**Оглавление**

[Аудит безопасности 2](#_Toc110282043)

[1. Уязвимости XSS 2](#_Toc110282044)

[2. SQL Injection 3](#_Toc110282045)

[Запросы + скрины исполнения с phpMyAdmin 4](#_Toc110282046)

[1. Запрос 1 4](#_Toc110282047)

[2. Запрос 2 5](#_Toc110282048)

[3. Запрос 3 7](#_Toc110282049)

[4. Запрос 4 8](#_Toc110282050)

[5. Запрос 5 9](#_Toc110282051)

[6. Запрос 6 10](#_Toc110282052)

[7. Дополнительный запрос 1 11](#_Toc110282053)

[8. Дополнительный запрос 2 11](#_Toc110282054)

# Аудит безопасности

## Уязвимости XSS

Слабые места были обнаружены в местах получения GET-параметров при выводе детальной страницы категории, страницы товара.

Для устранения была создана функция-обработчик intHandling(), которая приводит полученное значение к типу *int*, тем самым убирает все возможные специальные символы и непредвиденные конструкции возвращая число (листинги 1-2).

Листинг 1 — Обработка числовых параметров (int).

1. function intHandling**(**$value**)** **{**
2. **return** intval**(**$value**);**
3. **}**

Пример использования в коде:

Листинг 2 — Получение данных о категории по параметру GET[‘id’] (productListController.php).

1. **if(**$\_SERVER**[**'REQUEST\_METHOD'**]** **==** 'GET' **&&** isset**(**$\_GET**[**'cat\_id'**]))** **{**
2. $catID **=** intHandling**(**$\_GET**[**'cat\_id'**]);**
3. $currentPage **=** **!**empty**(**$\_GET**[**'page'**])** **?** intHandling**(**$\_GET**[**'page'**])** **:** 1**;**
4. $category **=** selectOne**(**'category'**,** **[**'category\_id' **=>** $catID**]);**
5. **if(**empty**(**$category**))** **{**
6. redirectNotFoundPage**();**
7. **}**
8. $catID **=** $category**[**'category\_id'**];**
9. $catName **=** $category**[**'name'**];**
10. $catDescription **=** $category**[**'description'**];**
11. $products **=** getAllProductsWithCatID**(**$catID**,** $currentPage**);**
12. **if(**empty**(**$products**))** **{**
13. redirectNotFoundPage**();**
14. **}**
15. $countRows **=** countFoundRowsOnLastQuery**();**
16. **}**

Так же слабые места были обнаружены в обработчике формы обратной связи. Для устранения слабых мест создана функция stringHandling(), которая обрабатывает строковые параметры убирая теги HTML, убирает лишние пробелы, и преобразует специальные символы в HTML-сущности (листинги 3-4).

Листинг 3 — Обработка строковых параметров.

1. function stringHandling**(**$value**)** **{**
2. $value **=** trim**(**$value**);**
3. $value **=** strip\_tags**(**$value**);**
4. **return** htmlspecialchars**(**$value**);**
5. **}**

Пример использования в коде:

Листинг 4 — Обработка данных полученных с формы методом POST (feedbackFormController.php)

1. **if(**$\_SERVER**[**'REQUEST\_METHOD'**]** **=** 'POST' **&&** isset**(**$\_POST**[**'btn-submit'**]))** **{**
2. $username **=** stringHandling**(**$\_POST**[**'username'**]);**
3. $email **=** stringHandling**(**$\_POST**[**'email'**]);**
4. $birthday **=** stringHandling**(**$\_POST**[**'birthday'**]);**
5. $gender **=** intHandling**(**$\_POST**[**'gender'**]);**
6. $title **=** stringHandling**(**$\_POST**[**'title'**]);**
7. $description **=** stringHandling**(**$\_POST**[**'description'**]);**
8. $params **=** **[**'username' **=>** $username**,**
9. 'email' **=>** $email**,**
10. 'birthday' **=>** $birthday**,**
11. 'gender' **=>** $gender**,**
12. 'title' **=>** $title**,**
13. 'description' **=>** $description**];**
14. **if(**checkFormParameters**(**$username**,** $email**,** $birthday**,** $title**,** $description**))** **{**
15. insertInQuestions**(**$params**);**
16. **}**
17. setcookie**(**'username'**,** $params**[**'username'**],** **time()** **+** 3600**);**
18. setcookie**(**'email'**,** $params**[**'email'**],** **time()** **+** 3600**);**
19. setcookie**(**'birthday'**,** $params**[**'birthday'**],** **time()** **+** 3600**);**
20. setcookie**(**'gender'**,** $params**[**'gender'**],** **time()** **+** 3600**);**
21. **}**

## SQL Injection

Для предотвращения sql-инъекций применяются все те же функции stringHandler(), intHandler() описанные выше.

Используется подготовленный запрос на вставку в таблицу с обращениями (с формы обратной связи).

