Complete CRUD Operations Notes - MySQLi

CRUD stands for: Create, Read, Update, Delete

C – CREATE (Insert data into database)

1. What is CREATE?

CREATE means **adding new data** to the database.

CREATE is used when we want to **save new information**.

2. Purpose of CREATE

CREATE is used to:

- Add new records to the database
- Save user registration data
- Insert new products, students, or any new information

3. SQL keyword used

The SQL keyword used for CREATE is:

INSERT INTO

4. Basic connection to database (always needed first)

```
<?php
$conn = mysqli_connect("localhost", "root", "", "school");
?>
```

This connects PHP to the MySQL database named "school".

5. Steps to perform CREATE using MySQLi

Step 1: Write the INSERT query

```
$sql = "INSERT INTO students (name, email, age) VALUES ('John', 'john@email.com', 20)";
```

This query means: Insert a new record into the students table.

Step 2: Execute the query

```
$result = mysqli_query($conn, $sql);
```

This sends the query to the database to save the data.

Step 3: Check if insert was successful

```
if ($result) {
    echo "Record inserted successfully";
} else {
    echo "Error inserting record";
}
```

6. Full example of CREATE (MySQLi)

```
<?php
$conn = mysqli_connect("localhost", "root", "", "school");

$sql = "INSERT INTO students (name, email, age) VALUES ('John', 'john@email.com', 20)";
$result = mysqli_query($conn, $sql);

if ($result) {
    echo "Record inserted successfully";
} else {
    echo "Error inserting record";
}
?>
```

7. CREATE using variables (from form)

HTML Form:

```
<form method="POST" action="insert.php">
    <input type="text" name="name" placeholder="Name" required>
```

```
    <input type="email" name="email" placeholder="Email" required>
    <input type="number" name="age" placeholder="Age" required>
    <button type="submit" name="submit">Submit</button>
</form>
```

PHP code (insert.php):

```
 <?php
$conn = mysqli_connect("localhost", "root", "", "school");

if (isset($_POST['submit'])) {
    $name = $_POST['name'];
    $email = $_POST['email'];
    $age = $_POST['age'];

    $sql = "INSERT INTO students (name, email, age) VALUES ('$name', '$email', '$age')";
    $result = mysqli_query($conn, $sql);

    if ($result) {
        echo "Record inserted successfully";
    } else {
        echo "Error inserting record";
    }
}
?>
```

8. Simple explanation for students

CREATE adds new data to the database.

Steps:

1. Connect to database
2. Write INSERT query
3. Execute query
4. Check if successful

9. Key points to remember

- CREATE uses **INSERT INTO** keyword
- `mysqli_query()` executes the query
- Always check if the query was successful
- Column names must match the table structure
- Values must match the data types (text, number, etc.)

10. One-line exam answer

What is CREATE?
CREATE is the operation used to insert new records into the database.

What SQL keyword is used for CREATE?
INSERT INTO

R – READ (Retrieve data from database)

1. What is READ?

READ means **retrieving data** from the database and **displaying it** on the web page.

READ is used when we want to **view saved information**.

2. Purpose of READ

READ is used to:

- Display records stored in the database
- View user information
- Show lists such as students, products, or users

3. SQL keyword used

The SQL keyword used for READ is:

SELECT

4. Basic connection to database (always needed first)

```
$conn = mysqli_connect("localhost", "root", "", "school");
```

This connects PHP to the MySQL database named "school".

5. Method 1: READ using while loop

Steps:

Step 1: Write the SELECT query

```
$sql = "SELECT * FROM students";
```

This query selects all records from the students table.

Step 2: Execute the query

```
$result = mysqli_query($conn, $sql);
```

The database processes the query and returns the result.

Step 3: Fetch and display data using while loop

```
while ($row = mysqli_fetch_assoc($result)) {
    echo $row['name'] . "<br>";
}
```

- `mysqli_fetch_assoc()` gets **one record at a time**
- The loop continues until all records are displayed

Full example (while loop):

```
<?php
$conn = mysqli_connect("localhost", "root", "", "school");

$sql = "SELECT * FROM students";
$result = mysqli_query($conn, $sql);

while ($row = mysqli_fetch_assoc($result)) {
    echo $row['name'] . "<br>";
}
?>
```

6. Method 2: READ using foreach loop

Steps:

Step 1: Write the SELECT query

```
$sql = "SELECT * FROM students";
```

Step 2: Execute the query

```
$result = mysqli_query($conn, $sql);
```

Step 3: Convert result to an array

```
$data = mysqli_fetch_all($result, MYSQLI_ASSOC);
```

- `mysqli_fetch_all()` gets **all records at once**
- Stores them in an array

Step 4: Display data using foreach

```
foreach ($data as $row) {
    echo $row['name'] . "<br>";
}
```

- `foreach` loops through each record in the array
- Each loop displays one record

Full example (foreach loop):

```
<?php
$conn = mysqli_connect("localhost", "root", "", "school");

$sql = "SELECT * FROM students";
$result = mysqli_query($conn, $sql);

$data = mysqli_fetch_all($result, MYSQLI_ASSOC);

foreach ($data as $row) {
    echo $row['name'] . "<br>";
}
?>
```

7. What is MYSQLI_ASSOC?

`MYSQLI_ASSOC` is a constant that tells PHP **how to organize** data from the database.

