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

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.

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

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.

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.

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

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.

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.

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.

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.

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.

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

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?

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.

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.

Question 1:

Answer: A query builder is a tool or API that allows you to write and execute queries without writing any database queries. Laravel Query Builder is built using PHP PDO. Laravel uses DB Facade to create queries

Question 2:

Answer:

$post = DB::table(‘posts’)

->select('excerpt', 'description')->get();

print\_r(‘$posts’);

Question 3:

Answer: The distinct method allows you to force the query to return distinct results:

$uMail = DB::table('users')

->select('email')->distinct()->get();

Question 4:

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

->where('id', 2)

->first();

echo $posts->description;

Question 5:

Answer:

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

->where('id', 2)

->value('description');

Print\_r $posts;

Question 6:

Answer:

In Laravel's query builder, both the first() and find() methods are used to retrieve a single record from the database. However, they differ in their approach and usage: first(): The first() method is used to retrieve the first record that matches the query criteria. It returns a single object representing the first matching row. $record = DB::table('users') ->where('status', 'active') ->first();

find(): The find() method is used to retrieve a single record based on its primary key value. It accepts the primary key value as an argument and returns a single object representing the matching row. $record = DB::table('users')->find(1);

Question 7:

Answer:

$posts = DB::table('posts') ->select('title') ->get();

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, ]);

print\_r($result);

Question 9:

Answer:

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

Question 10 Answer:

$affectedRows = DB::table('posts') ->where('id', 3) ->delete(); echo "Number of affected rows: " . $affectedRows;

Question 11. Answer:

count(): The count() method is used to retrieve the number of rows matching the query criteria. $count = DB::table('users')->count(); In this example, the count() method retrieves the total number of rows in the "users" table

sum(): The sum() method is used to calculate the sum of the values in a specific column. $total = DB::table('sales')->sum('amount'); In this example, the sum('amount') method calculates the total sum of the "amount" column in the "sales" table.

avg(): The avg() method is used to calculate the average value of a specific column. $average = DB::table('products')->avg('price'); In this example, the avg('price') method calculates the average price of the products stored in the "products" table.

max(): The max() method is used to retrieve the maximum value from a specific column. $maximum = DB::table('scores')->max('points'); In this example, the max('points') method retrieves the highest score (maximum value) from the "points" column in the "scores" table.

min(): The min() method is used to retrieve the minimum value from a specific column. $minimum = DB::table('temperatures')->min('degrees'); In this example, the min('degrees') method retrieves the lowest temperature (minimum value) from the "degrees" column in the "temperatures" table.

12

Answer: In Laravel's query builder, the whereNot() method is used to add a "not" condition to the query. It allows you to exclude rows that match a specific condition from the result set. The whereNot() method takes two arguments: the column name and the value to compare against. It adds a WHERE condition to the query, stating that the column value should not be equal to the provided value. Here's an example to illustrate the usage of

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

13)

Answer: exists(): The exists() method is used to check if any records exist in the result set of a query. $exists = DB::table('users') ->where('status', 'active') ->exists(); doesntExist(): The doesntExist() method is used to check if no records exist in the result set of a query.

$doesntExist = DB::table('users') ->where('status', 'active') ->doesntExist();

14) Answer: $posts = DB::table('posts') ->whereBetween('min\_to\_read', [1, 5]) ->get(); print\_r($posts);

15) Answer: $affectedRows = DB::table('posts') ->where('id', 3) ->increment('min\_to\_read', 1); echo "Number of affected rows: " . $affectedRows;