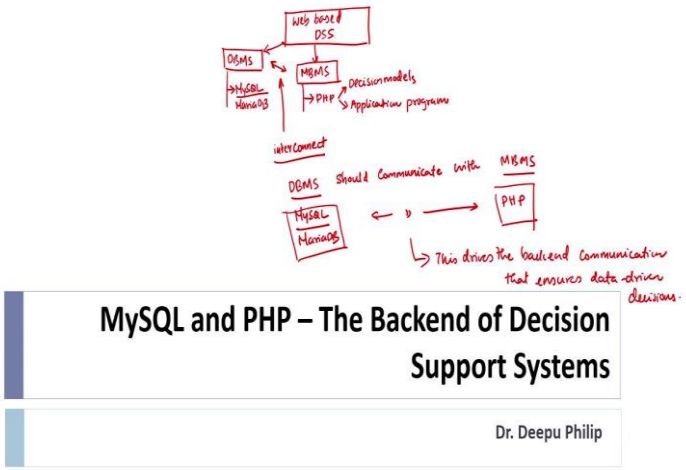
Computer Aided Decision Systems -
Industrial practices using Big Analytics
Professor Deepu Philip
Department of Industrial & Management Engineering
Indian Institute of Technology, Kanpur
Professor Amandeep Singh
Imagineering Laboratory
Indian Institute of Technology, Kanpur
Lecture 48
MySQL and PHP – The Backend of Decision Support Systems

Good afternoon, everyone. Welcome back to yet another lecture on Web-based Decision Support Systems for managers, business managers and practitioners. I am Deepu Philip from IIT Kanpur and we are continuing our lecture on the application program and especially how to interconnect.

(Refer Slide Time: 00:38)



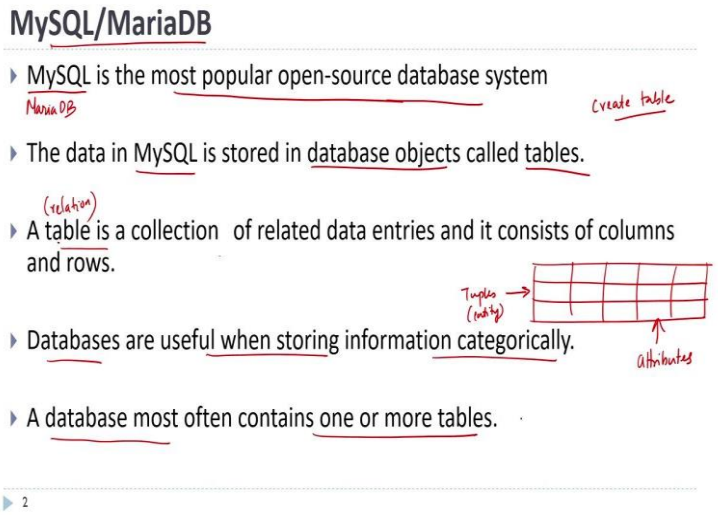
So, if you look at the slide, we have seen so far that the Web-based DSS has four major components. The first major component is DBMS (Database Management System) and the second major component is called MBMS (Model-based Management System). So, in DBMS, we have studied something like MySQL or MariaDB, the same. And, the Model-based Management System, we studied PHP which can build the decision models and it can also build the application program or application layer.

So, these two DBMS and MBMS need to talk with each other. So, another way to think about it is, this is the interconnect. So, this interconnect implies that, DBMS should communicate with MBMS. So, we can replace this with as MySQL or MariaDB should talk with PHP or should communicate with PHP. So, because you are using PHP to make this connection

happen. So, that is what we are trying to study or take a quick overview in today's lecture. And, this interconnection that drives the backend communication that ensures data driven decisions.

To make the decision you need data and how do you get the data? You get it from the DBMS and once you get it from the DBMS then, you can use the data to drive your decision. So, how is that going to happen is what we are going to see today.