Листинг 5 — Вставка в таблицу questions при помощи подготовленного запроса.

1. function insertInQuestions**(**$params**)** **{**
2. global $pdo**;**
3. $sql **=** "INSERT INTO questions(name, email, birthday, gender, title, description)
4. VALUE(:username, :email, :birthday, :gender, :title, :description)"**;**
5. $query **=** $pdo**->**prepare**(**$sql**);**
6. $query**->**execute**(**$params**);**
7. **return** $pdo**->**lastInsertId**();**
8. **}**

# Запросы + скрины исполнения с phpMyAdmin

Здесь отдельно вынесены запросы с файла ‘app/database/requests.php’. Все данные (id, string) передаются в функции уже в обработанном виде, алгоритмы преобразования описаны выше.

## Запрос 1

Посылает запрос к БД на получение списка категорий и кол-ва товаров в ней.

Опционально: при передаче параметра (id товара) получает категории только конкретного товара.

Листинг 1 — Получение списка категорий и кол-ва товаров в ней.

1. /\*\*
2. \* Посылает запрос к БД на получение списка категорий и кол-ва товаров в ней.
3. \* Опционально: при передаче параметра (id товара) получает категории только конкретного товара
4. \* @param $productID int
5. \* @return array|false
6. \*/
7. function selectCategoriesAndCountProducts**(**$productID **=** null**)** **{**
8. global $pdo**;**
9. $searchWithProductID **=** ""**;**
10. **if(**$productID**)** **{**
11. $searchWithProductID **=** " AND pc.product\_id = $productID"**;**
12. **}**
13. $sql **=** "SELECT c.category\_id as cat\_id,
14. c.name as cat\_name,
15. COUNT(pc.product\_id) as count\_product
16. FROM product\_categories pc
17. JOIN category c ON pc.category\_id = c.category\_id
18. JOIN product p ON pc.product\_id = p.product\_id
19. WHERE p.is\_active = true
20. $searchWithProductID
21. GROUP BY c.category\_id, c.name
22. ORDER BY count\_product DESC"**;**
23. $query **=** $pdo**->**prepare**(**$sql**);**
24. $query**->**execute**();**
25. **return** $query**->**fetchAll**();**
26. **}**

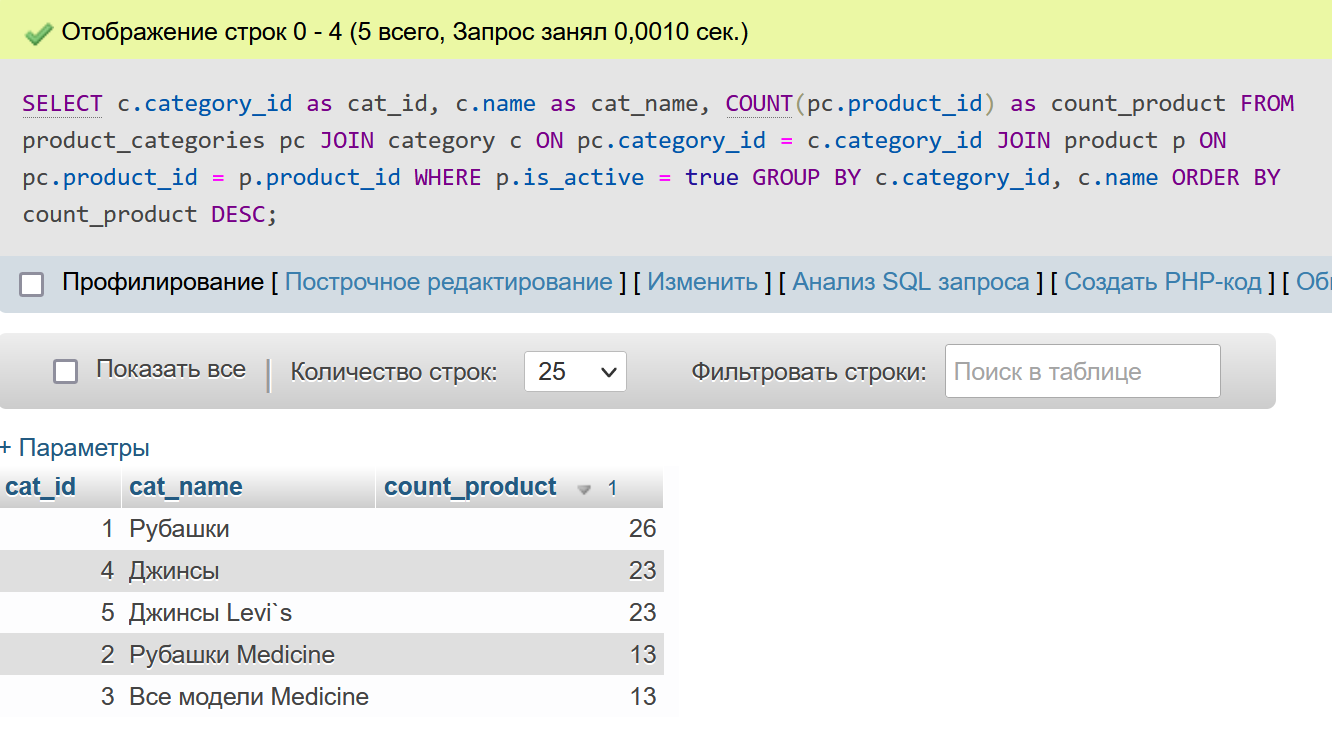


Рисунок 1 – Результат запроса.

