```php
<?php
//Prepared statements are very useful against SQL injections.
// Create connection
$conn = mysqli_connect("localhost","root","","tanmay");

// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}
// prepare and bind
$stmt = $conn->prepare("INSERT INTO MyGuests (firstname, lastname, email) VALUES (?, ?, ?)");  //In our SQL, we insert a question mark (?) where we want to substitute in an integer, string, double or blob value.
$stmt->bind_param("sss", $firstname, $lastname, $email);
                //s->string
    // The argument may be one of four types:
      // i - integer
      // d - double
      // s - string
      // b - BLOB
// set parameters and execute
$firstname = "John";
$lastname = "Doe";
$email = "john@example.com";
$stmt->execute();

$firstname = "Mary";
$lastname = "Moe";
$email = "mary@example.com";
$stmt->execute();

$firstname = "Julie";
$lastname = "Dooley";
$email = "julie@example.com";
$stmt->execute();

echo "New records created successfully";

$stmt->close();
$conn->close();
?>
```

1. **Prepare Statement:**
   - $stmt = $conn->prepare("INSERT INTO MyGuests (firstname, lastname, email) VALUES (?, ?, ?)"): This line prepares a statement for inserting data into the MyGuests table. The ? placeholders represent values that will be bound later.
2. **Bind Parameters:**
   - $stmt->bind_param("sss", $firstname, $lastname, $email): This line binds the variables $firstname, $lastname, and $email to the prepared statement. The first argument, "sss", specifies the data types of the bound variables (all strings in this case).
3. **Set Parameters:**
   - $firstname = "John";
   - $lastname = "Doe";
   - $email = "john@example.com";: These lines assign values to the variables that will be used for insertion.
4. **Execute Statement:**
   - $stmt->execute();: This line executes the prepared statement and inserts the provided data into the MyGuests table.