**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**імені ТАРАСА ШЕВЧЕНКА**

****

**ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**

**Кафедра прикладних інформаційних систем**

**Звіт до практичної роботи №4**

**з курсу**

**«Бази Даних»**

*студента 2 курсу*

*групи ПП-22*

*спеціальності 122 «Комп'ютерні науки»*

*ОП «Прикладне програмування»*

Шевлюк Вікторії Віталіївни

*Викладач:*

асистент

Криволапов Я. В.

**Київ – 2022**

**Тема:** Захист інтерфейсу користувача від SQL-ін’єкцій. Виконання складних запитів та процедур.

**Мета:** Набуття навичок захисту інтерфейсу користувача від SQL-ін’єкцій.

**Завдання:**

За допомогою PHP-скриптів реалізувати виконання запитів 1.4, 1.6, 2.1, 2.2. Запити мають бути захищеними від SQL-ін’єкцій, а результат виконання виводитись у вигляді таблиці.

**1.4** Інформацію про реалізовану за певний період часу та не сплачену продукцію кондитерської фабрики. Дата впровадження (початковий та кінцевий термін) повинна задаватись під час виконання запиту у вигляді параметра, передбачити також можливість отримання інформації для всього періоду часу. Динамічний набір записів повинен мати наступні поля: Повна назва продукції, Кількість продукції, Дата реалізації, Дата сплати, Прогноз ціни, Реалізація.

**1.6** Визначити дані про реалізацію кондитерських виробів за останні дні, за основу взяти кінцеву дату реалізації. Кількість останніх днів має вводитись у вигляді параметра. Динамічний набір записів складається з таких полів: Технолог цеху, Повна назва продукції, Кількість продукції, Дата реалізації, Дата сплати.

**2.1** Для кондитерської продукції загальну кількість виробництва та загальну реалізацію (назва повинно задаватися під час виконання запиту у вигляді параметра, передбачити можливість отримання інформації про всю продукцію одночасно);

**2.2** Для технологів загальну кількість виробництва кондитерської продукції та загальну реалізацію готової продукції за деякий місяць деякого року (значення параметрів для розрахункових полів Рік та Місяць, що будуються по полю Дата впровадження, повинні вводитись під час виконання запиту).

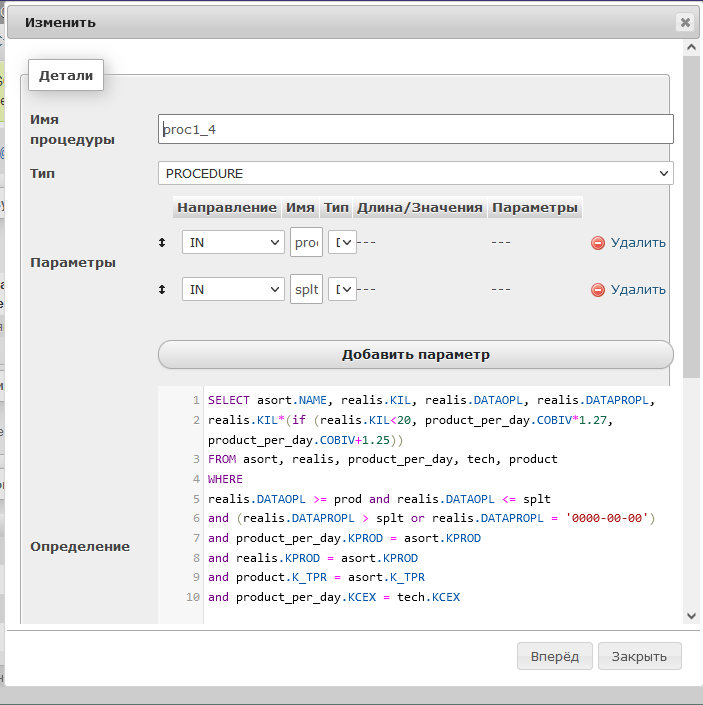
**Хід роботи:**

Для того, щоб захистити користувача від SQL-ін’єкцій я використала можливості такі СУБД як процедури.