(Refer Slide Time: 03:16)

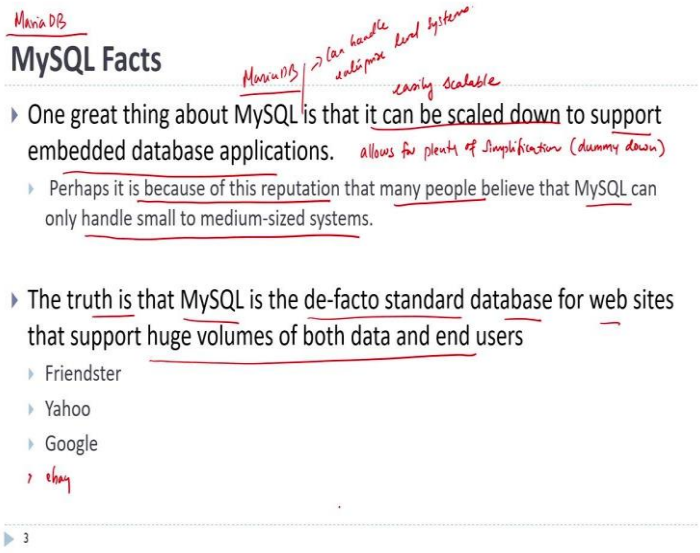


So, what is MySQL, MariaDB quick refresher not going to spend too much of a time, we already spent time on this, we already saw SQL, many aspects we already went through.

- MySQL is the most popular open-source database system and MySQL you can read it as MariaDB because MySQL is now bought by Oracle. So, even though MariaDB is the open-source fork of MySQL. It is the most popular open-source database system.
- And, in MySQL or MariaDB, the data is stored in the database objects called tables, there are different tables. We saw how to create the table. Remember, the create table, syntax and etcetera. We are going to create tables, query tables, select tables and all those kinds of things. So, what is a table, a table or another word to talk about, it is a relation. We talked about that also, what is the relation, what is an object, what is it tuple, all these things we have mentioned.
- So, a table is a collection of related data entries and it consists of columns and rows. So, remember, a database table will be something like this. And, each one of these columns were the attributes. Columns are the attributes and rows are the tuples or the entity. An instance of an entity goes into the row, we already saw that.

- So then, why do we use databases? Briefly, we said that databases are very useful or they are very powerful when storing information categorically. You want to classify it and store the information for that is why you use the databases.
- And, most of the time a database consists of one or more tables, so, tables or entities, you can think about it that way. So, most of the time, the database has more than one table.

(Refer Slide Time: 05:14)

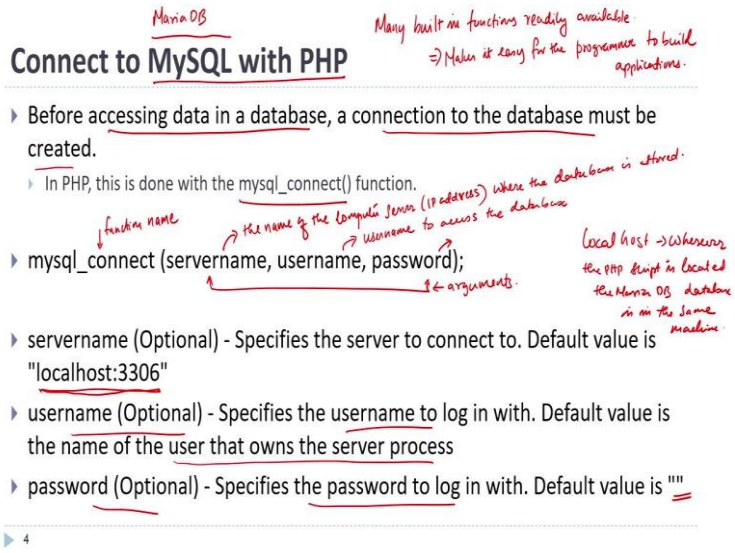


Now, some of the MySQL or MariaDB facts we need to look into again, as a quick refresher.

- The great thing about this one is that it can be scaled down, easily scalable to support embedded database applications. It allows, in another way to say, it allows for plenty of simplification. We can call it dummy dawn, keep it simple and stupid, that kind of a thing. So, it can be very easily scalable. And, because of the scalability, this beautiful scalability and this reputation of the scalability, many people believe that MySQL can handle only small to medium sized systems. MySQL is actually large. MySQL or MariaDB both can handle enterprise level systems. They are designed to handle enterprise level systems. But, unlike Oracle, or Microsoft SQL Server, or server or something like that, it beautifully scales down, you can make simplest applications out of it. Whereas, an oracle or Microsoft SQL Server, that is not equally easily possible. So, that is why people use it to make simple applications. People believe that it can only handle small and medium sized databases, small or medium sized systems.
- So, the main truth is that MySQL or MariaDB is the de-facto standard database. It is the basic database that for websites that supports huge volumes of data, both at the end

users stores data, and it also does analysis. So, Friendster, Yahoo, Google, eBay, all these things are driven by the backend database, MySQL. So, that says, it can handle very large systems and one side and simpler systems on the other side.

