Seventeen Module Assignment

Questions:

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

Ans: Laravel's query builder is a set of classes and methods that provide a simple and elegant way to interact with databases. Here some example of method is select, where, join, groupby, having, first(), value(), put(), find(), pluck(), join, insert, update, delete, insertOrIgnore(), insertGetId() etc.

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.**

Ans:
```
$posts = DB::table('posts')
    ->select('excerpt', 'description')
    ->get();
return $posts;
```

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

Ans: The distinct() method in Laravel's query builder is used to retrieve only unique values from a specific column in the database table.

Example:
```
$distinctValues = DB::table('posts')
    ->select('title')
    ->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.

Ans: $post = DB::table('posts')

->where('id', 2)

->first();

return $post->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.

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

->select('description')

->find(2);

return $posts;

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

Ans: In Laravel's query builder, the first() and find() methods are used to retrieve single records from the database. first() method is used to retrieve the first record that matches the query criteria. find() method is used to retrieve a record by its primary key value. Both method will return null values if no matching record is found.

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.**

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

->select('title')

```
        ->get();

        return $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.**

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

    return $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.**

Ans:
```
$UpdatedRows = DB::table('posts')
        ->where('id', 2)
        ->update([
            'excerpt' => 'Laravel 10',
            'description' => 'Laravel 10',
        ]);
```

```
echo "Number of Updated rows: " . $UpdatedRows;
```

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.

Ans:

```
$DeletedRows = DB::table('posts')
    ->where('id', 3)
    ->delete();

echo "Number of Deleted rows: " . $DeletedRows;
```

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.**

Ans: the aggregate methods are used to perform calculations on database columns and retrieve aggregate values from the result set.

Example:-

```
$totalUsers = DB::table('users')->count();
$totalSales = DB::table('products')->sum('stock');
$averageRating = DB::table('products')->avg('price');
$highestScore = DB::table('products')->max('score');
$lowestPrice = DB::table('products')->min('price');
```

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

Ans: whereNot() method is used to add a "WHERE NOT" condition to a query. It allows you to specify a column and a value that should not match in order to include a record in the result set. This method is useful when you want to exclude specific records based on certain criteria.

Example:-

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

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?**

Ans: Mentions methods are used to check the existence of records in a table.

exists(): This method is used to check if records exist in the result set of a query.

It returns true if at least one record is found, and false if no records are found.

doesntExist(): This method is used to check if no records exist in the result set of a query.

It returns true if no records are found, and false if at least one record is found.

Example:-

```
return $exists = DB::table('products')->where('price', 2000)->exists();
return $doesntExist = DB::table('products')->where('price', 8000)->doesntExist();
```

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.**

Ans:
```
$posts = DB::table('posts')
    ->whereBetween('min_to_read', [1, 5])
    ->get();
return $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.**

Ans:
```
$affectedRows = DB::table('posts')
    ->where('id', 3)
    ->increment('min_to_read', 1);

return "Number of affected rows: " . $affectedRows;
```