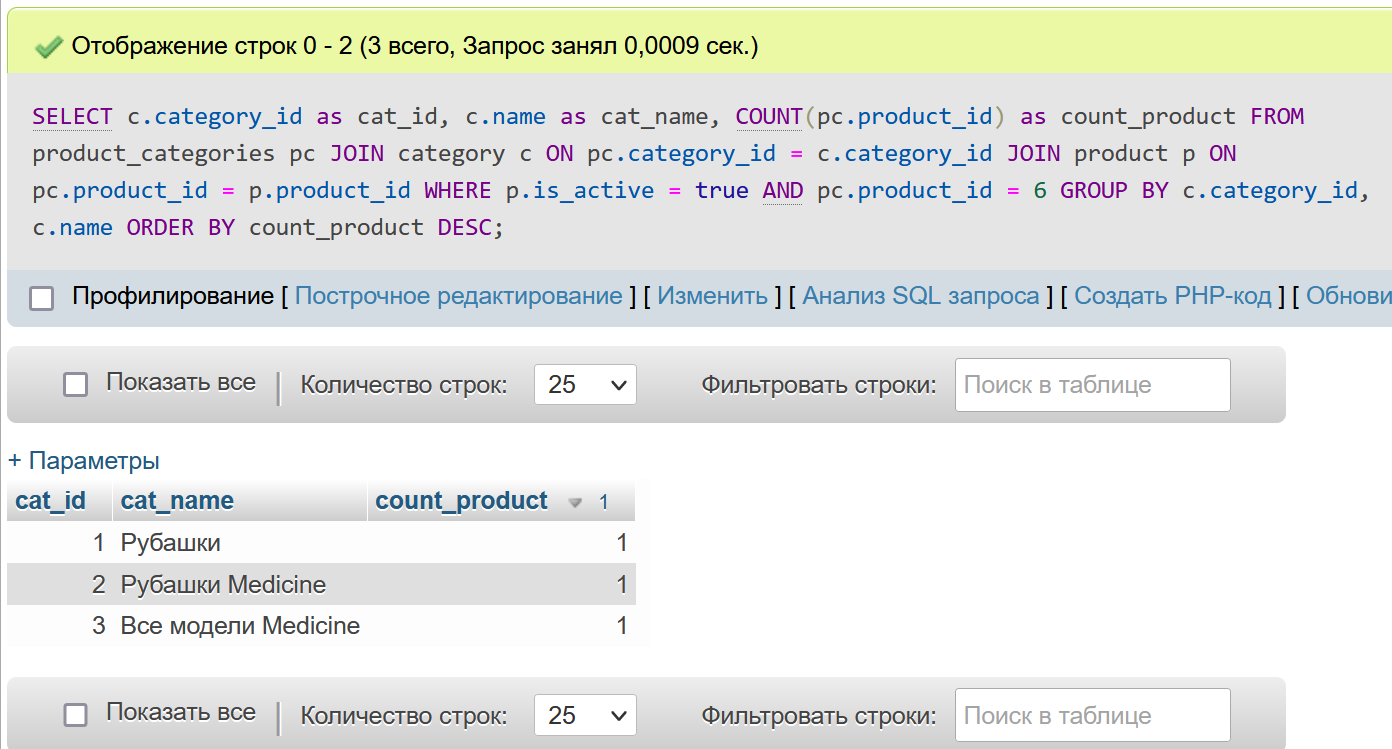


Рисунок 2 – Результат запроса с получением категорий конкретного товара.

## Запрос 2

Посылает запрос на получение одной записи с БД. В коде применяется для получения данных о категории из таблицы category (в дальнейшем – можно получить запись из любой таблицы и необходимыми параметрами или без них).

Опционально: можно передать параметры.

Листинг 2 — Получает одну запись из БД.