(Refer Slide Time: 07:26)



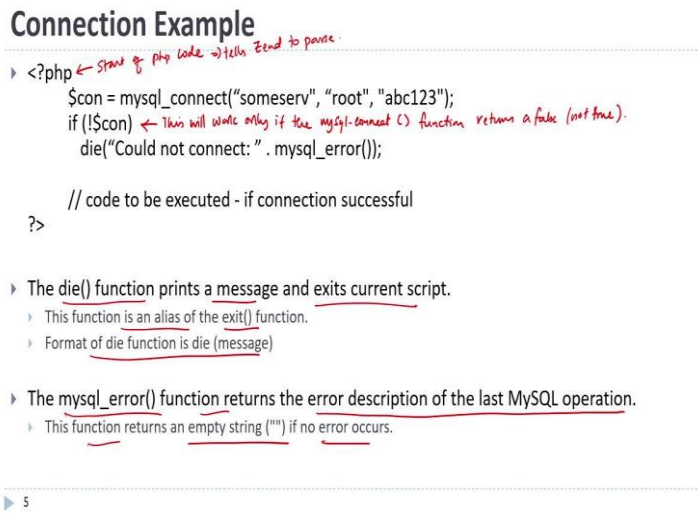
And, another beauty of MySQL, we should understand is, MySQL integrates with PHP or MySQL or MariaDB, either one, both of them have very good seamless integration with PHP. So, another way is that many built-in functions are readily available. So, which means it makes easy for the programmer to build applications. You do not worry too much about how to do this, you can easily build the application.

- So, the first thing you need to do is before accessing the data in a database using PHP, first, you need to establish a connection to the database. So, PHP provides you with a built-in function called MySQL connect function. MySQL_connect also connects with MariaDB, you do not have to change, this is the same thing. So, the MySQL_connect, this is a function as I told you earlier it is a built-in function, you do not need to code the function, it is already coded for you, pre-done for you, you just need to call the function.
- So, the MySQL_connect function, the format is, this is the function name, and these are the three arguments.
  - i) The first one is a servername. The servername is the name of the computer server or you can call it as IP address, where the database is stored. The servername is the name of the computer server where you are storing the database.
  - ii) Then, the username is the username to access the database in that server

iii) And, the password is the associated password.

- So, as I said earlier, servername is optional. If you do not give it, that is fine, it will default. There is a default value. The default value is Localhost. "localhost:3306" means that, if you are not given any servername, it assumes that wherever this localhost implies that wherever the PHP script is located, the MariaDB database is on the same computer, same machine. So, it says, wherever the script is, it will also be at the same place. So, that is what this 'localhost:3306' means, otherwise you give it a name to whichever server you need it to be connected to.
- The username is optional. So, again, you need to give the username ideally. If you do not give the username, the default value is the name of the user who owns that server process. So, if I do not do anything, whoever owns the server process, or if I am connecting to something, whatever the name it came through, it will take that and connect accordingly.
- And then, the password is also optional. The default value is an empty, null string. So, as I mentioned earlier, the username and password and the servername are required to connect to the database. This establishes the connection to the database.

(Refer Slide Time: 11:28)



So, here is an example.

- So, <?php is the start of the PHP code. So, it tells ZEND to parse, and starts the parsing. So, I have a variable called $connection con equal to MySQL_connect, I call the function and "someserv" some name of the server, username is "root", password is

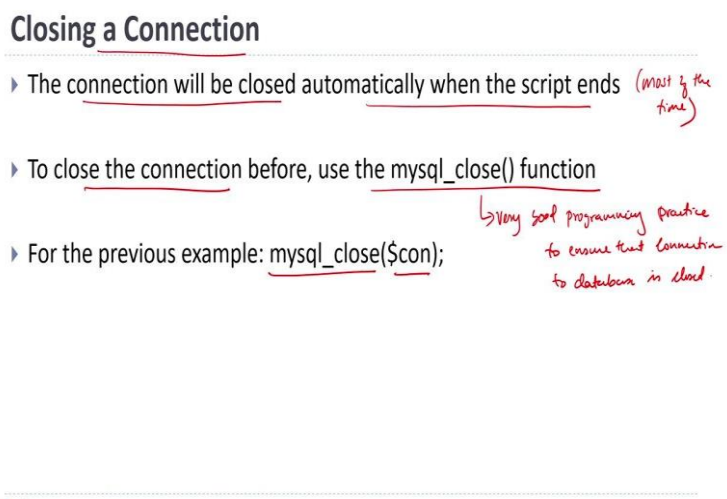
"abc123" for the time being. So, if the connection is successful, if these parameters are fine and the connection is successful, MySQL_connect returns a value called true.

