MySQL Access Using PHP

PDO and mysqli connectors

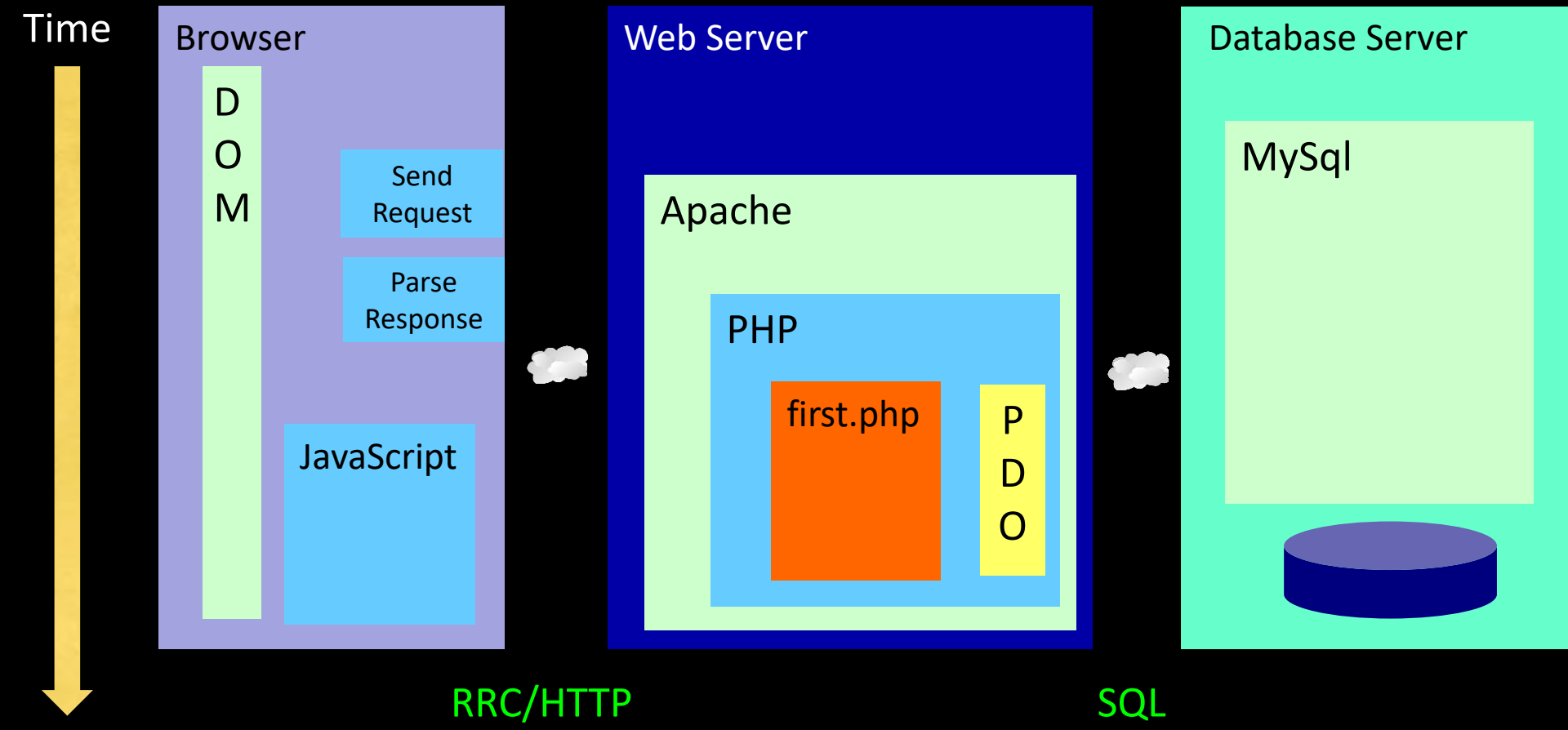
PHP Database Connectors

PHP has the ability to access and manipulate any database that is ODBC compliant

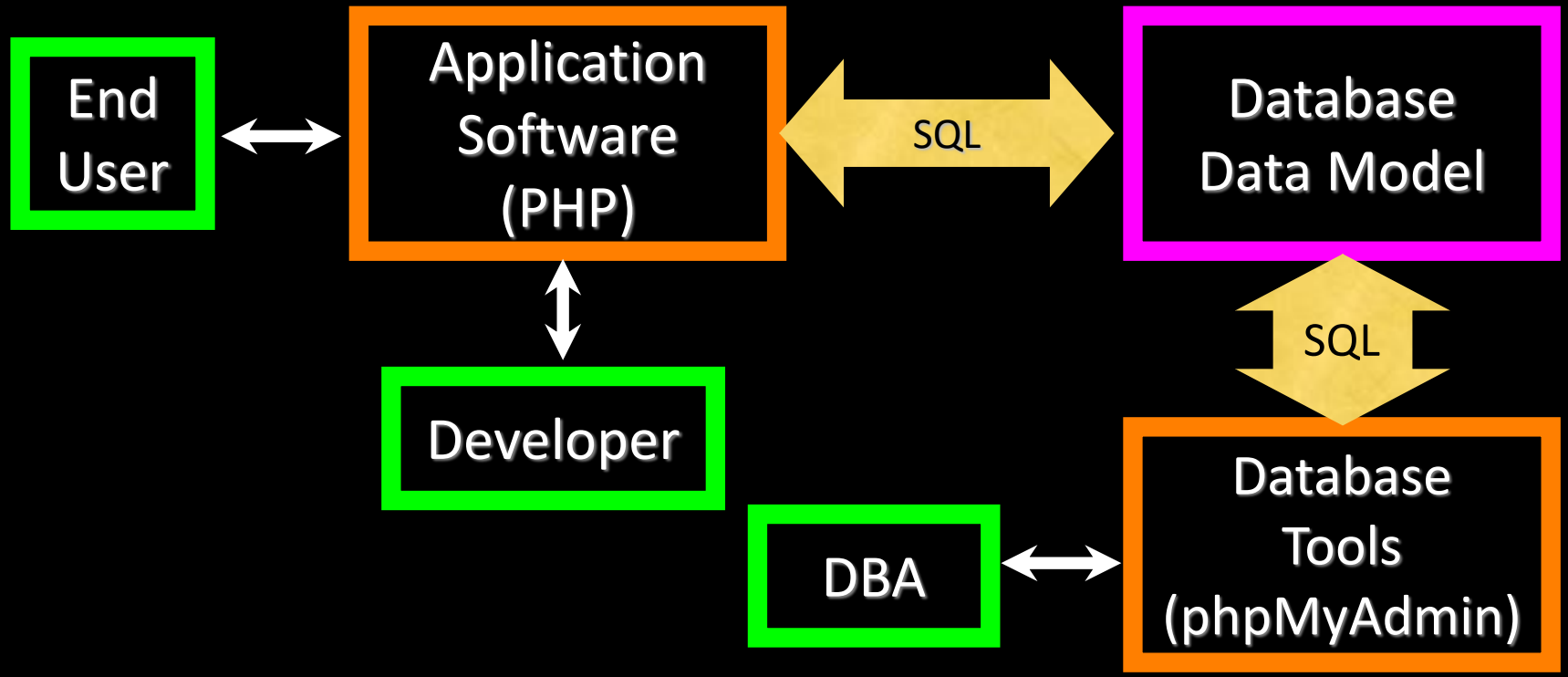
PHP includes functionality that allows you to work directly with different types of databases, without going through ODBC

PHP supports SQLite, database abstraction layer functions, and PEAR DB

PHP Database Connectors



PHP Database Connectors



PHP MySQL Connectors

PHP is evolving - there are three ways to access MySql

- Legacy non-OO mysql_ routines (deprecated)
- New mysqli (OO version that is similar to mysql_)
- PDO - Portable Data Objects

PHP MySQL Connectors

```php
<?php
// mysqli
$mysqli = new mysqli("example.com", "user", "password", "database");
$result = $mysqli->query("SELECT 'Hello, dear MySQL user!' AS _message FROM DUAL");
$row = $result->fetch_assoc();
echo htmlentities($row['_message']);

// PDO
$pdo = new PDO('mysql:host=example.com;dbname=database', 'user', 'password');
$statement = $pdo->query("SELECT 'Hello, dear MySQL user!' AS _message FROM DUAL");
$row = $statement->fetch(PDO::FETCH_ASSOC);
echo htmlentities($row['_message']);

// mysql
$c = mysql_connect("example.com", "user", "password");
mysql_select_db("database");
$result = mysql_query("SELECT 'Hello, dear MySQL user!' AS _message FROM DUAL");
$row = mysql_fetch_assoc($result);
echo htmlentities($row['_message']);
?>
```

