

Architecting Web Applications using PHP

Session 9

Database Management in PHP

Session Overview

In this session, you will be able to:

- Describe PHP support for MySQL
- Identify prerequisites for databases in PHP
- Elaborate how to establish MySQL database connection
- Describe how to create and delete MySQL Database and tables using MySQL in PHP
- Explain data insertion and data retrieval to/from MySQL database
- Explain how to update data into MySQL database
- List and explain the three ways to backup MySQL database
- Explain selecting and filtering data from MySQL database using WHERE, ORDER BY, and LIMIT clauses

PHP and MySQL

PHP Web applications will involve data handling at some point or the other. This data will be stored in databases to persist them.

To store, retrieve, and maintain such data through scripts, PHP supports many database management systems including MySQL, MariaDB, Db2, MongoDB, Oracle, PostgreSQL, and SQLite.

What is MySQL?

- First developed by Oracle, MySQL is a Relational Database Management System (RDBMS) that is a free and open-source software.
- MySQL mostly uses the standard SQL and runs on the server, along with being used in Web application development.
- Being exceptionally fast, easy-to-use, and reliable, it helps organize the data in databases and is an ideal choice for small and large applications. PHP, along with MySQL, is cross-platform.

Prerequisites for Database in PHP

When user is using XAMPP with PHP and MySQL installations, user can configure the `php.ini` as follows:

- Click Config in XAMPP Control Panel and then, select `php.ini` to open it.
- Locate the line `extension=mysqli` and uncomment it.
- Locate for the line `mysqli.default_port` and set it to 3306.
- Save the file.
- Start Apache in XAMPP Control Panel.
- Start MySQL in XAMPP Control Panel.
- Click Admin option in XAMPP Control Panel next to MySQL. This will launch phpMyAdmin, the client through which user can connect to MySQL. A variant of MySQL called MariaDB will be used with XAMPP.
- Verify through phpMyAdmin that MySQL is running.

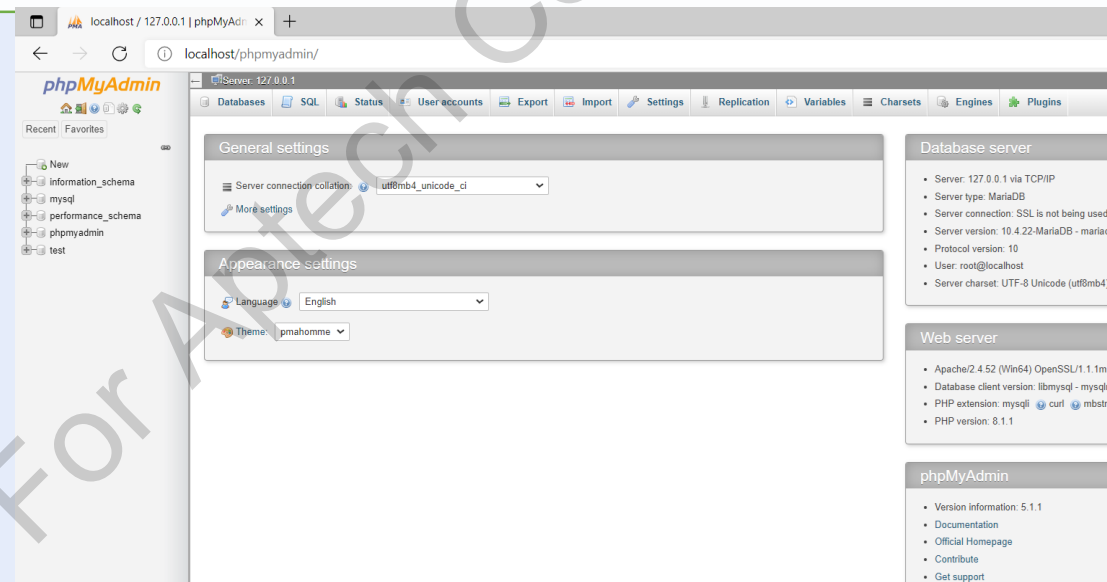
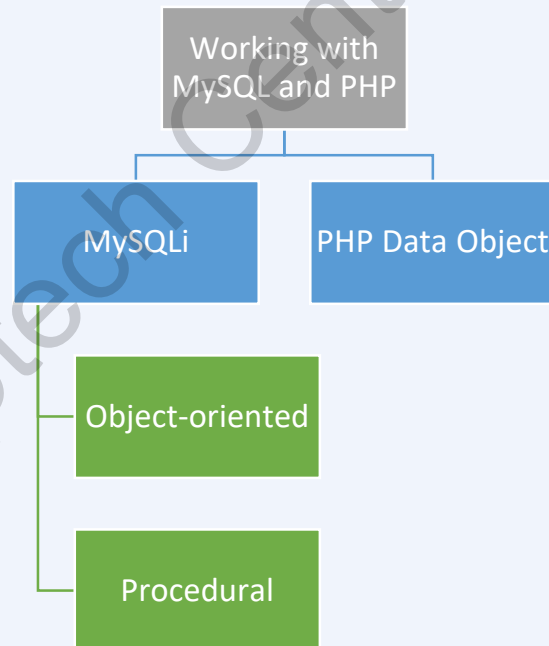


Figure: phpMyAdmin Page

MySQL Database Connection Through PHP Script [1-2]

PHP `mysqli_connect()` : The `PHP mysqli_connect()` function is used to connect with the MySQL database. It can return a resource, depending on whether the connection is null or established successfully.