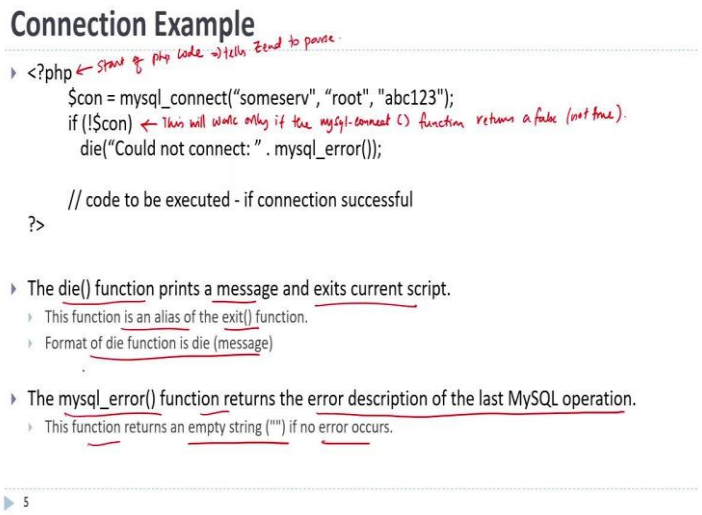
So then, what I am saying is that if not $ connection means if the value of $ connection is not true, that means I could not establish the connection. Then, I use PHP. The PHP function called die could not connect, and whatever the MySQL error, I use wrong username, wrong password, whatever it is, I print out whatever the error is.

So, this will only be done if the connection establishment fails. Otherwise, it will continue the remaining part of the code, if the connection is successful. So, that is the logic. So, this will work only if the MySQL_connect function returns a false or not true. So, by default, it returns true, if it does not return true, returns something else, then, this $connection will not be true. And then, we can print whatever is there.

- So, the die function prints a message and exits the current script. So, whatever you are doing, die will just make sure that you do not go any forward, print, error, message and etcetera. So, this is an alias of the exit() function. Also, the advantage is that the message, whatever you want to print, can be the format of the die function is die with whatever message, you want to print it.
- And then, the other function that you are having is the MySQL error function. And returns, what it does is, it returns an error description of the latest MySQL operation. If no errors if the connection is successful, absolutely no issues, it will return an empty string. If it does not return an empty string, then, that means the connection could not be established and there was an error. So, that is the important part.

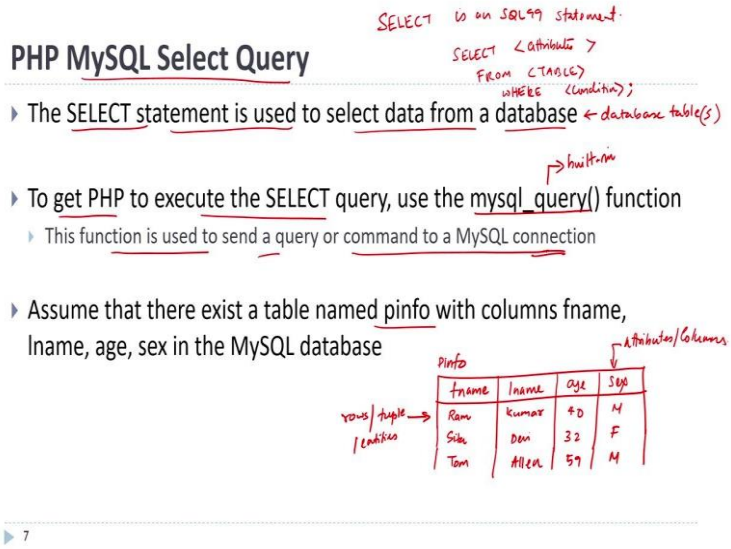
(Refer Slide Time: 14:04)





- Now, once you establish a connection with the database, it is important that you should also close the connection. Most of the time, the connection will be closed automatically when the script ends. There are certain times this does not happen. So, it is a good practice to make sure that you close the connection.
- And, to close the connection, you use the MySQL_close() function. So, this makes sure that there is an unnecessary connection to the database and it will not slow down your server. So, it is a very good programming practice to ensure that connection to the database is closed.
- So, if you use this connection as the variable, if that is the variable that you are going to use, then, the example is MySQL_close ($con) which means close the connection. So, it will make sure that everything, all connections with the server, database server is cut, and it will free out the resource.