1. /\*\*
2. \* Посылает запрос на получение одной записи с БД
3. \* Опционально: можно передать параметры
4. \* @param $table string
5. \* @param $params array-key
6. \* @return array|false
7. \*/
8. function selectOne**(**$table**,** $params **=** **[])** **{**
9. global $pdo**;**
10. $sql **=** "SELECT \* FROM $table"**;**
11. // Обработка параметров **(**при наличии**)**
12. $sql **=** prepareParamsToString**(**$sql**,** $params**);**
13. $sql **=** $sql **.** " LIMIT 1"**;**
14. $query **=** $pdo**->**prepare**(**$sql**);**
15. $query**->**execute**();**
16. **return** $query**->**fetch**();**
17. **}**
18. /\*\*
19. \* При наличии параметров для select составляет строку запроса с переданными параметрами
20. \* @param $sql string
21. \* @param $params array-key
22. \* @return string
23. \*/
24. function prepareParamsToString**(**$sql**,** $params**)** **{**
25. **if(!**empty**(**$params**))** **{**
26. $i **=** 0**;**
27. **foreach** **(**$params as $key **=>** $value**)** **{**
28. **if(!**is\_numeric**(**$value**))** **{**
29. $value **=** "'" **.** $value **.** "'"**;**
30. **}**
31. **if(**$i **===** 0**)** **{**
32. $sql **=** $sql **.** " WHERE $key = $value"**;**
33. **}**
34. **else** **{**
35. $sql **=** $sql **.** " AND $key = $value"**;**
36. **}**
37. $i**++;**
38. **}**
39. **}**
40. **return** $sql**;**
41. **}**

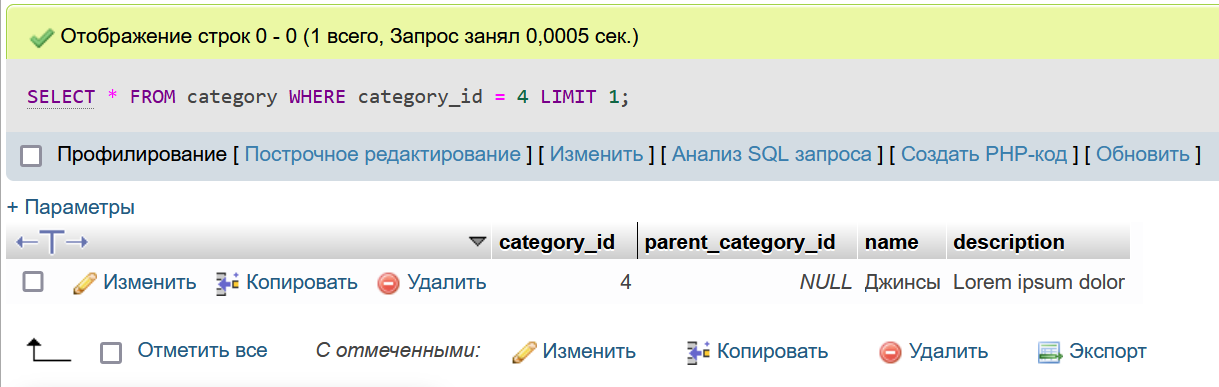


Рисунок 3 – Результат запроса.

## Запрос 3

Посылает запрос к БД на получение списка товаров из выбранной категории. Директива SQL\_CALC\_FOUND\_ROWS используется для подсчёта кол-ва товара для дальнейшего их разделения на страницы. Запрос FOUND\_ROWS() нужен для получения кол-ва строк полученных предыдущим запросом с директивой SQL\_CALC\_FOUND\_ROWS.

Листинг 3 — Запрос на получение списка товаров из выбранной категории.

1. /\*\*
2. \* Посылает запрос к БД на получение списка товаров из выбранной категории
3. \* @param $catID int
4. \* @param $page int
5. \* @param $countPerPage int
6. \* @return array|false
7. \*/
8. function selectAllProductsByCatID**(**$catID**,** $page**,** $countPerPage**)** **{**
9. global $pdo**;**
10. $sql **=** "SELECT SQL\_CALC\_FOUND\_ROWS
11. pc.product\_id as p\_id,
12. p.name as p\_name,
13. i.path,
14. i.alt,
15. c.name as cat\_name
16. FROM product\_categories pc
17. JOIN product p ON pc.product\_id = p.product\_id
18. JOIN category c ON p.main\_category\_id = c.category\_id
19. JOIN image i ON p.main\_image\_id = i.image\_id
20. WHERE pc.category\_id = $catID
21. AND p.is\_active = true
22. LIMIT $page, $countPerPage"**;**
23. $query **=** $pdo**->**prepare**(**$sql**);**
24. $query**->**execute**();**
25. **return** $query**->**fetchAll**();**
26. **}**
27. /\*\*
28. \* Возвращает кол-во строк, полученных предыдущим запросом
29. \* @return array-key
30. \*/
31. function countFoundRowsOnLastQuery**()** **{**
32. global $pdo**;**
33. $sql **=** "SELECT FOUND\_ROWS() as found\_rows"**;**
34. $query **=** $pdo**->**prepare**(**$sql**);**
35. $query**->**execute**();**
36. **return** $query**->**fetch**();**
37. **}**