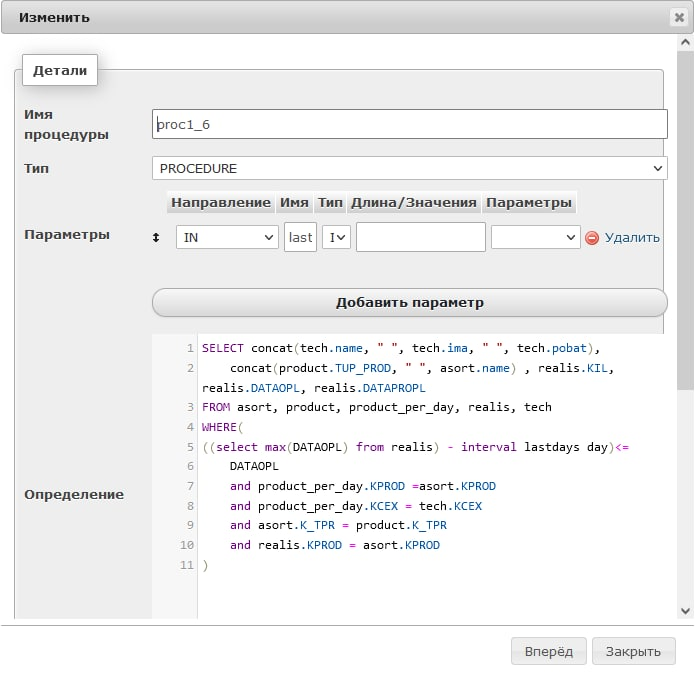
Роботу починаємо з того, що створюємо процедури у нашій базі даних.

Нижче наведені скріншоти моїх процедур:

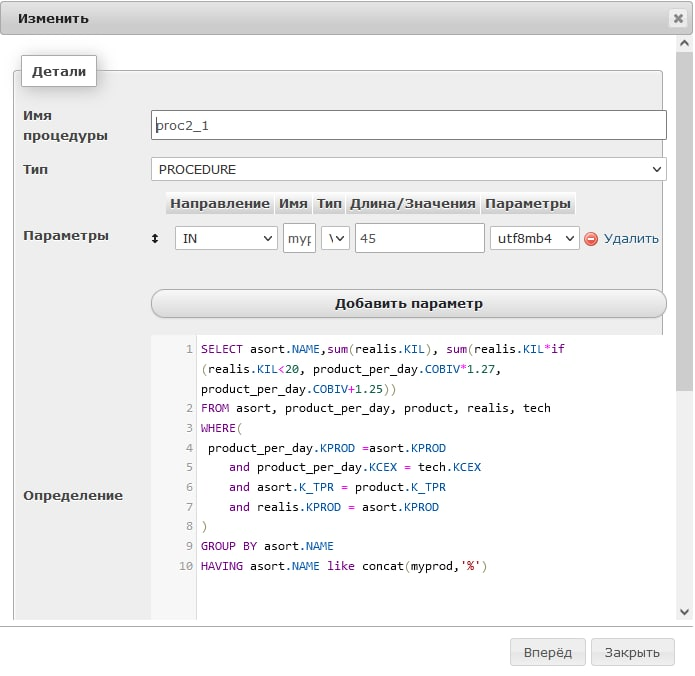
► **Процедура 1\_4**



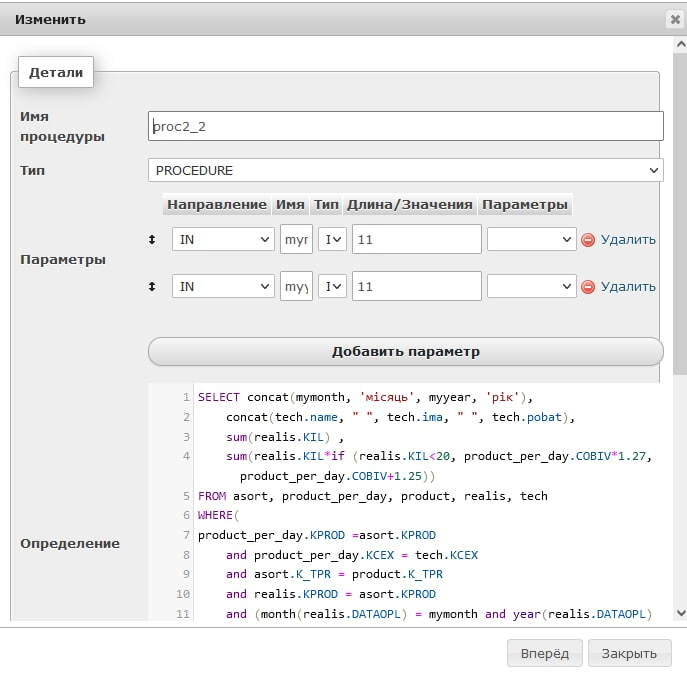
**► Процедура 1\_6**



**►Процедура 2\_1**



**► Процедура 2\_2**



Тепер напишемо скрипти, що будуть виконувати наші процедури через графічний інтерфейс. У кожному скрипті будуть створюватись поля для введення інформації від користувача та кнопки для підтвердження запиту, після натискання якої будуть виводитись відповідні таблиці.

