*** Assignment-17 ***

Question -1 :

Answer : Laravel's query builder is a fluent and expressive interface provided by Laravel framework to interact with databases. It allows developers to build database queries using a chainable set of methods, providing a more readable and intuitive syntax compared to writing raw SQL queries. The query builder abstracts the underlying database system, allowing developers to write database-agnostic code.

Question -2 :

Answer :

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


print_r($posts);
```

Question -3 :

Answer :

```
$users = DB::table('users')
          ->select('name')
          ->distinct()
          ->get();
```

Question -4 :

Answer :

```
$post = DB::table('posts')
          ->where('id', 2)
          ->first();


if ($post) {
   echo $post->description;
}
```

Question -5 :

Answer :

```
$posts = DB::table('posts')
          ->where('id', 2)
          ->pluck('description');

print_r($posts);
```

Question -6 :

Answer :

The first() and find() methods in Laravel's query builder are used to retrieve single records, but they have some differences in their usage:

The first() method retrieves the first record that matches the specified conditions from the database table. It is commonly used when you want to retrieve a single record based on certain criteria, such as retrieving the first user with a specific role. The first() method returns a single object representing the record or null if no record is found.

The find() method retrieves a record by its primary key value. It is specifically designed to fetch a record based on its primary key. The find() method returns a single object representing the record or null if no record is found.

Question -7 :

Answer :

```
$posts = DB::table('posts')
          ->pluck('title');

print_r($posts);
```

Question -8 :

Answer :

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

echo $result;
```

Question -9 :

Answer :

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

echo $affectedRows;
```

Question -10 :

Answer :

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

echo $affectedRows;
```

Question -11 :

Answer : In Laravel's query builder, the aggregate methods count(), sum(), avg(), max(), and min() are used to perform aggregate operations on a column or a set of columns in a database table. Here are the explanations and examples for each:

count(): It returns the number of records in a table or the number of records that match a specific condition.

```
  $totalUsers = DB::table('users')->count();
$activeUsers = DB::table('users')->where('active', true)->count();
```

sum(): It returns the sum of the values in a column.

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

avg(): It returns the average value of a column.

```
$averageRating = DB::table('reviews')->avg('rating');
```

max(): It returns the maximum value from a column.

```
$maxPrice = DB::table('products')->max('price');
```

min(): It returns the minimum value from a column.

```
$minPrice = DB::table('products')->min('price');
```

Question -12 :

Answer : The whereNot() method in Laravel's query builder is used to add a "where not" condition to a query. It allows you to retrieve records where a column's value is not equal to a specific value.

Here's an example of how the whereNot() method is used:

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

In this example, the query retrieves all users from the "users" table where the "status" column is not equal to 'active'. The whereNot() method adds the "where not" condition to the query

----

Question -13 :

Answer : The exists() and doesntExist() methods in Laravel's query builder are used to check the existence of records:

exists(): It checks if any records exist that match the specified conditions. It returns true if at least one record is found; otherwise, it returns false.

doesntExist(): It checks if no records exist that match the specified conditions. It returns true if no record is found; otherwise, it returns false.

These methods are useful when you want to determine if records exist before performing certain actions or conditions based on their existence

----

Question -14 :

Answer :

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

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

        ->get();


print_r($posts);
```

Question -15 :

Answer :

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

          ->where('id', 3)

          ->increment('min_to_read');


echo $affectedRows;
```