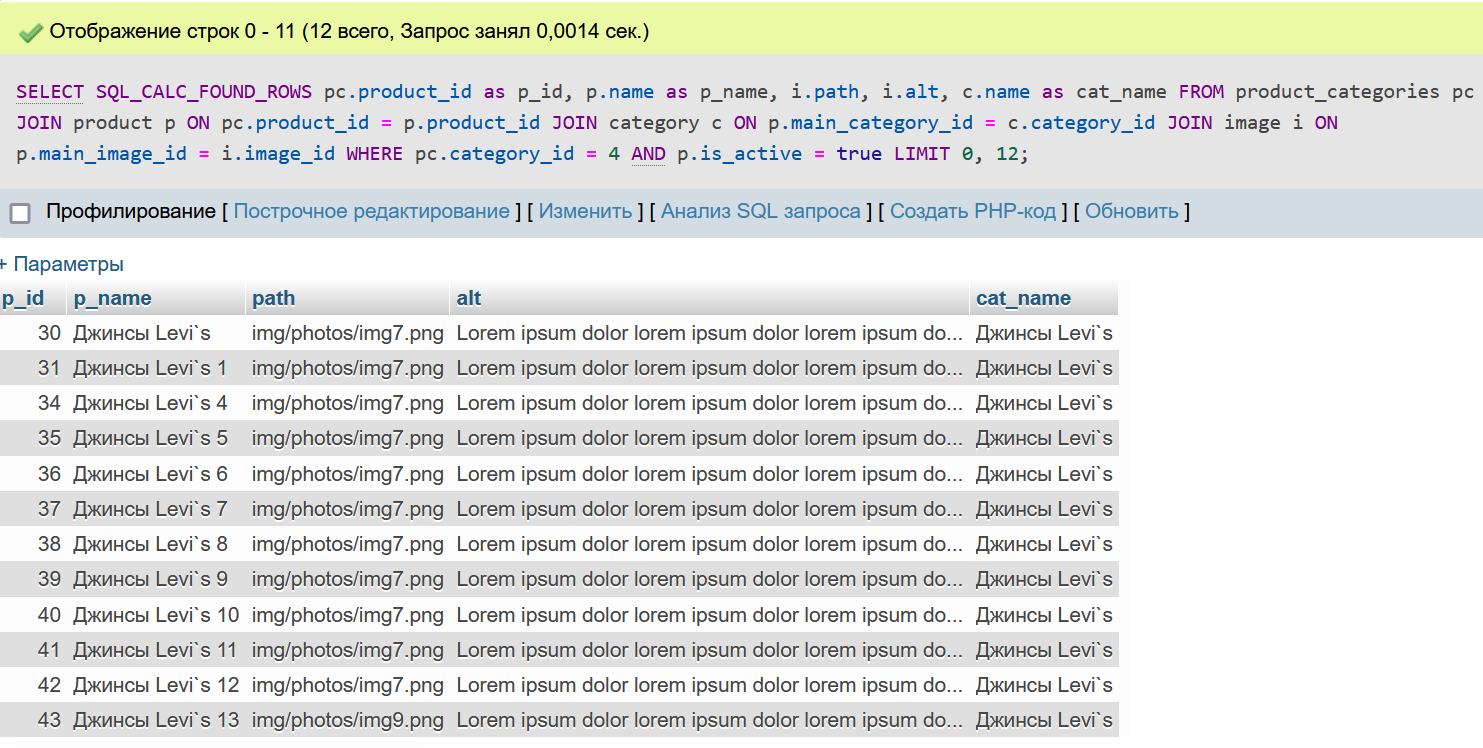


Рисунок 4 – Результат запроса.

## Запрос 4

Посылает запрос к БД на получение массива картинок товара по его id.

Листинг 4 — Получение массива картинок товара по id.

1. /\*\*
2. \* Посылает запрос к БД на получение массива картинок товара по его id
3. \* @param $productID int
4. \* @return array|false
5. \*/
6. function selectImagesOfProductByID**(**$productID**)** **{**
7. global $pdo**;**
8. $sql **=** "SELECT i.path,
9. i.alt
10. FROM images
11. JOIN product p ON images.product\_id = p.product\_id
12. JOIN image i ON images.image\_id = i.image\_id
13. WHERE images.product\_id = $productID
14. AND p.is\_active = true"**;**
15. $query **=** $pdo**->**prepare**(**$sql**);**
16. $query**->**execute**();**
17. **return** $query**->**fetchAll**();**
18. **}**