Where it is used:

```
$data = mysqli_fetch_all($result, MYSQLI_ASSOC);
```

What it does:

It returns data as an **associative array**, meaning:

- You can access data using **column names**
- You don't need to remember numbers

Example WITHOUT MYSQLI_ASSOC:

Data looks like this:

```
$row[0] = 1;
$row[1] = 'John';
$row[2] = 20;
```

To display name:

```
echo $row[1];  // Confusing! What is 1?
```

Example WITH MYSQLI_ASSOC:

Data looks like this:

```
$row['id'] = 1;
$row['name'] = 'John';
$row['age'] = 20;
```

To display name:

```
echo $row['name'];  // Clear! We know it's the name
```

Why we use MYSQLI_ASSOC:

Without MYSQLI_ASSOC	With MYSQLI_ASSOC
Access using numbers	Access using names
Hard to understand	Easy to understand

Easy to make mistakes	Clear and safe

8. Comparison: while vs foreach

Feature	while loop	foreach loop
Fetches data	One record at a time	All records at once
Function used	mysqli_fetch_assoc()	mysqli_fetch_all()
When to use	Small data	Large data lists
Speed	Slightly slower	Slightly faster

9. Simple explanation for students

Using while:

- PHP asks the database for one record
- Displays it
- Asks for the next record
- Repeats until no records are left

Using foreach:

- PHP asks the database for all records at once
- Stores them in an array
- Loops through the array and displays each record

10. Key points to remember

- READ uses **SELECT** keyword
- `mysqli_query()` executes the query
- `mysqli_fetch_assoc()` fetches one record at a time (for while)
- `mysqli_fetch_all()` fetches all records at once (for foreach)
- **MYSQLI_ASSOC** is used to access data using column names
- `while` loop is good for small data
- `foreach` loop is good for displaying lists

11. One-line exam answers

What is READ?
READ is the operation used to retrieve and display data from the database.

What SQL keyword is used for READ?
SELECT

What is MYSQLI_ASSOC?
MYSQLI_ASSOC is used to fetch database results as an associative array so that data can be accessed using column names.

D – DELETE (Remove data from database)

1. What is DELETE?

DELETE means **removing data** from the database permanently.

DELETE is used when we want to **delete records** that are no longer needed.

2. Purpose of DELETE

DELETE is used to:

- Remove unwanted records
- Delete user accounts
- Remove old or incorrect data

3. SQL keyword used

The SQL keyword used for DELETE is:

DELETE

4. Basic connection to database (always needed first)

```
$conn = mysqli_connect("localhost", "root", "", "school");
```

This connects PHP to the MySQL database named "school".

5. Steps to perform DELETE using MySQLi

Step 1: Write the DELETE query

```
$sql = "DELETE FROM students WHERE id = 5";
```

This query means: Delete the record from students table where id is 5.

Step 2: Execute the query

```
$result = mysqli_query($conn, $sql);
```

This sends the query to the database to delete the record.

Step 3: Check if delete was successful

```
if ($result) {
    echo "Record deleted successfully";
} else {
    echo "Error deleting record";
}
```

This checks whether the delete operation worked or not.

6. Full example of DELETE (MySQLi)

```
<?php
$conn = mysqli_connect("localhost", "root", "", "school");

$sql = "DELETE FROM students WHERE id = 5";
$result = mysqli_query($conn, $sql);

if ($result) {
    echo "Record deleted successfully";
} else {
    echo "Error deleting record";
}
?>
```

7. DELETE using variable (from form or URL)

Example: Delete using ID from URL

```php
<?php
$conn = mysqli_connect("localhost", "root", "", "school");

$id = $_GET['id'];  // Get ID from URL

$sql = "DELETE FROM students WHERE id = $id";
$result = mysqli_query($conn, $sql);

if ($result) {
    echo "Record deleted successfully";
} else {
    echo "Error deleting record";
}
?>
```

How it works:

If the URL is:

```
delete.php?id=5
```

Then `$_GET['id']` will be 5, and the record with id 5 will be deleted.

8. DELETE with confirmation (safer method)

It's good practice to ask the user to confirm before deleting.