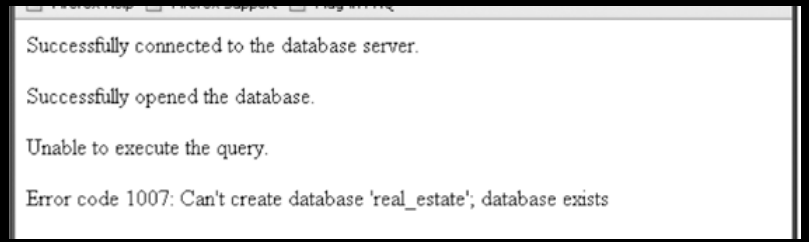
SQL Create Database and User

```
CREATE DATABASE misc;

GRANT ALL ON misc.* TO 'fred'@'localhost' IDENTIFIED BY 'zap';

GRANT ALL ON misc.* TO 'fred'@'127.0.0.1' IDENTIFIED BY 'zap';

USE misc;    (if you are at the command line)
```

PHP Creating Databases

Use the `CREATE DATABASE` statement with the `mysqli_query()` function to create a new database

```
$SQLstring = "CREATE DATABASE real_estate";
$QueryResult = @mysqli_query($DBConnect, $SQLstring)
    Or die("<p>Unable to execute the query.</p>"
    . "<p>Error code " . mysqli_errno($DBConnect)
    . ": " . mysqli_error($DBConnect)) . "</p>";
echo "<p>Successfully executed the query.</p>";
mysqli_close($DBConnect);
```



SQL Create Tables

```
CREATE TABLE users (
   user_id INTEGER NOT NULL AUTO_INCREMENT,
   name VARCHAR(128),
   email VARCHAR(128),
   password VARCHAR(128),
   PRIMARY KEY(user_id),
   INDEX(email)
) ENGINE=InnoDB CHARSET=utf8;
```

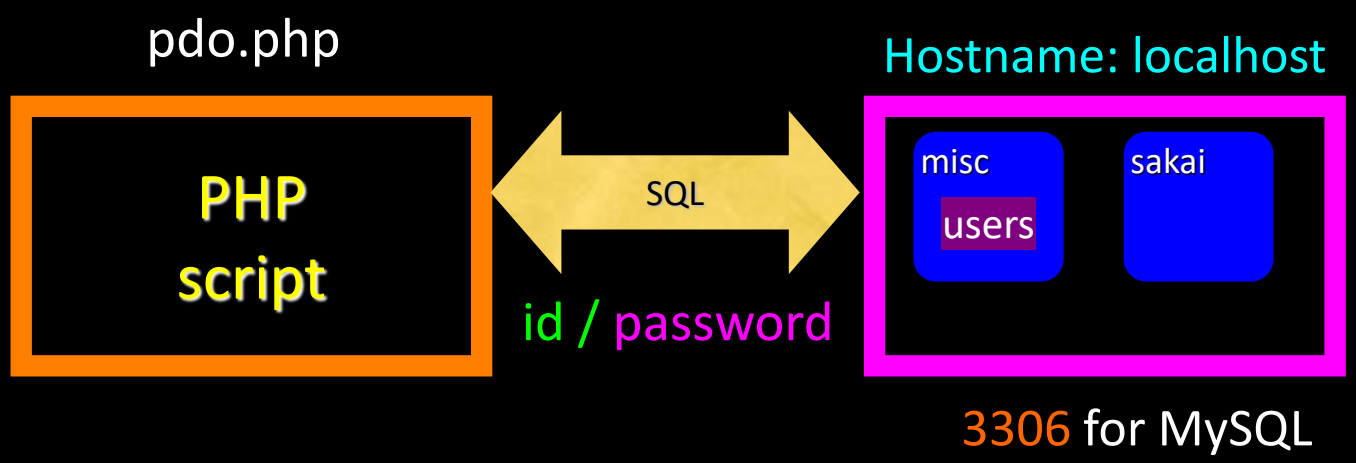
```
mysql> describe users;
+----------+------------------+------+-----+---------+----------------+
| Field    | Type             | Null | Key | Default | Extra          |
+----------+------------------+------+-----+---------+----------------+
| user_id  | int(11)          | NO   | PRI | NULL    | auto_increment |
| name     | varchar(128)     | YES  |     | NULL    |                |
| email    | varchar(128)     | YES  | MUL | NULL    |                |
| password | varchar(128)     | YES  |     | NULL    |                |
+----------+------------------+------+-----+---------+----------------+
```