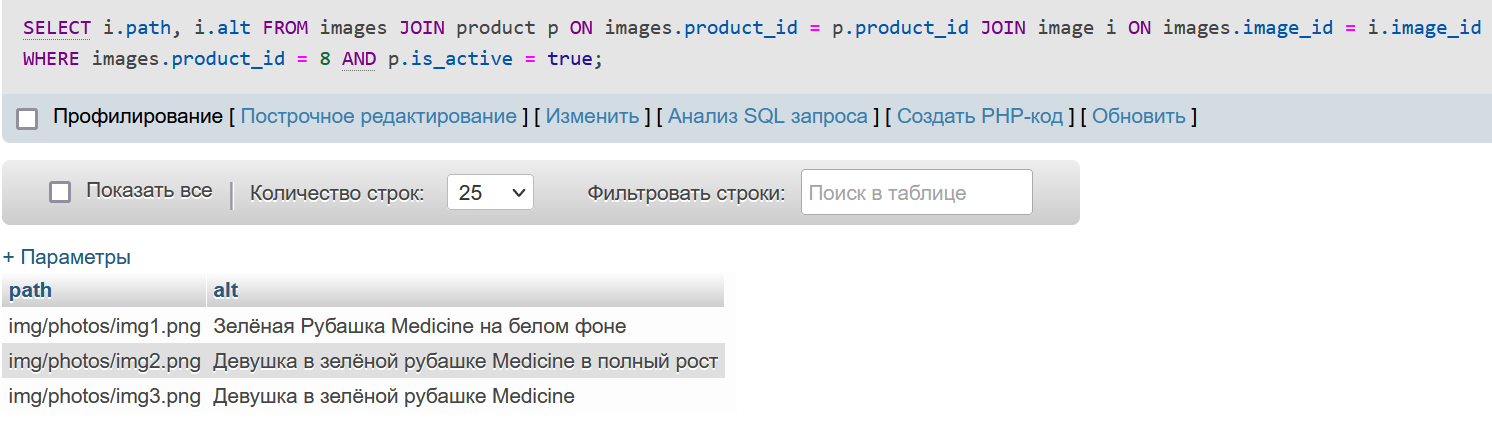


Рисунок 5 – Результат запроса.

## Запрос 5

Посылает запрос на получение данных о товаре.

Листинг 5 — Получение данных о товаре + путь до основной картинки и alt.

1. /\*\*
2. \* Посылает запрос на получение основной иформации о товаре
3. \* @param $productID int
4. \* @return array-key
5. \*/
6. function selectProductWithMainImageByID**(**$productID**)** **{**
7. global $pdo**;**
8. $sql **=** "SELECT p.product\_id as p\_id,
9. p.name,
10. i.path as main\_image\_path,
11. i.alt as main\_image\_alt,
12. p.description,
13. p.price,
14. p.price\_with\_sale,
15. p.price\_with\_promo,
16. p.is\_active
17. FROM product p
18. JOIN image i ON p.main\_image\_id = i.image\_id
19. WHERE p.product\_id=$productID
20. LIMIT 1"**;**
21. $query **=** $pdo**->**prepare**(**$sql**);**
22. $query**->**execute**();**
23. **return** $query**->**fetch**();**
24. **}**

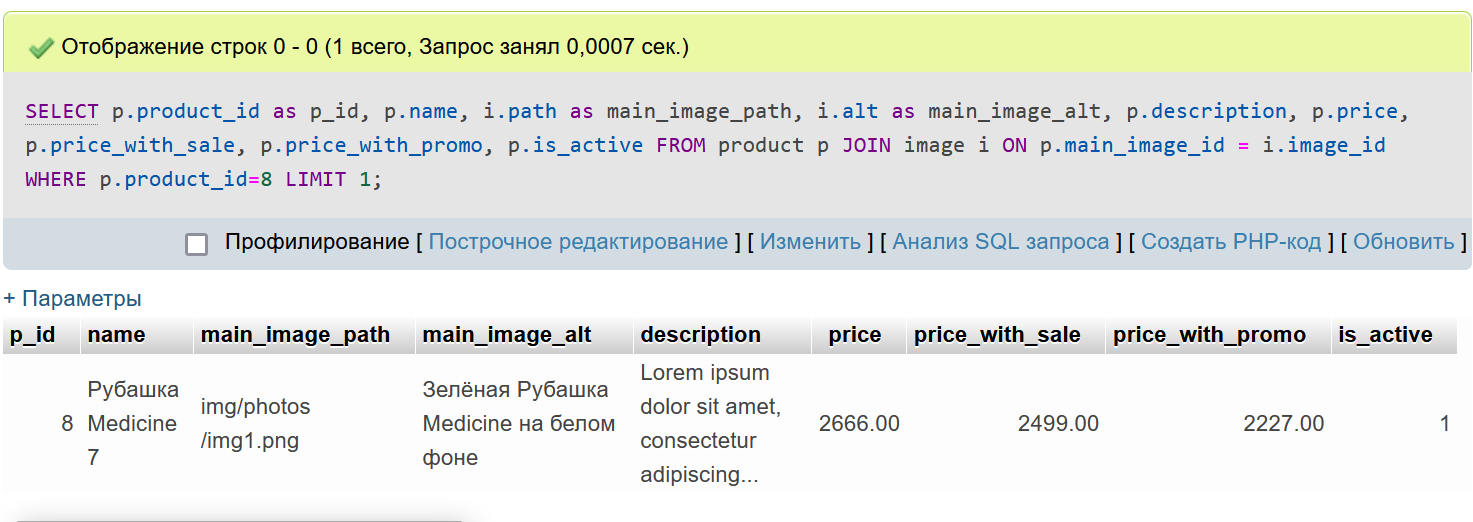


Рисунок 6 – Результат запроса.