For Aptech

MySQL Database Connection Through PHP Script [2-2]

Code Snippet:

```
<?php
$host = "localhost:3306";
$username = "root";
$password = "root";
$phpconn = mysqli_connect($host, $username, $password);
if(! $phpconn) {
    die("Could not connect to database: Please verify the
privileges" . mysqli_error());
}
echo "Connection to database is successful";
mysqli_close($phpconn);
?>
```

In the Code Snippet, PHP's built-in function `mysqli_connect` is used to connect with the database. The variables `host`, `username`, and `passwd` are passed.



Figure: Output for Code Snippet

Creating and Deleting a MySQL Database [1-3]

Code Snippet:

```
<?php
$servername = "localhost";
$username = "root";
$password = "root";
// Create a connection
$conn = new mysqli($servername, $username, $password);
// Verify whether connection was successful or not
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
// Create database
$sql = "CREATE DATABASE sampleDB";
if ($conn->query($sql) === TRUE) {
    echo "Task complete. You have successfully created a database";
} else {
    echo "You have an error while creating database: " . $conn->error;
}
$conn->close();
?>
```

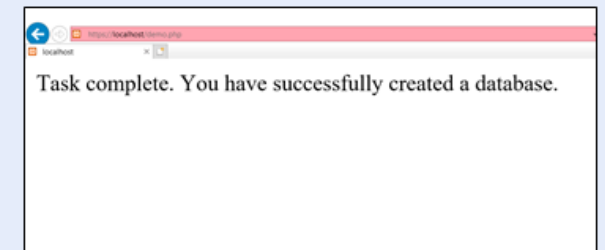


Figure: Output for Code Snippet

The Code Snippet shows an example of creating a database called `sampleDB` while working with MySQLi Object-oriented.

Creating and Deleting a MySQL Database [2-3]

Code Snippet:

```
<?php
$servername = "localhost";
$username = "root";
$password = "root";
// Initially you should create connection
$conn = mysqli_connect($servername, $username, $password);
// Check connection to return the status of connection
if (!$conn) {
    die("Error in Connection: " . mysqli_connect_error());
}
// SQL command is given to create database
$sql = "CREATE DATABASE sample2DB";
if (mysqli_query($conn, $sql)) {
    echo "Task complete. You have successfully created a database";
} else {
    echo "Error while creating database: Please check" .
    mysqli_error($conn);
}
mysqli_close($conn);
?>
```

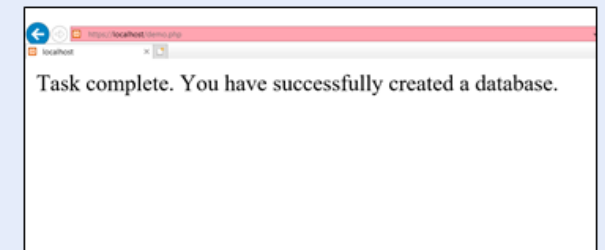


Figure: Output for Code Snippet

Code Snippet shows an example of using MySQLi procedural.

Creating and Deleting a MySQL Database [3-3]

Code Snippet:

```
<?php
$dbhost = "localhost";
$username = "root";
$password = "root";
$conn = mysqli_connect($dbhost, $username, $password);
if(! $conn) {
    die("Could not connect: " . mysqli_error());
}
$sql = "DROP DATABASE sampleDB";
$retval = mysqli_query($conn, $sql);
if(! $retval) {
    die("Due to SQL error, deleting database sampleDB is not possible: "
. mysqli_error());
}
echo " Database deleted successfully here";
mysqli_close($conn);
?>
```



Figure: Output for Code Snippet

Code Snippet shows an example of deleting a database.

Create/Add Tables in MySQL Database [1-2]

NOT NULL

- No null values are allowed as it is necessary for each row to have a value for the column.

DEFAULT VALUE

- The default value can be added at a time when no more values are passed.

UNSIGNED

- Unsigned is used for the number types. It limits the data stored, to zero and positive numbers.

AUTO INCREMENT

- Each time a new record is added, the value of the field automatically increases by one.

PRIMARY KEY

- The main use of the primary key is to figure out rows in the table. The column with the `PRIMARY KEY` setting is often an ID number and is often used with `AUTO_INCREMENT`. Each table should have a primary key column and its value must be unique for each record in the table.