Inserting Records

```
INSERT INTO users (name,email,password) VALUES ('Chuck','csev@umich.edu','123');
INSERT INTO users (name,email,password) VALUES ('Glenn','gg@umich.edu','456');
```

```
mysql> select * from users;
+---------+-------+-----------------+----------+
| user_id | name  | email           | password |
+---------+-------+-----------------+----------+
|       1 | Chuck | csev@umich.edu  | 123      |
|       2 | Glenn | gg@umich.edu    | 456      |
+---------+-------+-----------------+----------+
```

PHP Database Connection




```
$pdo = new PDO('mysql:host=localhost;port=3306;dbname=misc','fred',`zap123');
```

PHP Database Connection

```php
<?php
echo "<pre>\n";
$pdo=new PDO('mysql:host=localhost;port=3306;dbname=misc','fred', 'zap');
$stmt = $pdo->query("SELECT * FROM users");
$rows = $stmt->fetchAll(PDO::FETCH_ASSOC);
print_r($rows);
echo "</pre>\n";?>
```

```
mysql> select * from users;
+---------+-------+----------------+--------------+
| user_id | name  | email          | password     |
+---------+-------+----------------+--------------+
|       1 | Chuck | csev@uonbi.ac.ke | zap123     |
|       2 | Glenn | gg@ uonbi.ac.ke  | zap456     |
+---------+-------+----------------+--------------+
```

```
Array(
    [user_id] => 1
    [name] => Chuck
    [email] => csev@uonbi.ac.ke
    [password] => zap123
)

Array(
    [user_id] => 2
    [name] => Glenn
    [email] => gg@uonbi.ac.ke
    [password] => zap456
)
```

PHP Database Connection

second.php

```
<?php
$pdo = new PDO('mysql:host=localhost;port=3306;dbname=misc',
    'fred', 'zap');
$stmt = $pdo->query("SELECT name, email, password FROM users");
$rows = $stmt->fetchAll(PDO::FETCH_ASSOC);
echo '<table border="1">'."\n";
foreach ( $rows as $row ) {
    echo "<tr><td>";
    echo($row['name']);
    echo("</td><td>");
    echo($row['email']);
    echo("</td><td>");
    echo($row['password']);
    echo("</td></tr>\n");
}
echo "</table>\n";?>
```

```
<table border="1">
<tr><td>Chuck</td><td>csev@uonbi.ac.ke</td><td>zap123</td></tr>
<tr><td>Glenn</td><td>gg@ uonbi.ac.ke</td><td>zap456</td></tr>
</table>
```

PHP Database Connection

```
<?php
$pdo = new PDO('mysql:host=localhost;port=3306;dbname=misc','fred', 'zap');
// See the "errors" folder for details...
$pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

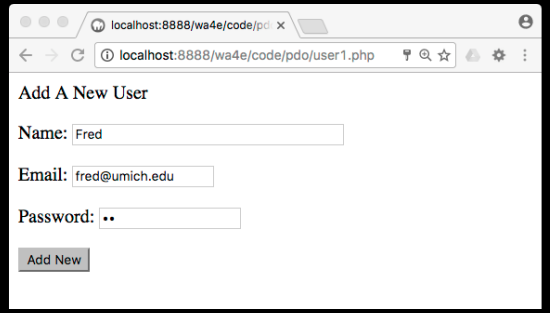
```
// This script imports the connection file named pdo.php like in C
<?php
require_once "pdo.php";
echo "<pre>\n";
$stmt = $pdo->query("SELECT * FROM users");
$rows = $stmt->fetchAll(PDO::FETCH_ASSOC);
print_r($rows);
echo "</pre>\n";?>
```

pdo.php

3306 MySQL

HTML Form

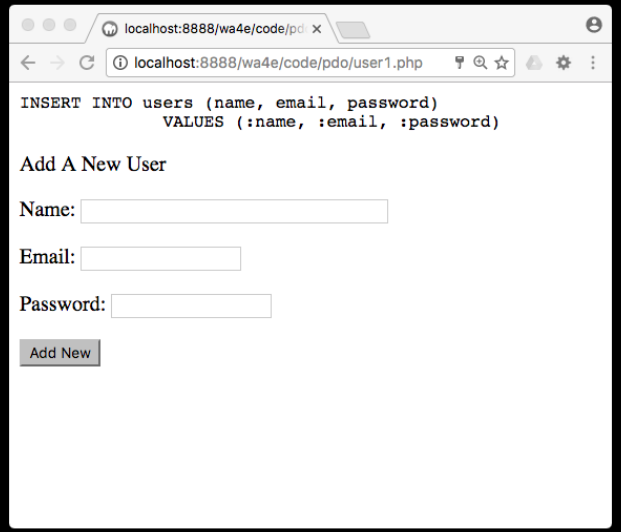
```
}
?><html><head></head><body>
<p>Add A New User</p>
<form method="post">
<p>Name:<input type="text" name="name" size="40"></p>
<p>Email:<input type="text" name="email"></p>
<p>Password:<input type="password" name="password"></p>
<p><input type="submit" value="Add New"/></p>
</form>
</body>
```



PHP Database Connection