## Запрос 6

Добавление новой записи в таблицу ‘questions’ (с формы обратной связи).

Листинг 6 — Вставка записи в таблицу с формы обратной связи.

1. /\*\*
2. \* Добавление новой записи в таблицу `questions`
3. \* @param $params array
4. \* @return false|string
5. \*/
6. function insertInQuestions**(**$params**)** **{**
7. global $pdo**;**
8. $sql **=** "INSERT INTO questions(name, email, birthday, gender, title, description)
9. VALUE(:username, :email, :birthday, :gender, :title, :description)"**;**
10. $query **=** $pdo**->**prepare**(**$sql**);**
11. $query**->**execute**(**$params**);**
12. **return** $pdo**->**lastInsertId**();**
13. **}**

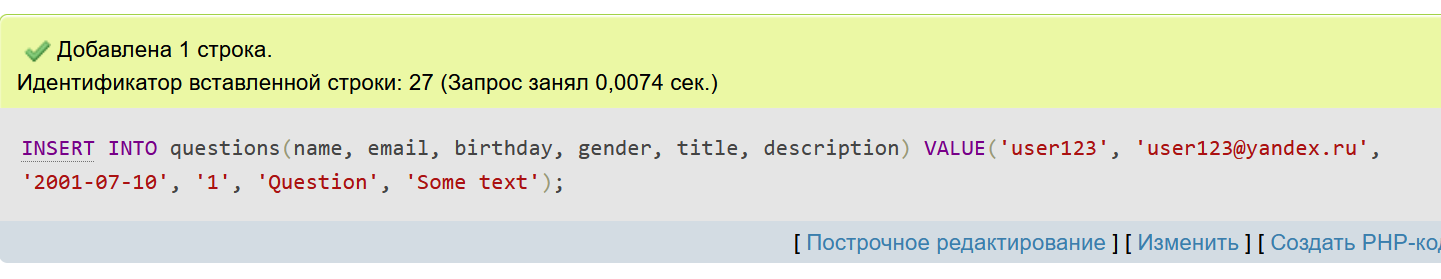


Рисунок 7 – Результат запроса.

## Дополнительный запрос 1

Составить sql запрос, который выведет все категории, даже с количеством товаров 0, но с указанием количества товара для каждой категории, как и в оригинальном задании.

Листинг 7 — Вывод всех категорий даже с количеством товаров 0.

1. SELECT c.name,
2. COUNT(pc.product\_id) as count
3. FROM category c
4. LEFT JOIN product\_categories pc ON c.category\_id = pc.category\_id
5. GROUP BY c.name;

