a table in MySQL?"
Ans: "CREATE TABLE table_name (column1 datatype, column2 datatype, ...);"

Q80: "Explain the use of SELECT statement."
Ans: "It is used to retrieve data from a database → SELECT * FROM table;"

Q81: "How do you insert data into a table?"
Ans: "INSERT INTO table_name (col1, col2) VALUES (val1, val2);"

Q82: "What is the purpose of UPDATE statement?"
Ans: "It modifies existing records → UPDATE table SET col=value WHERE condition;"

Q83: "What are joins in SQL?"
Ans: "Joins are used to combine rows from multiple tables based on related columns."

Q84: "Explain the different types of joins in MySQL."
Ans: "INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL OUTER JOIN, CROSS JOIN."

Q85: "What is the difference between INNER JOIN and OUTER JOIN?"
Ans: "INNER JOIN returns only matching rows; OUTER JOIN returns matching + non-matching rows."

Q86: "How do you perform a LEFT JOIN?"
Ans: "SELECT cols FROM table1 LEFT JOIN table2 ON table1.id=table2.id;"

Q87: "What is a CROSS JOIN?"
Ans: "It returns the Cartesian product of two tables (all combinations of rows)."

Q88: "What is an index in MySQL?"
Ans: "A database object that speeds up retrieval of rows using pointers."

Q89: "What are the different types of indexes?"
Ans: "PRIMARY, UNIQUE, INDEX (non-unique), FULLTEXT."

Q90: "How do you create an index on a table?"
Ans: "CREATE INDEX idx_name ON table(col);"

Q91: "What is the purpose of UNIQUE index?"
Ans: "Ensures all values in the indexed column are unique."

Q92: "How do indexes improve query performance?"
Ans: "They reduce the number of rows scanned by using fast lookups."

Q93: "What are constraints in SQL?"
Ans: "Rules applied on columns to maintain data integrity (e.g., PRIMARY KEY, FOREIGN KEY, NOT NULL, UNIQUE, CHECK)."

Q94: What is a PRIMARY KEY constraint?
Ans: A PRIMARY KEY uniquely identifies each row in a table. It enforces uniqueness and does not allow NULL values.

Q95: Explain the use of FOREIGN KEY constraint.
Ans: A FOREIGN KEY links two tables by referencing the PRIMARY KEY of another table, ensuring referential integrity.

Q96: What is a UNIQUE constraint?
Ans: UNIQUE ensures that all values in a column are different but unlike PRIMARY KEY, it allows one NULL.

Q97: How do you define a CHECK constraint?
Ans: CHECK ensures values in a column meet a specific condition, e.g., CHECK(age >= 18).

Q98: What are stored procedures?
Ans: Stored procedures are precompiled SQL blocks stored in the database that can be executed with parameters for efficiency.

Q99: How do you create a stored procedure in MySQL?
Ans: Use CREATE PROCEDURE proc_name(...) BEGIN SQL statements; END; and call it with CALL proc_name();.

Q100: What is a trigger in MySQL?
Ans: A trigger is an automatic action executed before or after INSERT, UPDATE, or DELETE on a table.

Q101: How do you create a trigger?
Ans: Use CREATE TRIGGER trigger_name BEFORE/AFTER INSERT ON table FOR EACH ROW BEGIN ... END;.

Q102: What are views in MySQL?
Ans: A view is a virtual table created using a SQL query. It simplifies complex queries and provides security by restricting column access.

Q103: What is a transaction in SQL?
Ans: A transaction is a sequence of SQL statements executed as a single unit of work to maintain consistency.

Q104: Explain the use of COMMIT and ROLLBACK.
Ans: COMMIT saves all changes permanently, while ROLLBACK undoes changes made during the transaction.

Q105: What are the ACID properties?
Ans: ACID = Atomicity, Consistency, Isolation, Durability — ensures reliability of transactions in a database.

Q106: How do you start a transaction in MySQL?
Ans: Use START TRANSACTION; or BEGIN; before executing queries, then use COMMIT/ROLLBACK.

Q107: What is the purpose of SAVEPOINT?
Ans: SAVEPOINT allows creating checkpoints in a transaction so you can roll back to a specific point.

Q108: How do you optimize SQL queries?
Ans: Use proper indexes, avoid SELECT *, normalize data, and analyze queries with EXPLAIN.

Q109: What is query caching in MySQL?
Ans: Query caching stores results of frequently executed queries, improving read performance.

Q110: Explain the use of EXPLAIN statement.
Ans: EXPLAIN shows how MySQL executes a query, helping identify performance bottlenecks.

Q111: What are the best practices for indexing?
Ans: Index columns used in WHERE, JOIN, and ORDER BY, but avoid indexing small or frequently updated columns.

Q112: How do you optimize database schema?
Ans: Normalize tables, use proper data types, partition large tables, and avoid redundant fields.

