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

Soln: In Laravel the database query builder provides an easy interface to create and run database queries. It can be used to perform all the database operations in our application, from basic database Connection, CRUD, Aggregates, etc. and it works on all supported database systems.

The notable factor about query builder is that, since it uses the PHP Data Objects (PDO), we need not worry about SQL injection attacks. We can avoid all those lines of code to sanitize the data before feeding it to the database.

The DB class is a facade that provides a convenient and expressive interface to interact with the database using the Laravel Query Builder. It allows us to perform database operations such as querying data, inserting records, updating records, deleting records, and executing raw SQL statements.

We create a simple select query to fetch all values from the students table using query builder in the following way,

```
$students = DB::table('students')->get();
```

DB::table is responsible to begin a fluent query against a database table. The table from which the value has to be selected is mentioned inside the brackets within quotes and finally the get() method gets the values. Similarly to fetch a single row we can modify the above code by adding a where clause in the following manner,

```
$students = DB::table('students')->where('name', 'Anik')->first();
```

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.

Soln:

```
class demoController extends Controller
{
    function question2(Request $r)
    {
        $posts= DB::table('posts')->select('excerpt','description')->get();
        print_r($posts->toArray());

    }
}
```

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

Soln: The distinct( ) method is used to fetch distinct records from the database, it is also part of Laravel query builder, which means it can be chained to other query builder methods as well. This method can be used on a posts data table to fetch distinct records in the following manner,

```
class demoController extends Controller
{
    function question3(Request $r)
    {
        $user_ids= DB::table('posts')->distinct()->get();
        print_r($user_ids->toArray());

    }}
```

The select() method can be used along distinct() method . For example we can get all the distinct user_id from the posts table by chaining the select() and distinct() methods. First we select all the user_id from the posts table then we use distinct method to take the unique user_id's.

```
class demoController extends Controller
{
    function question3(Request $r)
    {
        $user_ids= DB::table('posts')->select('user_id')->distinct()->get();
        print_r($user_ids->toArray());
    }
}
```

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.

Soln:

```
class demoController extends Controller
{
    function question4(Request $r)
    {
        $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.

Soln:

```
class demoController extends Controller
{
    function question5(Request $r)
    {
   $posts= DB::table('posts')->select('description')->where('id','=',2)->get();
        print_r($posts->toArray());
    }
}
```

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

Soln:

first()	find()
1.The **first()** method retrieves the first record that matches the query criteria.	The **find()** method retrieves a record by its primary key value.
2. It returns a single model instance or a null value if no matching record is found.	It returns a single model instance if a matching record is found, and it returns null if no record with the specified primary key is found
3. It is commonly used when you expect a single record to be returned or when you want to retrieve the first record that satisfies your query conditions.	It is typically used when you want to retrieve a record based on its unique identifier (primary key).

The **find()** method can be used on a posts table to find a post with a perticular id in the following manner,

```
$post= DB::table('posts')->find(2);
```

This query returns a single record with id value of 2.
Similary we can use **first()** method on a posts table to retrieve the first record that has a particular user_id in the following manner,

```
$posts= DB::table('posts')->where('user_id','=',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.

Soln:

```
class demoController extends Controller
{
    function question7(Request $r)
    {
        $posts= DB::table('posts')->pluck('title');
        print_r($posts->toArray());
    }
}
```