For Apteck Centre Use Only

Create/Add Tables in MySQL Database [2-2]

Code Snippet:

```
<?php
$servername = "localhost";
$username = "root";
$password = "root";
$dbname = "sample2DB";
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Next, you should check the connection
if ($conn->connect_error) {
    die("Connection failed: Recheck the connection details" . $conn->connect_error);
}
// Write an SQL command to create table and store in a variable
$sql = "CREATE TABLE Passengers (
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR(30) NOT NULL,
lastname VARCHAR(30) NOT NULL,
email VARCHAR(50),
reg_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
)";
if ($conn->query($sql) === TRUE) {
    echo "Table Passengers created successfully";
} else {
    echo "Error creating table: " . $conn->error;
}
$conn->close();
?>
```



Figure: Output for Code Snippet

Code Snippet shows how to create a table in PHP (MySQLi Object-oriented).

Inserting Data into MySQL Database [1-2]

Code Snippet:

```
<?php
$dbhost = "localhost";
$username = "root";
$password = "root";
$dbname = "sample2DB";
$conn = mysqli_connect($dbhost, $username, $password);
if(! $conn) {
    die("Issue in the connection. Please check the details again: " .
    mysqli_error());
}
$sql = "INSERT INTO Passengers (id,firstname,
lastname,email,reg_date) ". "VALUES (101,'Derek', 'Houston',
'derek@sample.com', NOW())";
mysqli_select_db($conn,"sample2DB");
$retval = mysqli_query($conn,$sql);
if(! $retval) {
    die("Could not enter data: " . mysqli_error());
}
    echo "Entered data successfully in the table\n";
mysqli_close($conn);
?>
```



Figure: Output for Code Snippet

Inserting Data into MySQL Database [2-2]

Code Snippet:

```
<?php
$servername = "localhost";
$username = "root";
$password = "root";
$dbname = "sample2DB";
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
$sql = "INSERT INTO Passengers (firstname, lastname, email)VALUES ('Hugh', 'Sheperd', 'hugh@sample.com')";
if ($conn->query($sql) === TRUE) {
    echo "Successfully created new record";
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}
$conn->close();
?>
```



Figure: Output for Code Snippet

Code Snippet shows how to add a new record to the table (MySQLi Object-oriented).

Multiple Records Insertion

Code Snippet:

```
<?php
$servername = "localhost";
$username = "root";
$password = "root";
$dbname = "sample2DB";
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
$sql = "INSERT INTO Passengers (firstname, lastname, email) VALUES ('Luke', 'Thompson', 'Luke@sample.com');";
$sql .= "INSERT INTO Passengers (firstname, lastname, email) VALUES ('Ben', 'Smith', 'ben@sample.com');";
$sql .= "INSERT INTO Passengers (firstname, lastname, email) VALUES ('Ruby', 'Stokes', 'ruby@sample.com')";
if ($conn->multi_query($sql) === TRUE) {
    echo "New records created successfully";
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}
$conn->close();
?>
```

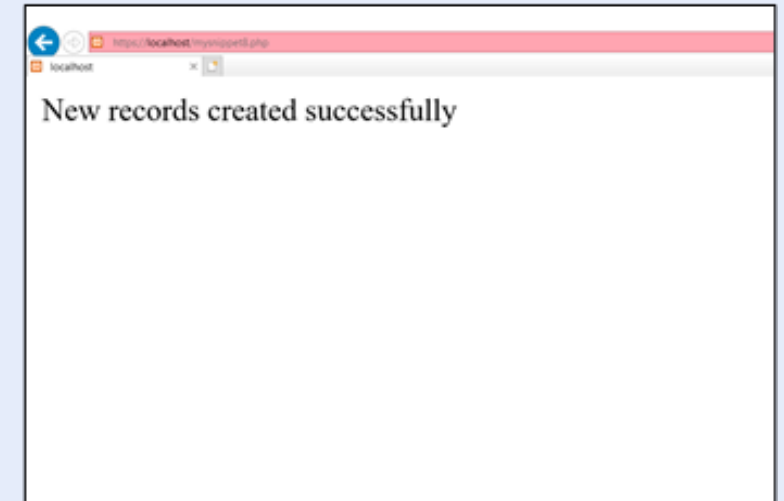


Figure: Output for Code Snippet

Code Snippet shows an example of adding three new records to the Passengers table using MySQLi Object-oriented.

Getting Last Inserted ID

Code Snippet:

```
<?php
$servername = "localhost";
$username = "root";
$password = "root";
$dbname = "sample2DB";
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
$sql = "INSERT INTO Passengers (firstname, lastname, email) VALUES
('Mark', 'Mortan', 'Mark@sample.com')";
if ($conn->query($sql) === TRUE) {
    $last_id = $conn->insert_id;
    echo "Created successfully new record in Table-Passengers. Last
inserted ID is: " . $last_id;
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}
$conn->close();
?>
```

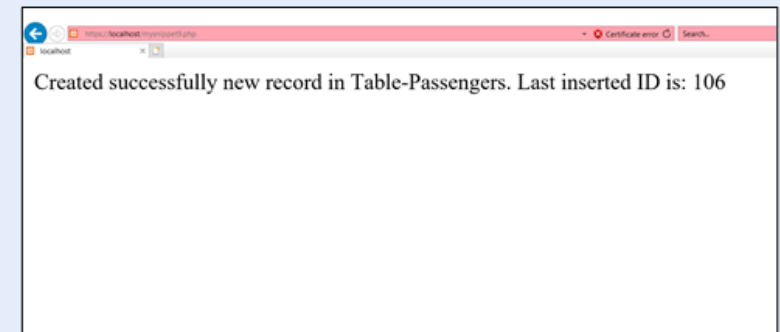


Figure: Output for Code Snippet

Code Snippet shows how to retrieve the ID of the last inserted record (MySQLi Object-oriented).