Q113: How do you connect to a MySQL database using PHP?
Ans: Use mysqli_connect("host","user","password","db"); or PDO for secure connections.

Q114: What is mysqli and how is it different from mysql?
Ans: mysqli is an improved extension with support for OOP, prepared statements, and better performance than the old mysql (deprecated).

Q115: How do you execute a SQL query in PHP?
Ans: Use mysqli_query($conn, $query); or with PDO $stmt = $pdo->query($sql);.

Q116: Explain the use of PDO in PHP.
Ans: PDO (PHP Data Objects) is a database abstraction layer that supports multiple databases and provides prepared statements.

Q117: How do you fetch data from a MySQL database in PHP?
Ans: Use mysqli_fetch_assoc() or PDO $stmt->fetch(PDO::FETCH_ASSOC);.

Q118: How do you handle MySQL connection errors in PHP?
Ans: Use mysqli_connect_errno() or exception handling in PDO (try-catch) to detect errors.

Q119: What is the purpose of mysqli_error() function?
Ans: It returns the error message from the last MySQL operation for debugging.

Q120: How do you handle SQL query errors in PHP?
Ans: Use mysqli_error($conn) or PDO exceptions inside try-catch blocks.

Q121: What are the common MySQL error codes?
Ans: Examples: 1049 (Unknown database), 1064 (Syntax error), 1062 (Duplicate entry), 1045 (Access denied).

Q122: How do you log MySQL errors in PHP?
Ans: Use error_log(mysqli_error($conn)); or configure PHP to log errors to a file.

Q123: How do you prevent SQL injection attacks in PHP?
Ans: Always use prepared statements, parameterized queries, and validate user input.

Q124: What are prepared statements and how do you use them in PHP?
Ans: Prepared statements separate SQL logic from data. Example:

$stmt = $conn->prepare("SELECT * FROM users WHERE id=?");

$stmt->bind_param("i",$id);

$stmt->execute();

Q125: How do you secure MySQL user accounts?
Ans: Use strong passwords, least privilege principle, disable root remote login, and apply SSL for connections.

Q126: What is the purpose of mysqli_real_escape_string()?
Ans: It escapes special characters in a string before using it in SQL queries, helping prevent SQL injection.

Q127: How do you handle sensitive data in MySQL?
Ans: Use encryption (AES_ENCRYPT), hashing (e.g., SHA2 for passwords), prepared statements, and access control.

Q128: What is the purpose of LIMIT clause in SQL?
Ans: LIMIT restricts the number of rows returned by a query, useful for pagination.

Q129: How do you sort query results in MySQL?
Ans: Use ORDER BY column ASC/DESC to arrange rows in ascending or descending order.

Q130: What is the use of GROUP BY clause?
Ans: GROUP BY groups rows with the same values, often used with aggregate functions like COUNT or SUM.

Q131: How do you use aggregate functions in MySQL?
Ans: Functions like COUNT(), SUM(), AVG(), MIN(), MAX() summarize data, usually with GROUP BY.

Q132: What is the purpose of HAVING clause?
Ans: HAVING filters grouped results (unlike WHERE, which filters individual rows).

Q133: How do you insert data into a MySQL database using PHP?
Ans: Use INSERT INTO with mysqli_query() or prepared statements:

mysqli_query($conn,"INSERT INTO users(name) VALUES('John')");

Q134: How do you update records in a MySQL table using PHP?
Ans: Use UPDATE table SET col='val' WHERE condition; with mysqli_query() or prepared statements.

Q135: How do you delete records from a MySQL table using PHP?
Ans: Use DELETE FROM table WHERE condition; via mysqli_query() or PDO.

Q136: How do you handle multiple queries in PHP?
Ans: Use mysqli_multi_query($conn, $sql) or execute queries sequentially.

Q137: What are the common MySQL functions in PHP?
Ans: mysqli_connect(), mysqli_query(), mysqli_fetch_assoc(), mysqli_num_rows(), mysqli_close().

Q138: What is ORM (Object-Relational Mapping)?
Ans: ORM maps database tables to classes/objects, letting developers work with objects instead of raw SQL.

Q139: How do you use an ORM in PHP?
Ans: By using libraries like Doctrine or frameworks like Laravel Eloquent, which provide ORM support.

Q140: What are the benefits of using an ORM?
Ans: Faster development, cleaner code, database abstraction, prevention of SQL injection.

Q141: What is database normalization?
Ans: It's the process of organizing data to reduce redundancy and improve data integrity.

Q142: Explain the different normal forms.
Ans:

- **1NF: No repeating groups, atomic values.**
- **2NF: 1NF + no partial dependency.**
- **3NF: 2NF + no transitive dependency.**
- **BCNF: Stronger form of 3NF.**

Q143: How do you analyze slow queries in MySQL?
Ans: Use the slow query log, EXPLAIN, and SHOW PROFILES to identify performance issues.

