Module 17 - Assignment answers

1. The database query builder provides a convenient, fluent interface to creating and running database queries. It can be used to perform most database operations in your application, and works on all supported database systems.
   - It's Allows to interact with your database
   - Provides a wide range of methods
   - Highly flexible easy to use
   - Used to perform complex queries
   - Well-documented

2. use Illuminate\Support\Facades\DB;
   $posts = DB::table('posts')->select('excerpt', 'description')->get();
   print_r($posts);

3. In Laravel the distinct() method is used to retrieve a query result set with unique records.
   The distinct() method is often used in conjunction with the select() method to specify the columns that should be considered when determining uniqueness.
   Examle:
   ```
   $posts = DB::table('posts')
     ->select('title', 'slug')
     ->distinct()
     ->get();
   ```

4. $posts = DB::table('posts')->where('id', 2)->first();
   echo $posts->description;

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

6. In Laravel's query builder, the first() and find() methods are both used to retrieve single records from a database table, but they differ in their behavior and usage. In Laravel's query builder, the first() and find() methods are both used to retrieve single records from a database table, but they differ in their behavior and usage.
   Example of first() method: The first() method retrieves the first record that matches the query criteria.
   $user = DB::table('users')->where('active', true)->first();

   Example of find() method: The find() method retrieves a single record by its primary key value.
   $user = DB::table('users')->find(1);

7. use Illuminate\Support\Facades\DB;
   $posts = DB::table('posts')->select('title')->get();
   print_r($posts);

8. use Illuminate\Support\Facades\DB;
   DB::table('posts')->insert([
   'title' => 'Ostad module 17',
   'slug' => 'ostad-module-17',
   'excerpt' => 'excerpt',
   'description' => 'description',
   'is_published' => true,
   'min_to_read' => 2,
   ]);
   $result = DB::table('posts')->latest()->first();
   print_r($result);

9.
```
use Illuminate\Support\Facades\DB;
$affectedRows = DB::table('posts')
->where('id', 2)
->update([
    'excerpt' => 'Laravel module 17 is really complex',
    'description' => 'Laravel module 17 is really complex and Laravel
     module 17 is really complex',
]);
print_r($affectedRows);
```

10.
```
use Illuminate\Support\Facades\DB;
class DemoController extends Controller {
        function DemoAction(){
        $id = 3;
        $results = DB::table('posts')
                ->where('id','=', $id)
                ->delete();
        return $results;
        }

}
```

11. In Laravel's query builder, the aggregate methods count(), sum(), avg(), max(), and min() are used to perform calculations on specific columns of a database table. These methods allow you to retrieve aggregated data based on various criteria. Here's an example of each method:
    - Count(): `$count = DB::table('users')->count();`
    - Sum(): `$totalPrice = DB::table('products')->sum('price');`
    - Avg(): `$averageRating = DB::table('products')->avg('rating');`
    - Max(): `$maxPrice = DB::table('products')->max('price');`
    - Min(): `$minAge = DB::table('users')->min('age');`

12. The whereNot methods used to negate a given group of query constraints. This method is useful when you want to exclude certain records based on specific criteria.
    Example:

```
Function DemoAction(){
$results = DB::table(products)
            ->where('price', '=',  '3000')
            ->whereNot('title', 'LIKE'  '%Ca%')
            ->get();
        return $result;
}
```

13. In Laravel's query builder, the exists() and doesntExist() methods are used to check the existence of records in a database table. They are particularly useful when you want to determine whether any records match certain conditions. The exists() method is used to check if any records exist in the table that match the specified conditions. It returns a boolean value, true if at least one record is found, and false otherwise. Here's example:
    - Exists():

```
if (DB::table('users')->where('status', 'active')->exists()) {
   // At least one active user exists
} else {
   // No active users found
}
```

    - doesntExist():

```
if (DB::table('users')->where('status', 'active')->doesntExist()) {
   // No active users found
} else {
   // At least one active user exists
}
```

14.
```
$posts = DB::table('posts')
        ->whereBetween('min_to_read', [1, 5])
        ->get();
print_r($posts);
```

15.
```
$results = DB::table('posts')
            ->where('id', '=', 3)
            ->increment('min_to_read', 1);

echo "Number of affected rows: " . $ results;
```