Retrieving Data from MySQL Database [1-3]

Code Snippet:

```
<?php
$dbhost = "localhost";
$username = "root";
$password = "root";
$dbname = "sample2DB";
$conn = mysqli_connect($dbhost, $username, $password);
if (! $conn) {
    die("Something went wrong!!Could not connect: " .
mysqli_error()); }
$sql = "SELECT id,firstname, lastname,email FROM passengers";
mysqli_select_db($conn,"sample2DB");
$retval = mysqli_query($conn,$sql);
if(! $retval) {
    die("Error encountered while fetching the data: " .
mysqli_error());
}
while($row = mysqli_fetch_array($retval)) {
    echo "Passenger ID. :{$row["id"]} <br> ".
        "First Name : {$row["firstname"]} <br> ".
        "Last Name: {$row["lastname"]} <br> ".
        "Email: {$row["email"]} <br> ".
        "-----<br>";
}
echo "Data successfully retrieved\n";
mysqli_close($conn);
?>
```

Code Snippet shows an example for fetching records from the Passengers table.

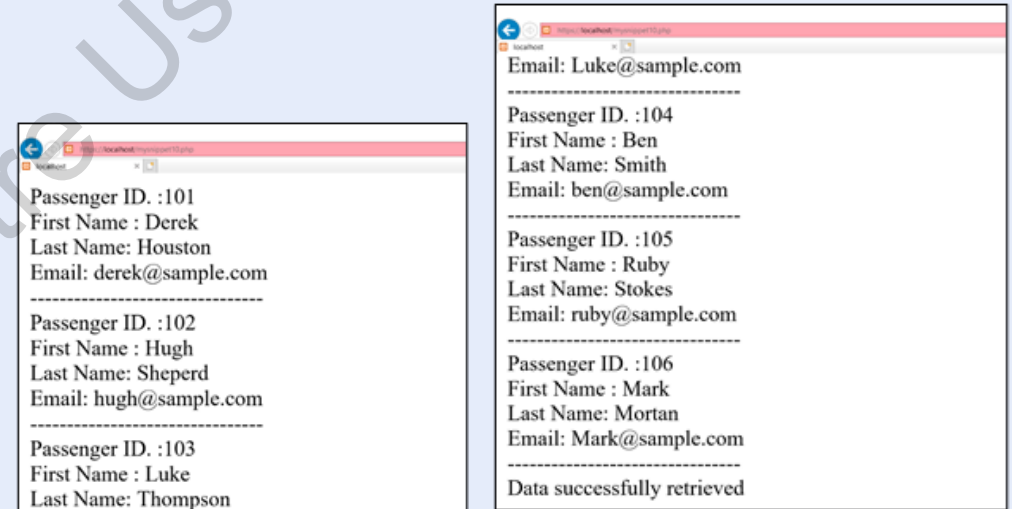


Figure: Output for Code Snippet

Retrieving Data from MySQL Database [2-3]

Code Snippet:

```
<?php
$dbhost = "localhost";
$username = "root";
$password = "root";
$dbname = "sample2DB";
$conn = mysqli_connect($dbhost, $username, $password);
if(! $conn) {
    die("Could not connect: " . mysqli_error());
}
$sql = "SELECT id,firstname, lastname,email FROM passengers";
mysqli_select_db($conn,"sample2DB");
$retval = mysqli_query($conn, $sql);
if(! $retval) {
    die("Could not get data: " . mysqli_error());
}
while($row = mysqli_fetch_assoc($retval)) {
    echo "Passenger ID. :{$row['id']} <br> ".
        "First Name : {$row['firstname']} <br> ".
        "Last Name: {$row['lastname']} <br> ".
        "Email: {$row['email']} <br> ".
        "-----<br>";
}
echo "Data successfully retrieved\n";
mysqli_close($conn);
?>
```

Code Snippet shows example to display all the records from the Passengers table using the `mysqli_fetch_assoc()` function.