(Refer Slide Time: 15:21)



Now, how do you SELECT query using PHP?

- So, we have seen that the SELECT statement is an SQL 99 statement. It has its own syntax. Remember, SELECT, then, we gave <attributes> FROM <TABLE> WHERE <condition> we have done this in the previous. We saw this structure of select SQL syntax. So, the SELECT statement is that we can use it to select data from a database. From where in a database? Database table, from a specific table you can collect the data. You can even take it from database tables also, you can use join and those kinds of things to do tables or tables. But, how do you do that with a PHP script? Why would you do that within a PHP script?
- So, to do it, get it in a PHP script. What you do is, to execute the SELECT query or for that matter any query, does not have to be a SELECT query, you use the MySQL_query() function. It is also a built-in function. This is also built-in, it is written, coded, tested, everything, all you need to do is just use the function.

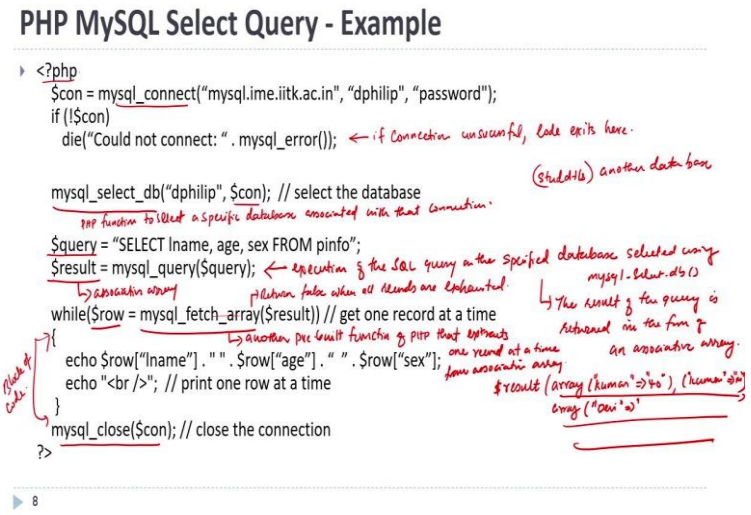
So, what does this function do? This function is capable of sending a query or command to the MySQL connection. So, you use the MySQL connection. Once you establish a connection, then, you use the MySQL query to access the database whether to run a SELECT query or insert query or delete query whatever the query you want to do, you use the MySQL query function.

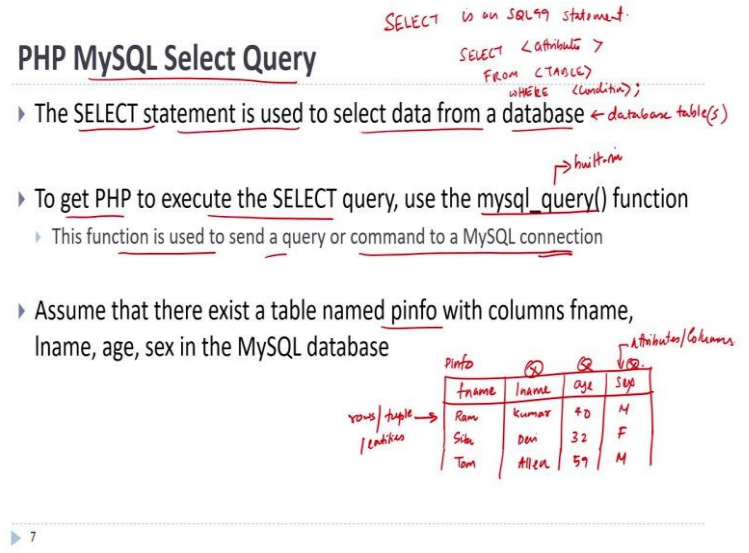
- So, let us assume for the time being that there exists a table named Pinfo. So, the table Pinfo is something like this, 'Personal Info'. Let us take that for example with columns

Fname (first name), Lname (last name), age, sex. So, we can have one thing called Ram Kumar, age 40, male. Sita Devi, 32, female like this and then, you can have Tom Allen, 59, male like this. So, there is data in this one.

