**Question** 1

Laravel's query builder provides a simple and elegant way to interact with databases. It allows you to build and run database queries using a fluent, chainable interface.

**Question** 2

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

print_r($posts);

**Question** 3

The `distinct()` method in Laravel's query builder is used to force the query to return distinct results. It is often used in conjunction with the `select()` method to specify the columns that should be returned by the query.

**Question** 4

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

echo $post->description

**Question** 5

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

echo $description

**Question** 6

The `first()` and `find()` methods in Laravel's query builder are both used to retrieve single records from a database table. The `first()` method returns the first record that matches the specified conditions, while the `find()` method retrieves a record by its primary key.

**Question** 7

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

print_r($titles);
```

**Question** 8

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

**Question** 9

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

```
        ->where('id', 2)
        ->update(['excerpt' => 'Laravel 10', 'description' => 'Laravel 10']);
echo $affected
```

**Question** 10

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

**Question** 11

The aggregate methods `count()`, `sum()`, `avg()`, `max()`, and `min()` in Laravel's query builder are used to perform aggregate operations on a database table. The `count()` method returns the number of records that match a given set of conditions, while `sum()` returns the sum of a given column for all records that match a given set of conditions. The `avg()` method returns the average value of a given column for all records that match a given set of conditions, while `max()` and `min()` return the maximum and minimum values of a given column for all records that match a given set of conditions.

**Question** 12

The `whereNot()` method in Laravel's query builder adds a basic where clause to a query that negates its condition. For example, here is how you would use it to retrieve all users whose name is not John:

Here is  an example:

```
$users = DB::table('users')
   ->whereNot('name', 'John')
   ->get();
echo "Result: ";
```

```
print_r($users);
```

**Question** 13

"The `exists()` and `doesntExist()` methods in Laravel's query builder are used to determine if any records exist that match a given set of conditions. The `exists()` method returns `true` if any records match the conditions, while `doesntExist()` returns `true` if no records match the conditions.

**Question** 14

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

**Question** 15

```
$affected = DB::table('posts')
     ->where('id', 3)
     ->increment('min_to_read');
echo $affected
```