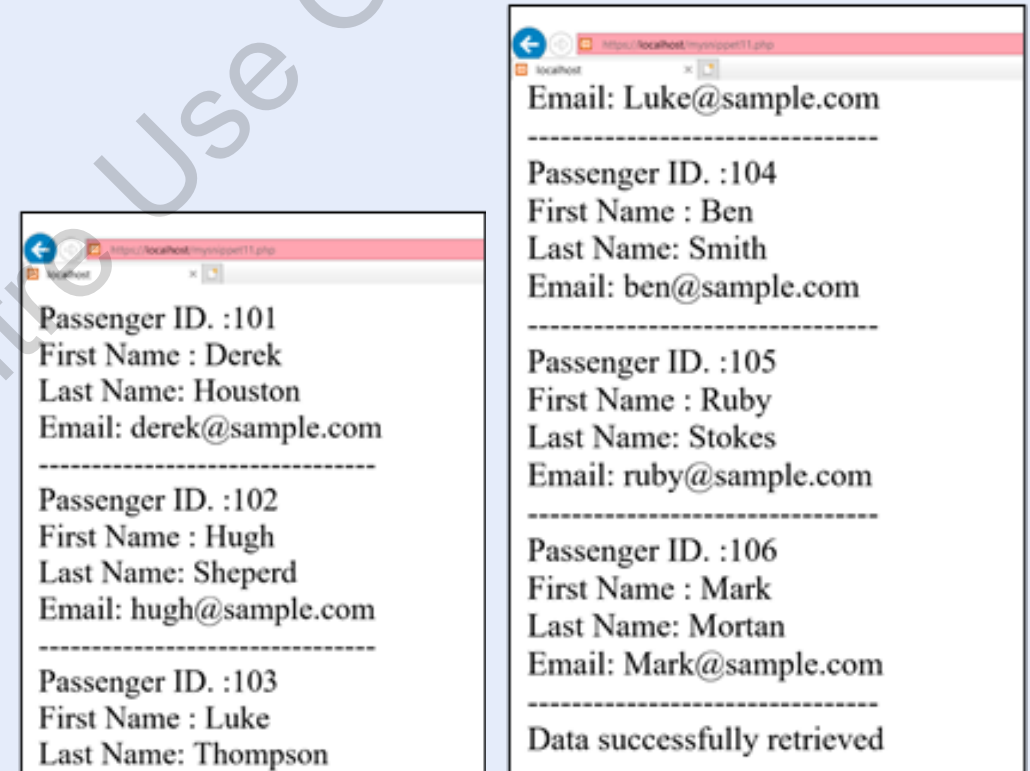


Figure: Output for Code Snippet

Retrieving Data from MySQL Database [3-3]

Code Snippet:

```
<?php
$dbhost = "localhost";
$username = "root";
$password = "root";
$dbname = "sample2DB";
$conn = mysqli_connect($dbhost, $username, $password);
if(! $conn) {
    die("Encountered an error, could not connect: " .
mysqli_error());
}
$sql = "SELECT id,firstname, lastname,email FROM passengers";
mysqli_select_db($conn,"sample2DB");
$retval = mysqli_query($conn, $sql);
if(! $retval) {
    die("Error while fetching data: " . mysqli_error());
}
while($row = mysqli_fetch_array($retval, MYSQLI_NUM)) {
    echo "Passenger ID. :{$row[0]} <br> ".
        "First Name: {$row[1]} <br> ".
        "Last Name: {$row[2]} <br> ".
        "Email: {$row[3]} <br> ".
        "-----<br>";
}
echo "Data successfully fetched\n";
mysqli_close($conn);
?>
```

Code Snippet shows an example where all records from the passengers table are displayed using the MYSQLI_NUM argument.