```
<?php
require_once "pdo.php";
if ( isset($_POST['name']) && isset($_POST['email'])&& isset($_POST['password'])) {
    $sql = "INSERT INTO users (name, email, password) VALUES (:name, :email, :password)";
    echo("<pre>\n".$sql."\n</pre>\n");
    $stmt = $pdo->prepare($sql);
    $stmt->execute(array(
        ':name' => $_POST['name'],
        ':email' => $_POST['email'],
        ':password' => $_POST['password']));
}
?>
```

HTML to Database Insert




```
mysql> select * from users;
+---------+-------+----------------+----------+
| user_id | name  | email          | password |
+---------+-------+----------------+----------+
|       1 | Chuck | csev@uon.ac.ke | 123      |
|       2 | Glenn | gg@ uon.ac.ke  | 456      |
|       3 | Sally | sally@uon.ac.ke| 123      |
|       4 | Fred  | fred@uon.ac.ke | YO       |
+---------+-------+----------------+----------+
```

Fetching data from a database table

```
<?php
require_once "pdo.php";
$stmt = $pdo->query("SELECT name, email, password FROM users");
$rows = $stmt->fetchAll(PDO::FETCH_ASSOC);
<table border="1">
foreach ( $rows as $row ) {
    echo "<tr><td>";
    echo($row['name']);
    echo("</td><td>");
    echo($row['email']);
    echo("</td><td>");
    echo($row['password']);
    echo("</td></tr>\n");
}
?>
```

Deleting data from a database table


```
<?php
require_once "pdo.php";

if ( isset($_POST['user_id']) ) {
    $sql="DELETE FROM users WHERE user_id = :zip";
    echo "<pre>\n$sql\n</pre>\n";
    $stmt = $pdo->prepare($sql);
    $stmt->execute(array(':zip'=>$_POST['user_id']));
}
?>
```

Opening and Closing MySQL Connection

Open a connection to a MySQL database server with the `mysqli_connect()` function

The `mysqli_connect()` function returns a positive integer if it connects to the database successfully or false if it does not

Assign the return value from the `mysqli_connect()` function to a variable that you can use to access the database in your script

Opening and Closing MySQL Connection

The syntax for the `mysqli_connect()` function is:

```
$connection = mysqli_connect("host"[,"user","password","database"])
```

The *host* argument specifies the host name where your MySQL database server is installed

The *user* and *password* arguments specify a MySQL account name and password

The *database* argument selects a database to use

MySQL Connection functions

Function	Description
`mysqli_get_client_info()`	Returns the MySQL client version
`mysqli_get_client_version()`	Returns the MySQL client version as an integer
`mysqli_get_host_info(connection)`	Returns the MySQL database server connection information
`mysqli_get_proto_info(connection)`	Returns the MySQL protocol version
`mysqli_get_server_info(connection)`	Returns the MySQL database server version
`mysqli_get_server_version(connection)`	Returns the MySQL database server version as an integer

Selecting a Database

Select a database with the `use` *`database`* statement when you log on to the MySQL Monitor

The syntax for the `mysqli_select_db()` function is:

```
mysqli_select_db(connection, database)
```

The function returns a Boolean value of true if it successfully selects a database or false if it does not

Reasons for not connecting to a database server include may be due to database server is not running, wrong credentials or network connectivity to remote host

Error handling

Writing code that anticipates and handles potential problems is often called **bulletproofing**

Bulletproofing techniques include:

- Validating submitted form data
- Using the **error control operator (@)** to suppress error messages
- The `die()` and `exit()` functions terminate script execution. Call the `die()` and `exit()` functions as separate statements or by appending either function to an expression with the `Or` operator

Error handling

