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

Ans: The Laravel Query Builder is a component of the Laravel framework that provides an easy and fluent interface for constructing database queries. It enables us to construct database queries using a chainable, object-oriented syntax rather than plain SQL queries.

The Laravel Query Builder provides a simple and elegant way to interact with databases through its intuitive and expressive API. Like:

1. The Query Builder has a number of techniques that roughly mimic SQL statements. The method names and arguments are intended to be straightforward, making it simple to comprehend and construct queries.
2. The Query Builder interface is fluent. This method makes it simple to design complicated searches by sequentially adding clauses and conditions, boosting code readability and maintainability.
3. We can use the any database system as Laravel's database connection configurations handle the differences between database engines transparently.

Ques: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:

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

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

The distinct() method used to fetch only unique values for the selected columns in the database. This is helpful in scenarios where we want to fetch distinct values, such as unique names, email addresses, or any other field, from a table.

In Laravel's query builder, the distinct() function is typically used in combination with the select() method to define which columns should be fetched for uniqueness. Here is an example:

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

In the above example, we first use the select() method to specify that we are going to retrieve only the email column and then using distinct() method to ensure that only unique email values are returned.

Ques4. 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: The code is given below:

```
$posts = DB::table('posts')
        ->where('id', 2)
        ->first();
if($posts){
        echo $posts->description;
}else{
        Echo "No post found";
}
```

Ques 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:

The code is given below:

$posts = DB::table(‘posts’)

->where(‘id’, 2)

->pluck(‘description’);

Print_r($posts);

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

Ans:

The difference between first() and find() method are given below:

The first() method retrieves the first record that matches the query criteria. It is used when we want to retrieve the first result of a query. On the other hand, find() method is used to retrieve a record by its primary key value. It is particularly useful when we know the primary key value of the record.

Example of using the methods to retrieve single records are:

$user = DB::table(‘users’)->first();

$user = DB::table(‘users’)->find(1);

Ques: 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:

The code is given below:

$posts = DB::table(‘posts’)

->pluck(‘title’);

print_r($posts);

Ques: 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:

The code is given below:

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

Ques: 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:

The code is given below:

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

Ques: 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:

The code is given below:

$rows = DB::table(‘ posts’)

->where(‘id’, 3)

->delete();

echo “ Number of affected rows: “. $rows;

Ques: 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 in Laravel’ s query builder are used to perform calculations on columns in database tables. The example are given below:

1. The count() method is used to retrieve the number of records that match a query. Like
   $totalEmployee = DB::table(‘Employee’)
   ->count();

2. The sum() method is used to calculate the sum of a column’s values. Like:

   $totalSalary = DB::table(‘Employee’)

   ->sum(‘Salary’);

3. The avg() method is used to calculate the average value of a column. Like:

   $avgSalary = DB::table(‘Employee’)

   ->avg(‘Salary’);

4. The max() method is used to retrieve the maximum value of a column. Like:

   $maxSalary = DB::table(‘Employee’)

   ->max(‘Salary’);

5. The min() method is used to retrieve the minimum value of a column.

   $minSalary = DB::table(‘Employee’)

   ->min(‘Salary’);

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

Ans:

The whereNot() method in Laravel's query builder is used to add a "not" condition to a query. It allows you to retrieve records that do not meet the specified condition. The 'whereNot()' method can be used with various operators such as "=", "<", ">" etc.

The example is given below:

For a product which price is not equal to 100 is:

```
$product = DB::table('products')
            ->whereNot('price', '=', 100)
            ->get();
```

Ques: 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?

The exists() and doesntExist() methods are used to check the existence of records in a table.

The exists() method is used to check if any records exist that match the given query. On the otherhand, the doesntExist() method is used to check if no records exist that match the given query.

Ques: 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:

The code is given below:

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

Ques: 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:

The code is given below:

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