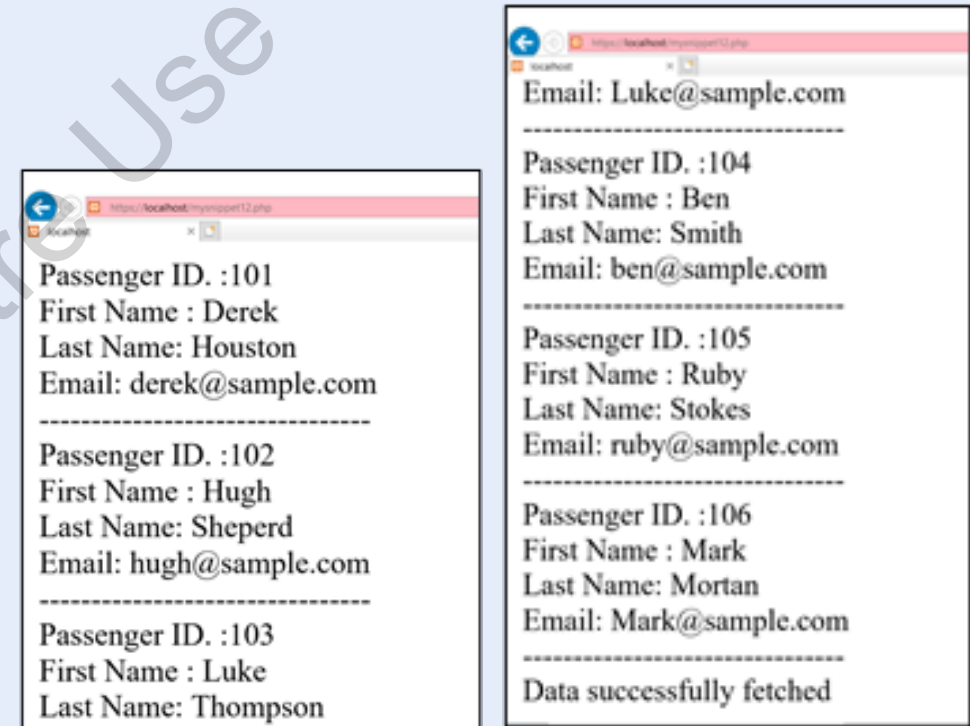


Figure: Output for Code Snippet

Data Selection with MySQLi

Code Snippet:

```
<?php
$servername = "localhost";
$username = "root";
$password = "root";
$dbname = "sample2DB";
// Let us create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection status
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
$sql = "SELECT id, firstname, lastname FROM Passengers";
$result = $conn->query($sql);
//The given code iterates through all the records in table
if ($result->num_rows > 0) {
    echo "<table><tr><th>ID</th><th>Name</th></tr>";
    // output data of each row
    while($row = $result->fetch_assoc()) {
        echo "<tr><td>".$row["id"]."</td><td>".$row["firstname"]."
".$row["lastname"]."</td></tr>";
    }
    echo "</table>";
} else {
    echo "0 results";
}
$conn->close();
?>
```

Code Snippet shows selecting the `firstname`, `lastname`, and `id` columns from the `Passengers` table and showcasing them on to the page.



ID	Name
101	Derek Houston
102	Hugh Sheperd
103	Luke Thompson
104	Ben Smith
105	Ruby Stokes
106	Mark Mortan

Figure: Output for Code Snippet

Updating Data into a MySQL Table

Code Snippet:

```
<html>
<head>
<title>Update a Record in mysql Database</title>
</head>
<body>
<?php
    if(isset($_POST["update"])) {
        $dbhost = "localhost";
        $username = "root";
        $password = "root";
        $conn = mysqli_connect($dbhost, $username, $password);
        if(! $conn) {
            die("Could not connect: " . mysqli_error());
        }
        $pasngr_id = filter_input(INPUT_POST, 'id');
        $pemail = filter_input(INPUT_POST, 'email');
        $sql = "UPDATE passengers SET email =' " . $pemail
            . "' WHERE id = $pasngr_id " ;
        mysqli_select_db($conn,"sample2DB");
        $retval = mysqli_query($conn,$sql);
        if(! $retval) {
            die("Could not update data: " . mysqli_error());
        }
        echo "Updated data successfully\n";
        mysqli_close($conn);
    }
    ?>
    <form method = "post" action = "<?php $_PHP_SELF ?>">
    <table width = "400" border = " 0" cellspacing =
        "1" cellpadding = "2">
        <tr>
            <td width = "100"> Passenger ID:</td>
```

```

            <input name = "passenger_id" type = "text"
                id = "id"></td>
        </tr>
        <tr>
            <td width = "100"> Email:</td>
            <td><input name = "email" type = "text"
                id = "email"></td>
        </tr>
        <tr>
            <td width = "100"> </td>
            <td> </td>
        </tr>
        <tr>
            <td width = "100"> </td>
            <td>
                <input name = "update" type = "submit" id =
                    "update" value = "Update">
            </td>
        </tr>
    </table>
</form>

</body>
</html>
```

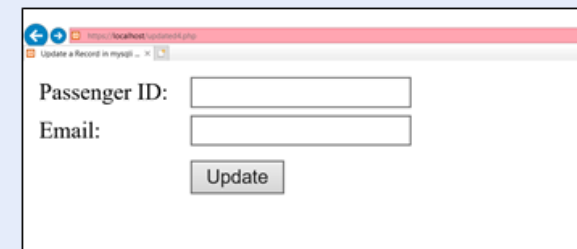


Figure: Output for Code Snippet

Code Snippet shows an example of the update operation.

Using SQL Command Through PHP [1-2]

Code Snippet:

```
<?php
$dbhost = "localhost";
$username = "root";
$password = "root";
$conn = mysqli_connect($dbhost, $username, $password);
if(! $conn) {
    die("Error while connecting" . mysqli_error());
}
$table_name = "passengers";
$backup_file = "E:\\tmp\\passengers.sql";
$sql = "SELECT * INTO OUTFILE '$backup_file' FROM $table_name";
mysqli_select_db($conn, "sample2DB");
$retval = mysqli_query($conn, $sql);
if(! $retval) {
    die("Error while attempting data backup: Sorry this operation is not successful ". mysqli_error());
}
echo "Backed up data successfully\n";
mysqli_close($conn);
?>
```

Code Snippet shows an example of using SELECT INTO OUTFILE query for creating back up of a table.