```
$DBConnect = @mysqli_connect("localhost", "root", "paris");
if (!$DBConnect)
    die("<p>The database server is not available.</p>");
echo "<p>Successfully connected to the database server.</p>";
$DBSelect = @mysqli_select_db($DBConnect, "flightlog");
if (!$DBSelect)
    die("<p>The database is not available.</p>");
echo "<p>Successfully opened the database.</p>";
// additional statements that access the database
mysqli_close($DBConnect);
```

Error handling

```
$DBConnect = @mysqli_connect("localhost", "dongosselin", "rosebud")
    Or die("<p>The database server is not available.</p>");
echo "<p>Successfully connected to the database server.</p>";
@mysqli_select_db($DBConnect, "flightlog")
    Or die("<p>The database is not available.</p>");
echo "<p>Successfully opened the database.</p>";
// additional statements that access the database server
mysqli_close($DBConnect);
```

MySQL error functions

Function	Description
`mysqli_connect_errno()`	Returns the error code from the last database connection attempt or zero if no error occurred
`mysqli_connect_error()`	Returns the error message from the last database connection attempt or an empty string if no error occurred
`mysqli_errno(`*connection*`)`	Returns the error code from the last attempted MySQL function call or zero if no error occurred
`mysqli_error(`*connection*`)`	Returns the error message from the last attempted MySQL function call or an empty string if no error occurred
`mysqli_sqlstate(`*connection*`)`	Returns a string of five characters representing an error code from the last MySQL operation or 00000 if no error occurred

MySQL error functions

```
$User = $_GET['username'];
$Password = $_GET['password'];
$DBConnect = @mysqli_connect("localhost", $User, $Password)
    Or die("<p>Unable to connect to the database server.</p>" . "<p>Error code
    " . mysqli_connect_errno()
    . ": " . mysqli_connect_error()) . "</p>";
echo "<p>Successfully connected to the database server.</p>";
@mysqli_select_db($DBConnect, "flightlog")
    Or die("<p>The database is not available.</p>");
echo "<p>Successfully opened the database.</p>";
// additional statements
mysqli_close($DBConnect);
```

MySQL error functions

```
$User = $_GET['username'];
$Password = $_GET['password'];
$DBConnect = @mysqli_connect("localhost", $User, $Password)
    Or die("<p>Unable to connect to the database server.</p>"
    . "<p>Error code " . mysqli_connect_errno()
    . ": " . mysqli_connect_error()) . "</p>";
echo "<p>Successfully connected to the database server.</p>";
@mysqli_select_db($DBConnect, "flightplan")
    Or die("<p>Unable to select the database.</p>"
    . "<p>Error code " . mysqli_errno($DBConnect)
    . ": " . mysqli_error($DBConnect)) . "</p>";
echo "<p>Successfully opened the database.</p>";
// additional statements that access the database
mysqli_close($DBConnect);
```

Executing SQL Statements

Use the `mysqli_query()` function to send SQL statements to MySQL

The syntax for the `mysqli_query()` function is:

```
mysqli_query(connection, query)
```

The `mysqli_query()` function returns one of three values:

For SQL statements that do not return results (`CREATE DATABASE` and `CREATE TABLE` statements) it returns a Boolean value of true if the statement executes successfully

Executing SQL Statements

For SQL statements that return results (`SELECT` and `SHOW` statements) the `mysqli_query()` function returns a result pointer that represents the query results

A **result pointer** is a special type of variable that refers to the currently selected row in a resultset (queryset)

The `mysqli_query()` function returns a value of false for any SQL statements that fail, regardless of whether they return results

MySQL query functions

Function	Description
`mysqli_data_seek($Result, position)`	Moves the result pointer to a specified row in the resultset
`mysqli_fetch_array($Result, MYSQLI_ASSOC | MYSQLI_NUM | MYSQLI_BOTH)`	Returns the fields in the current row of a resultset into an indexed array, associative array, or both and moves the result pointer to the next row
`mysqli_fetch_assoc($Result)`	Returns the fields in the current row of a resultset into an associative array and moves the result pointer to the next row
`mysqli_fetch_lengths($Result)`	Returns the field lengths for the current row in a resultset into an indexed array
`mysqli_fetch_row($Result)`	Returns the fields in the current row of a resultset into an indexed array and moves the result pointer to the next row

Retrieving Records into Array

The `mysqli_fetch_row()` function returns the fields in the current row of a resultset into an indexed array and moves the result pointer to the next row