So, you have first name, last name, age and sex. These are the attributes or columns. These are the rows or tuples or entities. Each one of them is, Ram Kumar is one entity with the personal info, Sita Devi is another entity of the person. So, this is just to refresh you. That is the case.

(Refer Slide Time: 18:41)





Then, the code works like this. First one is, it says <?php. You call the mysql_connect function. And here, I am giving it a servername for example, just to show "mysql.ime.iitk.ac.in". So, it is one of the servers in my lab and "dphilip" is the username and this "password", I am not going to show you what my password is. And then, $con is the variable so, it is a connection

variable established. So, what it does is, the MySQL connection will go to query the server and say here I have a username and a password. Do you have a database in which I can access this credentials work?

So, this function checks and MySQL says yes, I have a database and these credentials are valid. So then, if it is true, it will return an empty string. If not, it will send an error. So then, I use the die statement and the code exits here. If connection unsuccessful, code exits here at mysql_error() leave out at this point. If this is not, then, go to the next one.

So, I will say mysql_select_db. This is to select a PHP function to select a specific database associated with that connection. So, let us say in this particular database we enter the username "dphilip" and password. I have another database called "dphilip". So, that is why I am connecting to the "dphilip" database.

Let us say I have another one called student details. Student underscore details (DTLS). If this was another database so, instead of "dphilip", I would have used this name there. So, you basically say whichever the database you want to collect, you want to use using that particular connection $connection, you specify that here in this line, so that you can connect to the database.

Now, here is a $query, it is a string variable and string variable it says SELECT Lname, age and sex from Pinfo. So, we look into a Pinfo we are selecting Lname, age and sex. These three things we are selecting, that is a query. So, you write the query in the form of a string here. Then, you say $result is another variable, you say MySQL query and you give the variable as $query. So, this query gets executed right here. So, this is execution of the SQL query on the specified database, selected using the mysql_select_db function. So, whatever database you selected using this, this query will run on that database. So, make sure that the database in which Pinfo is stored is exactly the correct one.

And then, another important thing is, the result of the query is returned in the form of an associative array. The data comes back in the form of an associative array. So, the result, this variable, is an associative array. So, in this associative array, how will that variable be stored? It will be stored somewhat in a very weird fashion. The associative array will create something like this:

$result (array (“Kumar”=”40”), (“Kumar”=”M”)

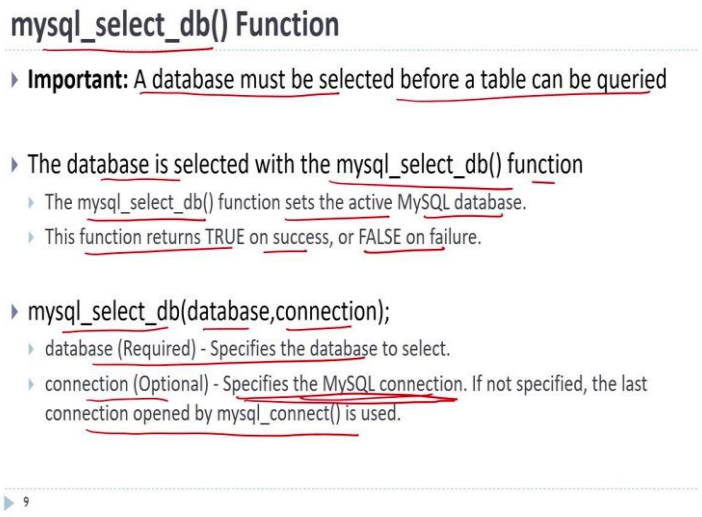
Array(“Devi”=)