Figure: Output for Code Snippet

Using SQL Command Through PHP [2-2]

Code Snippet:

```
<?php
$dbhost = "localhost";
$username = "root";
$password = "root";
$conn = mysqli_connect($dbhost, $username, $password);
if(! $conn) {
    die("Could not connect: " . mysqli_error());
}
$table_name = "passengers";
$backup_file = "E:\\tmp\\passengers.sql";
$sql = "LOAD DATA INFILE '$backup_file' INTO TABLE $table_name";
mysqli_select_db($conn, "sample2DB");
$retval = mysqli_query($conn, $sql);
if(! $retval) {
    die("Encountered an error, Could not load data: " . mysqli_error());
}
echo "Loaded data successfully\n";
mysqli_close($conn);
?>
```

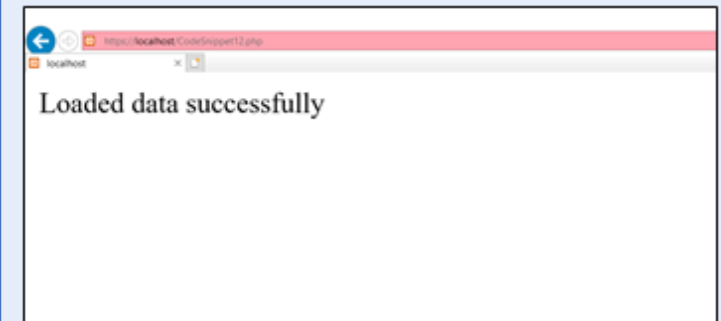


Figure: Output for Code Snippet

In case data that has already been backed up must be restored, run the `LOAD DATA INFILE` query. Code Snippet shows the same.

Using MySQL Binary mysqldump Through PHP

Code Snippet:

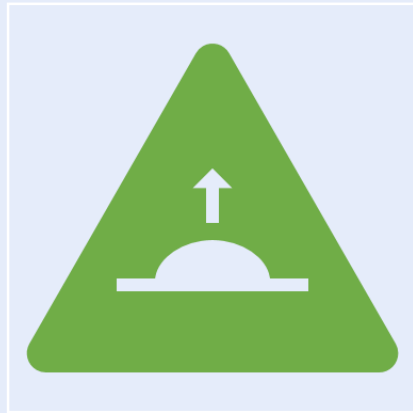
```
<?php
$dbhost = "localhost:3036";
$username = "root";
$password = "root";
$dbname = "sample2DB";
$backup_file = $dbname. date("Y-m-d-H-i-s") .
'.gz';
$command = "mysqldump --opt -h $dbhost -u
$username -p $password ". "lct2DB | gzip >
$backup_file";
system($command);
echo "Data backed up successfully";
?>
```

Code Snippet shows an example to perform a full database dump.

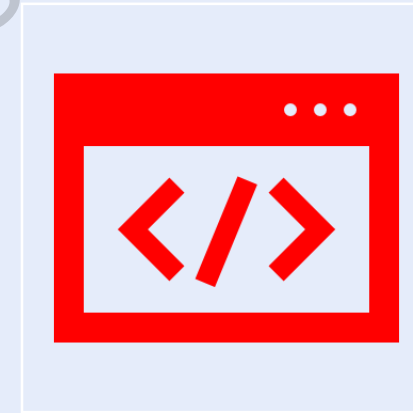


Figure: Output for Code Snippet

Using phpMyAdmin User Interface



If the phpMyAdmin user interface is available, then it is easy to take backup of the database. To do this, click the **export** link on the phpMyAdmin main page.



From the options available, choose any of the database options that user wishes to backup, he/she should go through the different SQL options, and enter the backup file name.

For Antech Centre Use Only

Selecting and Filtering Data from a MySQL Table

Adding filters to a query ensures that only the required data is displayed in the query result.

Sorting helps arrange the rows in the query result in an order which gives meaning to the data that will be used.

The `LIMIT` clause makes it easy to code multi page results or pagination with SQL and is very useful while working on large tables.

The three clauses used for these purposes are as follows:

- `WHERE`
- `ORDER BY`
- `LIMIT`

WHERE Clause

Code Snippet:

```
<html>
<body>
<?php
$servername = "localhost";
$username = "root";
$password = "root";
$dbname = "sample2DB";
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection status
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
$sql = "SELECT id, firstname, lastname FROM Passengers WHERE lastname='Shepard'";
$result = $conn->query($sql);
//The given code checks for the number of rows retrieved
if ($result->num_rows > 0) {
    // output data of each row
    while($row = $result->fetch_assoc()) {
        echo "<br> Passenger Id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"] . "<br>";
    }
} else {
    echo "0 results";
}
$conn->close();
?>
</body>
</html>
```

Code Snippet shows an example that selects the `firstname`, `lastname`, and `id` columns from the `Passengers` table and displays it on the page.