```
echo "<table width='100%' border='1'>";
echo "<tr><th>Make</th><th>Model</th>
    <th>Price</th><th>Quantity</th></tr>";
$Row = mysqli_fetch_row($QueryResult);
do {
    echo "<tr><td>{$Row[0]}</td>";
    echo "<td>{$Row[1]}</td>";
    echo "<td align='right'>{$Row[2]}</td>";
    echo "<td align='right'>{$Row[3]}</td></tr>";
    $Row = mysqli_fetch_row($QueryResult);
} while ($Row);
```

Make	Model	Price	Quantity
Martin	D15 Spruce and Rosewood	1370.00	2
Washburn	D30s	799.99	5
Washburn	D100	329.90	10
Martin	D15 Limited Edition	1138.00	1
Fender	DG11	285.70	8
Martin	DX1 Dreadnought	699.00	9
Fender	DG7	368.20	14
Taylor	Baby Taylor Mahogany	348.00	7
Washburn	D10s	349.99	18
Yamaha	FG720S	279.99	3

Done

Retrieving Records into Array

The `mysqli_fetch_assoc()` function returns the fields in the current row of a resultset into an associative array and moves the result pointer to the next row

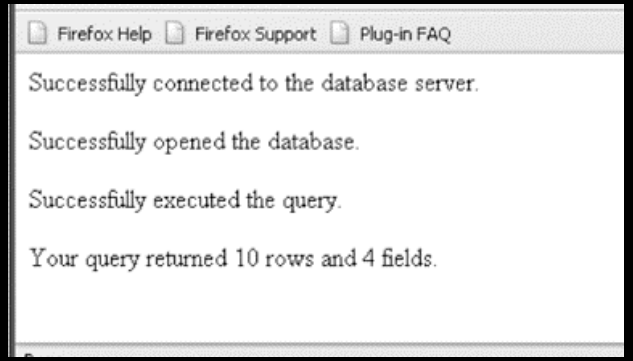
The difference between `mysqli_fetch_assoc()` and `mysqli_fetch_row()` is that instead of returning the fields into an indexed array, `mysqli_fetch_assoc()` function returns the fields into an associative array and uses each field name as the array key

The `mysqli_num_rows()` function returns the number of rows in a query result

The `mysqli_num_fields()` function returns the number of fields in a query result

Retrieving Records into Array

```
$SQLstring = "SELECT * FROM inventory";
$QueryResult = @mysqli_query($DBConnect, $SQLstring)
    Or die("<p>Unable to execute the query.</p>"
    . "<p>Error code " . mysqli_errno($DBConnect)
    . ": " . mysqli_error($DBConnect)) . "</p>";
echo "<p>Successfully executed the query.</p>";
$NumRows = mysqli_num_rows($QueryResult);
$NumFields = mysqli_num_fields($QueryResult);
if ($NumRows != 0 && $NumFields != 0)
    echo "<p>Your query returned" .
mysqli_num_rows($QueryResult)."rows and"
    . mysqli_num_fields($QueryResult). "fields.</p>";
else
    echo "<p>Your query returned no results.</p>";
mysqli_close($DBConnect);
```



Closing Query Results

When you are finished working with query results retrieved with the `mysqli_query()` function, use the `mysqli_free_result()` function to close the resultset

To close the resultset, pass to the `mysqli_free_result()` function the variable containing the result pointer from the `mysqli_query()` function

PHP Delete Database

Deleting a database is almost identical to creating one, except use the `DROP DATABASE` with the `mysqli_query()` function

Use the `mysqli_db_select()` function to check whether a database exists before you create or delete it

```
$DBName = "real_estate";
if (@!mysqli_select_db($DBConnect, $DBName))
    echo "<p>The $DBName database does not exist!</p>";
else {
    $SQLstring = "DROP DATABASE $DBName";
    $QueryResult = @mysqli_query($DBConnect, $SQLstring)
        Or die("<p>Unable to execute the query.</p>"
        . "<p>Error code " . mysqli_errno($DBConnect)
        . ": " . mysqli_error($DBConnect)) . "</p>";
    echo "<p>Successfully deleted the database.</p>";
}
mysqli_close($DBConnect);
```

PHP Create and Delete Tables

```
$DBName = "real_estate";

$SQLstring = "CREATE TABLE commercial (city VARCHAR(25), state
    VARCHAR(25),sale_or_lease VARCHAR(25), type_of_use
    VARCHAR(40),Price INT, size INT)";