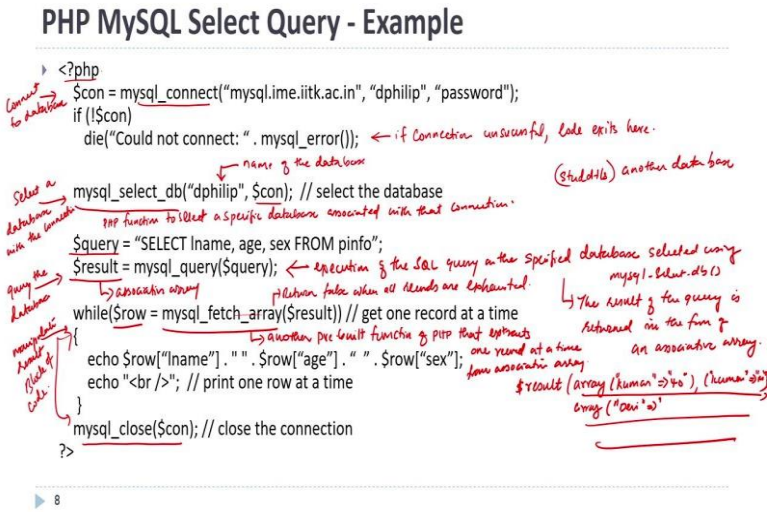
So, it will store the format of a fun associative array. So, all I just need to do is, to get one row out of this associative array I need to use another function called mysql_fetch_array. This is another pre-built function of PHP that extracts one record at a time from an associative array. So, it fetches one line at a time or one row at a time. So, it will first take this and then, take the next one then, it takes the next one like this and will keep on repeating until there are no more records available. So, the $row variable will be one line or one record of the associative array of the result. So, use that and the $row will contain that and then, you say $row you the key Lname. So, this is the Lname, age and sex will be the key so, that is what it is.

So, Lname it will print the last name of Kumar, age it will print it and then, the sex it will print it. And then, you line break and it will keep on repeating all of this. So, while the $row = mysql_fetch_array, if there is one row to return then, the stuff is true. So, this whole execution, this whole condition will remain true, remember. So, the While Loop, this is the block of code of the While Loop. And then, it will run and when all the records are over, then, this mysql_fetch_array will return false. So, this will return false when all records are exhausted.

When all records are exhausted, it will return false, and the loop will stop, and things will come up. So, you keep on seeing the results. And then, once it is done, you close the connection to the MySQL database. So, that is the MySQL close connection. So, this is a simple example of how you can run the SQL in a particular database within MySQL.

(Refer Slide Time: 25:56)





So, as I mentioned earlier, the mysql_select_db function. So, one important thing that you remember is that the first thing is connect to the database then, select a database with the connection, then query the database, manipulate the result. So, this order of logic.

- So, in that regard, a database must be selected before a table can be queried, it is very important. So, after the connection, you have to select the database.
- The selection of the database is done with the help of MySQL_select_db function. So, the MySQL_select_db function determines which database to be selected. So, this function sets the active MySQL database. So, using the connection, it will find this database is there. I am going to make it an active database. The function will return TRUE on success, if it finds a database like this and I can make it active, it will return TRUE on success or return FALSE on failure. I have no database like this, it will return false.
- The format of this is MySQL_select_database, the name of the database and the connection. The name of the database is a required thing you can specify what it is. So, this is the name of the database and connection is optional, in this case specified what the connection is. If it is not specified then, it will use the last opened connection, whatever we the last opened what, it will pick it up and use it by default. So, it is always a better idea to specify the MySQL connection, whichever connection you want to use is a better idea to specify, otherwise, you can get into bigger trouble, sort of.

(Refer Slide Time: 28:02)

mysql_fetch_array() Function

- The mysql_fetch_array() function returns a row from a recordset as an associative array and/or a numeric array
- This function gets a row from the mysql_query() function and returns an array on success, or FALSE on failure or when there are no more rows
- mysql_fetch_array(data, array_type)
  - If array_type is not specifed MYSQL_BOTH is used – For both associative and numeric array
- After the data is retrieved, this function moves to the next row in the recordset.
  - Each subsequent call to mysql_fetch_array() returns the next row in the recordset
  - Field names returned by this function are case-sensitive

10

PHP MySQL Select Query - Example

```
<?php
$con = mysql_connect("mysql.ime.iitk.ac.in", "dphilip", "password");
if (!$con)
  die("Could not connect: " . mysql_error());

mysql_select_db("dphilip", $con); // select the database

$query = "SELECT lname, age, sex FROM pinfo";
$result = mysql_query($query);

while($row = mysql_fetch_array($result)) // get one record at a time
{
  echo $row["lname"] . " " . $row["age"] . " " . $row["sex"];
  echo "<br />"; // print one row at a time
}
mysql_close($con); // close the connection
?>
```