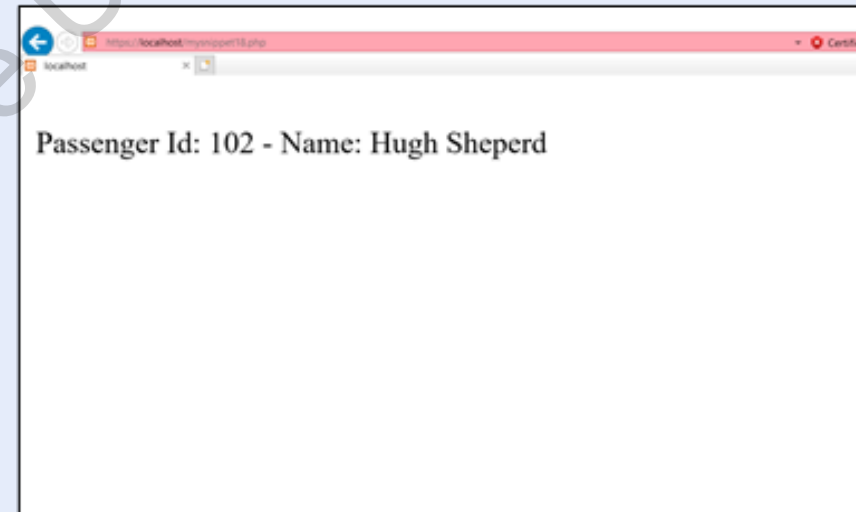


Figure: Output for Code Snippet

ORDER BY Clause

Code Snippet:

```
<html>
<body>
<?php
$servername = "localhost";
$username = "root";
$password = "root";
$dbname = "sample2DB";
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
$sql = "SELECT id, firstname, lastname FROM Passengers ORDER BY lastname";
$result = $conn->query($sql);
//The given if condition checks if any rows are fetched or not
if ($result->num_rows > 0) {
    // output data of each row
    while($row = $result->fetch_assoc()) {
        echo "<br> ID: ". $row["id"]. " - Name: ". $row["firstname"]. " " .
        $row["lastname"] . "<br>";
    }
} else {
    echo "0 results";
}
$conn->close();
?>
</body>
</html>
```

Code Snippet shows an example that selects the `firstname`, `lastname`, and `id` columns from the `Passengers` table and displays it on the page.

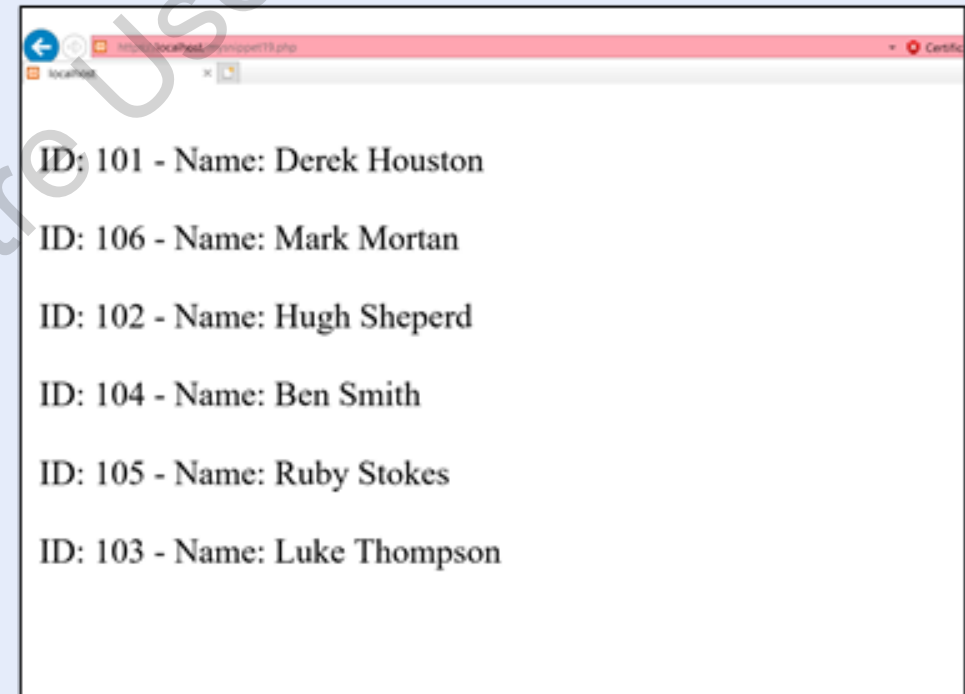


Figure: Output for Code Snippet

LIMIT Clause

When a large number of records are returned, it can impact the overall performance.

The `LIMIT` clause is used to limit returned rows in a MySQL query to a specific number.

The `LIMIT` clause is extremely useful while working with large tables, since it is easier to code multiple page results simultaneously.

For Antech Centre Use Only

Summary

- MySQL is an open-source relational database system and is often used with Web applications.
- PHP `mysqli_connect()` function is used to connect with the MySQL database.
- Using PHP, a MySQL database can be created and deleted.
- The `CREATE TABLE` statement is used to create a table in MySQL and can be used in a PHP script.
- Data can be both inserted into and retrieved from a MySQL database through PHP scripts.
- The `SQL UPDATE` statement through PHP function `mysqli_query` is used to update data in to a MySQL table.
- Data can be selected from a MySQL database using the `SELECT` statement.
- The `WHERE` clause is used to filter data based on specific criteria.
- The `ORDER BY` clause is used to display extracted results either in ascending or descending order.
- The `LIMIT` clause is used to specify the numbers of returning records.
- MySQL database can be backed up in three ways, namely, using SQL Command through PHP, using MySQL binary `mysqldump` through PHP, and using phpMyAdmin user interface.