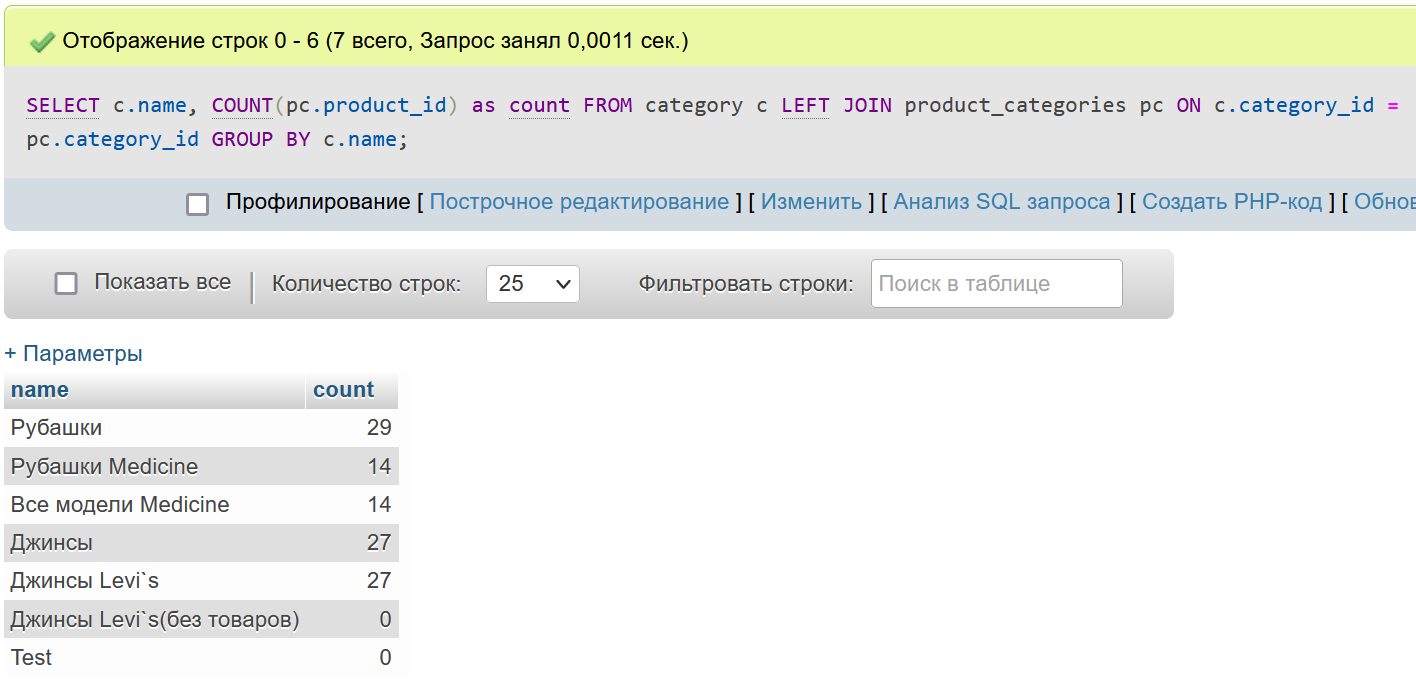


Рисунок 8 – Результат запроса.

## Дополнительный запрос 2

Составить sql запрос, который выведет все категории товаров с указанием количество товаров, имеющих количество товаров больше либо равно 2.

Листинг 8 — Вывод всех категорий имеющих количество товаров больше либо равно 2.

1. SELECT c.name,
2. COUNT(pc.product\_id) as count\_product
3. FROM category c
4. JOIN product\_categories pc ON c.category\_id = pc.category\_id
5. JOIN product p ON pc.product\_id = p.product\_id
6. WHERE p.is\_active = true
7. GROUP BY c.category\_id
8. HAVING count\_product >= 2;

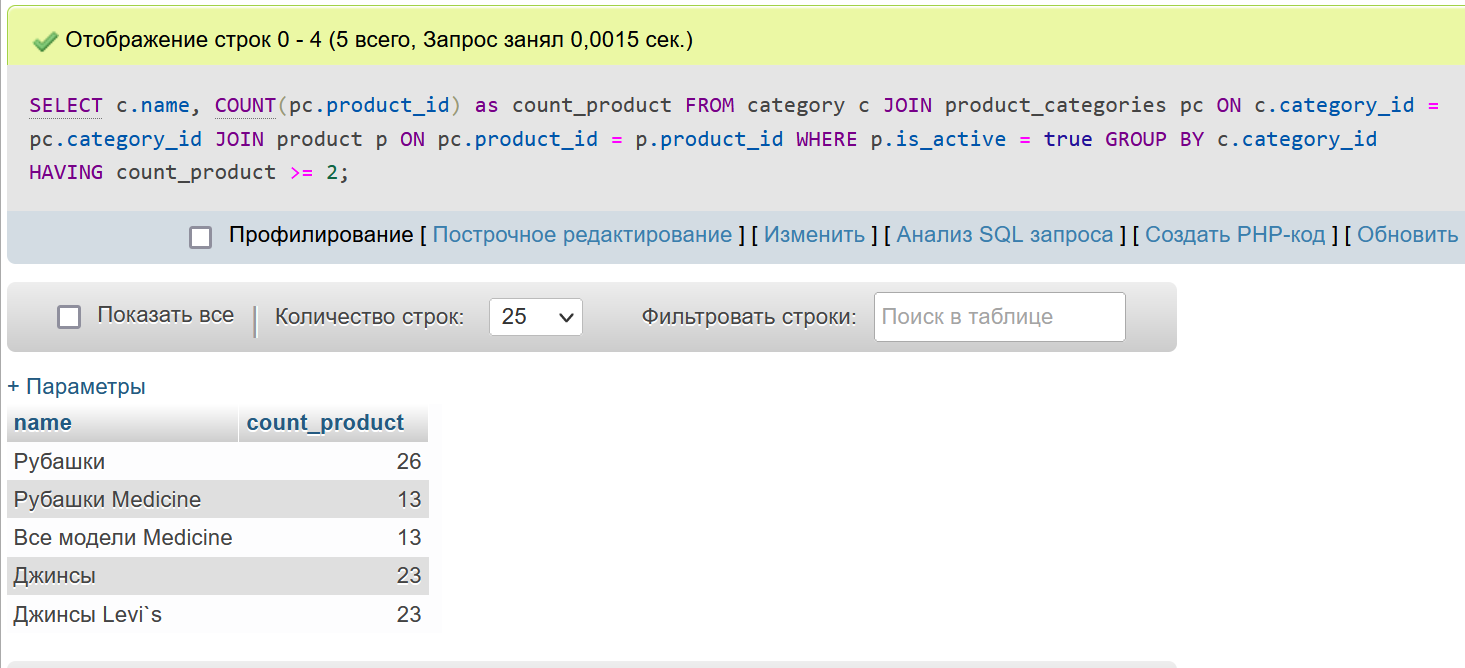


Рисунок 9 – Результат запроса.