Module Seventeen Assignment

12.06.2023

—

Quazi Tariqul Alam

PHP/LARAVEL Batch 1

Narayanganj

1.Explain what Laravel's query builder is and how it provides a simple and elegant way to interact with databases.

Answer: Laravel’s query builder provides us with a convenient and simple interface to create and run database queries. It has lots of easy to use functions and flexible methods that can run complex queries behind the scene to interact with the database. Most of the time, for example, you will understand what the function is doing just by reading the function name and guess it.

2.Write the code to retrieve the "excerpt" and "description" columns from the "posts" table using Laravel's query builder. Store the result in the $posts variable. Print the $posts variable.

Answer:

```
$posts = DB::table(‘posts’)

        ->select(‘excerpt’, ‘description’)

        ->get();
```

3.Describe the purpose of the distinct() method in Laravel's query builder. How is it used in conjunction with the select() method?

Answer: The distinct() method is used to retrieve unique records from a database table. It ensures that duplicate records are removed. It is typically used with select() method to select only unique values from database. For example, the following code only select unique name from the name column of users tabel:

```
$uniqueNames = DB::table('users')

                ->select('name')

                ->distinct()

                ->get();
```

4.Write the code to retrieve the first record from the "posts" table where the "id" is 2 using Laravel's query builder. Store the result in the $posts variable. Print the "description" column of the $posts variable.

Answer:

```
$posts = DB::table('posts')

        ->where('id', 2)

        ->first();

echo $posts->description;
```

5.Write the code to retrieve the "description" column from the "posts" table where the "id" is 2 using Laravel's query builder. Store the result in the $posts variable. Print the $posts variable.

Answer:

```
$posts = DB::table('posts')

            ->where('id', 2)

            ->pluck('description');

echo $posts;
```

6.Explain the difference between the first() and find() methods in Laravel's query builder. How are they used to retrieve single records?

Answer:

The find method retrieves a single record by its primary key. For example:

```
$user = DB::table('users')->find(1);
```

On the other hand, the first method retrieves the first record from the table. For example:

```
$user = DB::table('users')

            ->where('active', 1)

            ->first();
```

7.Write the code to retrieve the "title" column from the "posts" table using Laravel's query builder. Store the result in the $posts variable. Print the $posts variable.

Answer:

```
$posts = DB::table('posts')

            ->select('title')

            ->get();

            echo $posts;
```

8.Write the code to insert a new record into the "posts" table using Laravel's query builder. Set the "title" and "slug" columns to 'X', and the "excerpt" and "description" columns to 'excerpt' and 'description', respectively. Set the "is_published" column to true and the "min_to_read" column to 2. Print the result of the insert operation.

Answer:

```
$result = DB::table('posts')->insertGetId([
    'title' => 'X',
    'slug' => 'X',
    'excerpt' => 'excerpt',
    'description' => 'description',
    'is_published' => true,
    'min_to_read' => 2
]);
echo $result;
```

9.Write the code to update the "excerpt" and "description" columns of the record with the "id" of 2 in the "posts" table using Laravel's query builder. Set the new values to 'Laravel 10'. Print the number of affected rows.

Answer:

```
$affectedRows = DB::table('posts')
        ->where('id', 2)
        ->update([
        'excerpt' => 'Laravel 10',
        'description' => 'Laravel 10'
        ]);
echo $affectedRows . " row(s) affected.";
```

10.Write the code to delete the record with the "id" of 3 from the "posts" table using Laravel's query builder. Print the number of affected rows.

Answer:

```
$affectedRows = DB::table('posts')

    ->where('id', 3)

    ->delete();

echo $affectedRows . " row(s) affected.";
```

11.Explain the purpose and usage of the aggregate methods count(), sum(), avg(), max(), and min() in Laravel's query builder. Provide an example of each.

Answer:

The `count()` method in Laravel's query builder is used to retrieve the total number of rows that match a given query.

```
$totalUsers = DB::table('users')->count();

echo "Total number of users: " . $totalUsers;
```

The sum() method in Laravel's query builder is used to calculate the sum of a specific column's values in a database table.

```
$totalSales = DB::table('orders')->sum('amount');

echo "Total sales amount: " . $totalSales;
```

The avg() method in Laravel's query builder is used to calculate the average of a specific column's values in a database table.

```
$avgSales = DB::table('orders')->avg('amount');

echo "The average sales amount: " . $avgSales;
```

The max() method in Laravel's query builder is used to return the highest value of a specific column's values in a database table.

```
$maxSales = DB::table('orders')->max('amount');

echo "The highest sales amount: " . $maxlSales;
```

The min() method in Laravel's query builder is used to return the lowest value of a specific column's values in a database table.

```
$minSales = DB::table('orders')->min('amount');

echo "The lowest sales amount: " . $maxlSales;
```

12.Describe how the whereNot() method is used in Laravel's query builder. Provide an example of its usage.

Answer:

This method allows you to specify a condition that must not be met for the records to be included in the result set. Following example exclude all the users who are inactive from the result returned.

```
$users = DB::table('users')
        ->whereNot('status', 'inactive')
        ->get();
```

13.Explain the difference between the exists() and doesntExist() methods in Laravel's query builder. How are they used to check the existence of records?

Answer:

The exists method checks if any records exist in the table that match a specified condition.

```
$hasAdmins = DB::table('users')
        ->where('role', 'admin')
        ->exists();
if ($hasAdmins) {
        echo "There are admins in the system.";
} else {
        echo "No admins found.";
}
```

The doesntExist method checks if no records exist in the table that match a specified condition.

```
$noInactiveUsers = DB::table('users')
        ->where('status', 'inactive')
        ->doesntExist();
if ($noInactiveUsers) {
        echo "There are no inactive users.";
} else {
        echo "Inactive users found.";

}
```

14.Write the code to retrieve records from the "posts" table where the "min_to_read" column is between 1 and 5 using Laravel's query builder. Store the result in the $posts variable. Print the $posts variable.

Answer:

```
$posts = DB::table('posts')

        ->whereBetween('min_to_read', [1, 5])

        ->get();

print_r($posts);
```

15.Write the code to increment the "min_to_read" column value of the record with the "id" of 3 in the "posts" table by 1 using Laravel's query builder. Print the number of affected rows.

Answer:

```
$rows = DB::table('posts')

           ->where('id', 3)

           ->increment('min_to_read', 1);

echo $rows . " row(s) affected.";
```