8

PHP MySQL Select Query

- The SELECT statement is used to select data from a database
- To get PHP to execute the SELECT query, use the mysql_query() function
  - This function is used to send a query or command to a MySQL connection
- Assume that there exist a table named pinfo with columns fname, lname, age, sex in the MySQL database

fname	lname	age	sex
Ram	kumar	40	M
Sita	Devi	32	F
Tom	Allen	59	M

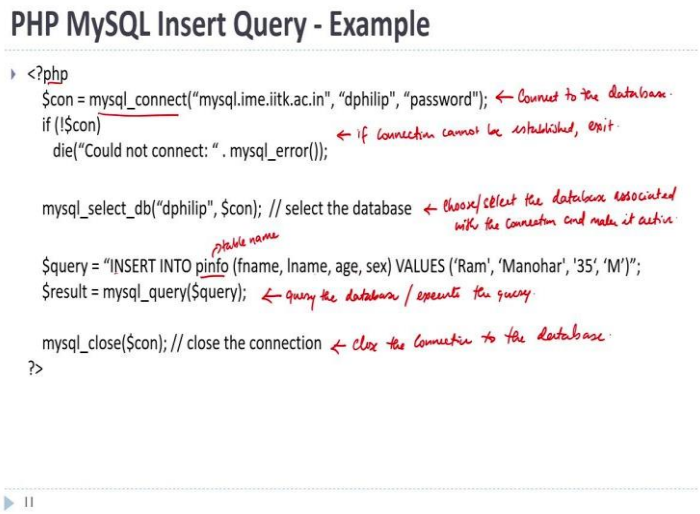
7

- Then, as I mentioned earlier, the MySQL array function, I have shown you this code, the mysql_fetch_array, I have briefly explained it. So, the mysql_fetch_array function returns a row from the recordset as an associative array, or a numeric array. So, most of the time, 99 percent of the time, you get it in the form of an associative array, because database tables are complicated. So, each time when you call mysql_fetch_array, it returns a row, one row. So, this function gets a row from the mysql query function, and returns an array on success or false on failure when there are no more rows.
- So, if there is a row, it returns a row of success. So, if there is this result mysql_fetch_array, it will return a row, if there is a record, if it is not, it will return false. And, if it is false, the while loop will fail. And, you can get out of this.
- So, the syntax of this is mysql_fetch_array, data and array type. If array type is not specified, MySQL_BOTH is used for both associative and numeric arrays. So, typically, what happens is, in this code, if you see we have just mysql_fetch_array result. We leave it to the freedom of PHP to decide whether you want to use a numeric array or an associative array. Most of the time it goes into an associative array.
- So, after the data is retrieved, once you run this code and you retrieve the data. After you retrieve the data, the function moves to the next row in the recordset. You do not have to worry, you do not increment about all this, this is all built-in and tested, tried, you can just use the whole thing. So, each subsequent call to MySQL_fetch_array returns the next row on the recordset. So, every time you call it will return to the next row.

So, once it extracts it points to the next one, extract, points to the next library like that. And, the field names returned by this function are case sensitive. So, in this case, for example, if I use a name like this, if this is the one, I am using as the attribute and if I type it Lname like this, I will get an error. So, make sure that whatever be the attribute name or the column name in the table that is what you use exactly here.

So, this should exactly match column names as critical because they are case sensitive and similar, so, now we have seen how to run a SELECT query and how to input data into the database.

(Refer Slide Time: 30:39)



Here is another example for that, the same way PHP as I said earlier, <?php starts the parsing, establishes a connection using the mysqk_connect.

So, this is the connection to the database. And, this is if a connection cannot be established, exit, do not do anything more.

If it is possible then, choose or select the database associated with the connection and make it active.

So then, the next one is to query the database or execute the query. So, we know that there is a table called Pinfo, already there. So, the Pinfo, you want to insert some values. So, your query is INSERT INTO Pinfo, it is the table name. And, these are Fname, Lname, age and sex. These are the four aspects of this one. And, we are saying that Fname insert the value Ram, Lname insert the value Manohar, age insert 35, sex insert M, that is what this whole thing is. And, it says the result, mysql_query ($query). So, in a select returns an associative array. In an insert, it does not return any associative array. It says whether the query got executed or not.

So, insert means you are storing data into the database. So, once you capture values from the HTML form, you can use insert queries to store it into the database. Once the query is done, you close the connection to the database. So, once this is done, your manipulation requirement with the connection of the database is over. So, in a general sense, this is how you do the backend integration.

I just showed you two examples, select and insert. Same way you can use delete, same way you can use update. I already showed you what the queries are. Using the same example, you can modify them or you can modify these examples and try it out with your MySQL or

MariaDB database and try to see how you can actually achieve it, as part of your course learning.

So, I showed you select and insert, try to update and delete on your own and try to make sure that this works in the backend. And, if there is any doubt or other things, we will be available in the forum to help you in this regard. Thank you very much for your patience. Thank you.