HTML form with delete button:

```php
<form method="POST" action="delete.php">
    <input type="hidden" name="id" value="5">
    <button type="submit" name="delete">Delete</button>
</form>
```

PHP code to handle delete:

```php
<?php
$conn = mysqli_connect("localhost", "root", "", "school");

if (isset($_POST['delete'])) {
    $id = $_POST['id'];

    $sql = "DELETE FROM students WHERE id = $id";
    $result = mysqli_query($conn, $sql);
```

```
    if ($result) {
        echo "Record deleted successfully";
    } else {
        echo "Error deleting record";
    }
}
?>
```

9. DELETE all records (DANGEROUS - use carefully)

```
$sql = "DELETE FROM students";
```

This deletes **ALL records** from the students table.

> **Warning:** This cannot be undone!

10. Important points about WHERE clause

With WHERE clause:

```
DELETE FROM students WHERE id = 5;
```

- Deletes **only one specific record** (where id is 5)

Without WHERE clause:

```
DELETE FROM students;
```

- Deletes **ALL records** from the table
- Very dangerous!

> **Rule:** Always use WHERE clause unless you really want to delete everything.

11. How to check how many records were deleted

```
$affected_rows = mysqli_affected_rows($conn);
echo "Records deleted: " . $affected_rows;
```

This tells you how many records were actually deleted.

Full example:

```
 <?php
$conn = mysqli_connect("localhost", "root", "", "school");

$sql = "DELETE FROM students WHERE age < 18";
$result = mysqli_query($conn, $sql);

$affected_rows = mysqli_affected_rows($conn);

if ($result) {
    echo "Records deleted: " . $affected_rows;
} else {
    echo "Error deleting records";
}
?>
```

12. Complete example: Display and Delete

This example shows all students with a delete button for each:

```
 <?php
$conn = mysqli_connect("localhost", "root", "", "school");

// Handle delete
if (isset($_GET['delete_id'])) {
    $id = $_GET['delete_id'];
    $sql = "DELETE FROM students WHERE id = $id";
    mysqli_query($conn, $sql);
    echo "Record deleted!<br>";
}

// Display all students
$sql = "SELECT * FROM students";
$result = mysqli_query($conn, $sql);

while ($row = mysqli_fetch_assoc($result)) {
    echo $row['name'] . " - ";
    echo "<a href='?delete_id=" . $row['id'] . "'>Delete</a><br>";
}
?>
```

13. Simple explanation for students

DELETE removes data from the database permanently.

Steps:

1. Connect to database
2. Write DELETE query with WHERE clause
3. Execute query
4. Check if successful

The WHERE clause tells which record to delete.

Without WHERE clause, all records will be deleted!

14. Key points to remember

- DELETE is used to remove records
- Always use WHERE clause to specify which record to delete
- Without WHERE, all records are deleted
- DELETE is permanent and cannot be undone
- Use `mysqli_affected_rows()` to check how many records were deleted
- Always confirm before deleting important data

15. Common mistakes to avoid

Mistake 1: Forgetting WHERE clause

```
DELETE FROM students;  // Deletes everything!
```

Fix:

```
DELETE FROM students WHERE id = 5;  // Deletes only id 5
```

Mistake 2: Not checking if delete was successful

Fix:

```
if ($result) {
   echo "Deleted successfully";
}
```

Mistake 3: Not sanitizing input from user

Better code:

```
$id = mysqli_real_escape_string($conn, $_GET['id']);
$sql = "DELETE FROM students WHERE id = '$id'";
```

16. One-line exam answers

What is DELETE?
DELETE is the operation used to remove data permanently from the database.

What SQL keyword is used for DELETE?
DELETE

Why do we use WHERE clause in DELETE?
WHERE clause is used to specify which record to delete. Without it, all records will be deleted.

What function checks how many records were deleted?
mysqli_affected_rows()

CRUD Operations Summary

Operation	SQL Keyword	Purpose	Main Function
Create	INSERT INTO	Add new data	mysqli_query()
Read	SELECT	Retrieve and display data	mysqli_fetch_assoc() or mysqli_fetch_all()
Update	UPDATE	Modify existing data	mysqli_query()
Delete	DELETE	Remove data	mysqli_query()

General Connection Code (Used for all operations)

```php
 <?php
$conn = mysqli_connect("localhost", "root", "", "database_name");

if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
?>
```

Important Notes

> - Always connect to the database first
> - Always check if queries were successful
> - Use WHERE clause in UPDATE and DELETE to avoid changing/deleting all records
> - Use MYSQLI_ASSOC to access data with column names
> - Sanitize user input to prevent SQL injection
> - Close database connection when done: `mysqli_close($conn);`

Complete CRUD Operations Notes - MySQLi

For Educational Purposes