$QueryResult = @mysqli_query($DBConnect, $SQLstring)

    Or die("<p>Unable to execute the query.</p>"

    . "<p>Error code " . mysqli_errno($DBConnect)

    . ": " . mysqli_error($DBConnect)) . "</p>";

echo "<p>Successfully created the table.</p>";

mysqli_close($DBConnect);
```

Insert, Delete, and Update Records

To add records to a table, use the `INSERT` and `VALUES` keywords with the `mysqli_query()` function

The values entered in the `VALUES` list must be in the same order in which you defined the table fields

You must specify `NULL` in any fields for which you do not have a value

To add multiple records to a database, use the `LOAD DATA` statement and the `mysqli_query()` function with a local text file containing the records you want to add

To update records in a table, use the `UPDATE`, `SET`, and `WHERE` keywords with the `mysqli_query()` function

Insert, Delete, and Update Records

The `UPDATE` keyword specifies the name of the table to update

The `SET` keyword specifies the value to assign to the fields in the records that match the condition in the `WHERE` keyword

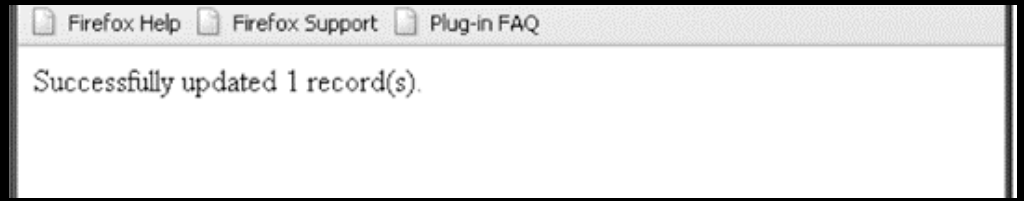
To delete records in a table, use the `DELETE` and `WHERE` keywords with the `mysqli_query()` function. The `WHERE` keyword determines which records to delete in the table

With queries that return results (`SELECT`), use `mysqli_num_rows()` function to find the number of records returned from the query

With queries that modify tables but do not return results (`INSERT`, `UPDATE`, and `DELETE` queries), use `mysqli_affected_rows()` function to determine the number of affected rows

Insert, Delete, and Update Records

```
$SQLstring = "UPDATE inventory SET price=368.20
    WHERE make='Fender' AND model='DG7'";
$QueryResult = @mysqli_query($DBConnect, $SQLstring)
    Or die("<p>Unable to execute the query.</p>"
    . "<p>Error code " . mysqli_errno($DBConnect)
    . ": " . mysqli_error($DBConnect)) . "</p>";
echo "<p>Successfully updated "
    . mysqli_affected_rows($DBConnect) . " record(s).</p>";
```



Insert, Delete, and Update Records

For queries that add or update records, or alter table's structure, use the `mysqli_info()` function to return information about the query

The `mysqli_info()` function returns the number of operations for various types of actions, depending on the type of query

The `mysqli_info()` function returns information about the last query that was executed on the database connection

The `mysqli_info()` function returns information about queries that match one of the following formats:

- `INSERT INTO...SELECT...`
- `INSERT INTO...VALUES (...),(...),(...)`
- `LOAD DATA INFILE ...`
- `ALTER TABLE ...`
- `UPDATE`

Insert, Delete, and Update Records

```
$SQLstring = "INSERT INTO inventory
    VALUES('Ovation', '1777 LX Legend', 1049.00, 2),
    ('Ovation', '1861 Standard Balladeer', 699.00, 1),
    ('Ovation', 'Tangent Series T357', 569.00, 3)";
$QueryResult = @mysqli_query($DBConnect, $SQLstring)
    Or die("<p>Unable to execute the query.</p>"
    . "<p>Error code " . mysqli_errno($DBConnect)
    . ": " . mysqli_error($DBConnect)) . "</p>";
echo "<p>Successfully added the records.</p>";
echo "<p>" . mysqli_info($DBConnect) . "</p>";
```

Successfully added the records.

Records: 3 Duplicates: 0 Warnings: 0

Summary

PHP includes functionality that allows you to work directly with different types of databases, without going through ODBC

Writing code that anticipates and handles potential problems is often called bulletproofing

The error control operator (@) suppresses error messages

A result pointer is a special type of variable that refers to the currently selected row in a resultset (queryset)