Q144: What are the best practices for writing efficient SQL queries?
Ans: Use indexes wisely, avoid SELECT *, normalize schema, use LIMIT, and optimize joins.

Q145: How do you use MySQL profiling?
Ans: Enable with SET profiling=1;, then run SHOW PROFILES; to check query execution time.

Q146: What is the purpose of EXPLAIN ANALYZE?
Ans: It shows the execution plan and actual run time of a query, useful for performance tuning.

Q147: How do you use indexing to improve performance?
Ans: Create indexes on frequently searched/joined columns to speed up lookups and queries.

Q148: What are PHP superglobals?
Ans: Superglobals are built-in PHP arrays like $_GET, $_POST, $_SESSION, $_COOKIE, $_SERVER accessible anywhere.

Q149: What is the purpose of the $_GET superglobal?
Ans: $_GET stores data sent via URL query string in an associative array.

Q150: What is the difference between include and require in PHP?
Ans: Both import files, but include gives a warning on failure, while require causes a fatal error.

Q151: How do you connect to a MySQL database in PHP?
Ans: Use mysqli_connect("host","user","pass","db"); or PDO for secure connections.

Q152: What is the isset() function used for in PHP?
Ans: isset() checks if a variable is set and not NULL.

Q153: How do you handle errors in PHP?
Ans: Use try-catch blocks, custom error handlers (set_error_handler()), and error_log().

Q154: What is a PHP namespace?
Ans: Namespaces group classes, functions, and constants to avoid name conflicts in large applications.

Q155: What is the difference between INNER JOIN and LEFT JOIN?
Ans: INNER JOIN returns matching rows from both tables, LEFT JOIN returns all rows from the left table plus matches.

Q156: What is normalization in MySQL?
Ans: It's structuring a database to minimize redundancy and improve consistency (same as DB normalization).

Q157: What are MySQL transactions?
Ans: A transaction is a group of queries executed as a single unit, ensuring ACID properties with COMMIT/ROLLBACK.

Q158: Write a program in PHP to INSERT data in a database file.

```
<?php

$conn = mysqli_connect("localhost","root","","testdb");

$name = "John";

$sql = "INSERT INTO users(name) VALUES('$name')";

if(mysqli_query($conn, $sql)){
```

```
   echo "Inserted Successfully";
}
?>
```

Q159: Write a program in PHP to DISPLAY data from a database file.

```
<?php
$conn = mysqli_connect("localhost","root","","testdb");
$result = mysqli_query($conn,"SELECT * FROM users");
while($row = mysqli_fetch_assoc($result)){
   echo $row['id']." - ".$row['name']."<br>";
}
?>
```

Q160: Write a program in PHP to DELETE data from a database file.

```
<?php
$conn = mysqli_connect("localhost","root","","testdb");
$id = 1;
$sql = "DELETE FROM users WHERE id=$id";
if(mysqli_query($conn,$sql)){
   echo "Record Deleted";
}
?>
```

Q161: Write a program in PHP to UPDATE data from a database file.

```
<?php
$conn = mysqli_connect("localhost","root","","testdb");
$id = 2;
$newName = "David";
$sql = "UPDATE users SET name='$newName' WHERE id=$id";
```

```
if(mysqli_query($conn,$sql)){
   echo "Record Updated";
}
?>
```

Q162: Write a program in PHP to prepare a Registration form using validation.

```
<?php
if($_SERVER["REQUEST_METHOD"]=="POST"){
   $name = trim($_POST["name"]);
   $email = trim($_POST["email"]);
   if(empty($name) || !filter_var($email, FILTER_VALIDATE_EMAIL)){
      echo "Invalid Input";
   } else {
      echo "Registration Successful!";
   }
}
?>
<form method="post">
  Name: <input type="text" name="name"><br>
  Email: <input type="text" name="email"><br>
  <button type="submit">Register</button>
</form>
```

Q163: Write a program in PHP to prepare a Login form using validation.

```
<?php
if($_SERVER["REQUEST_METHOD"]=="POST"){
   $user = $_POST["username"];
   $pass = $_POST["password"];
```

```
    if($user=="admin" && $pass=="1234"){
        echo "Login Successful!";
    } else {
        echo "Invalid Credentials!";
    }
}
?>
<form method="post">
  Username: <input type="text" name="username"><br>
  Password: <input type="password" name="password"><br>
  <button type="submit">Login</button>
</form>
```

Q164: How do you handle user authentication in PHP?
Ans:

- **Use registration + login forms with hashed passwords (password_hash(), password_verify()).**
- **Store user data in sessions ($_SESSION).**
- **Use prepared statements to prevent SQL injection.**
- **Implement logout by destroying session with session_destroy().**