Розглянемо код скрипту для першої процедури:

**<form method="get">**

**DATAOPL:<br>**

**<input type="text" name="rel"><br>**

**<br>**

**DATAPROPL:<br>**

**<input type="text" name="splt"><br>**

**<br>**

**<input type="submit" value="GO">**

**</form>**

**<br> <br>**

**<?php**

**mysqli\_report(MYSQLI\_REPORT\_ERROR | MYSQLI\_REPORT\_STRICT);**

**if(isset($\_GET['rel'], $\_GET['splt']))**

**{**

**include ("config.php");**

**$query = "call proc1\_4 ('$\_GET[rel]', '$\_GET[splt]')";**

**echo "<br><br>";**

**$ver=mysqli\_query($dbcon,$query);**

**if(!$ver){**

**echo "<P>Не вдалося виконати запит</P>";**

**exit(mysqli\_error($dbcon));**

**}**

**echo "<P><B> Запит 1.4 </B></P>";**

**echo "<table border=1>";**

**while(list($NAME, $TUP\_PROD, $DATAOPL, $DATAPROPL) = mysqli\_fetch\_row($ver))**

**{**

**echo "<tr>**

**<td> $NAME </td>**

**<td> $TUP\_PROD </td>**

**<td> $DATAOPL </td>**

**<td> $DATAPROPL </td>**

**</td>";**

**}**

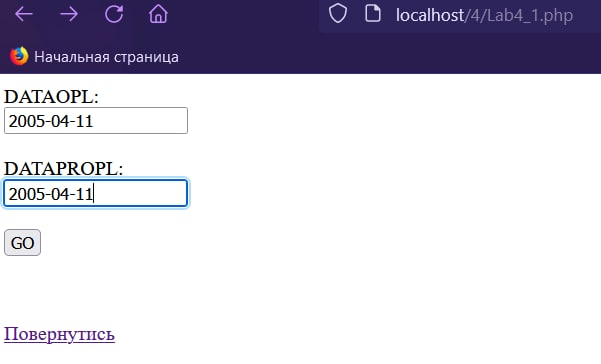
**echo "</table>";**

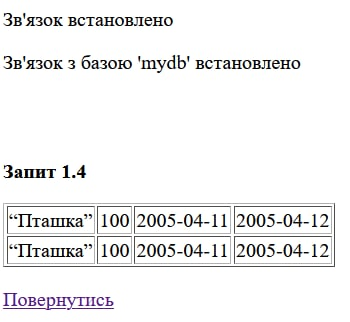
**echo "<P> </P>";**

**}**

**?>**

Результат виконання цього скрипту виглядає наступним чином:





Оскільки за варіантом дедлайн сплати має бути в день реалізації, в полях я вказую однакову дату.

Тепер переглянемо скрипт для другої процедури:

<form method="get">

Lastdays:<br>

<input type="text" name="lastdays"><br>

<br>

<input type="submit" value="GO">

</form>

<br> <br>

<?php

mysqli\_report(MYSQLI\_REPORT\_ERROR | MYSQLI\_REPORT\_STRICT);

if(isset($\_GET['lastdays']))

{

include ("config.php");

$query = "call proc1\_6 ('$\_GET[lastdays]')";

echo "<br><br>";

$ver=mysqli\_query($dbcon,$query);

if(!$ver){

echo "<P>Не вдалося виконати запит</P>";

exit(mysqli\_error($dbcon));

}

echo "<P><B> Запит 1.6 </B></P>";

echo "<table border=1>";

while(list($fullTechName, $fullProdName, $KIL, $DATAOPL, $DATAPROPL) = mysqli\_fetch\_row($ver))

{

echo "<tr>

<td> $fullTechName </td>

<td> $fullProdName </td>

<td> $KIL </td>

<td> $DATAOPL </td>

<td> $DATAPROPL </td>

</td>";

}

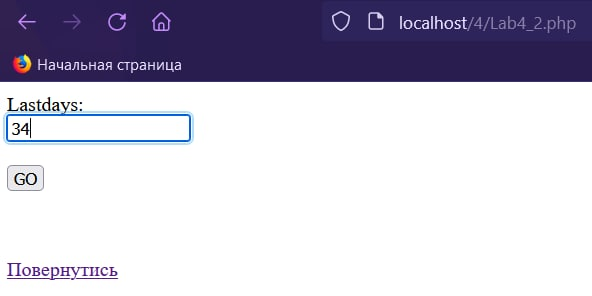
echo "</table>";

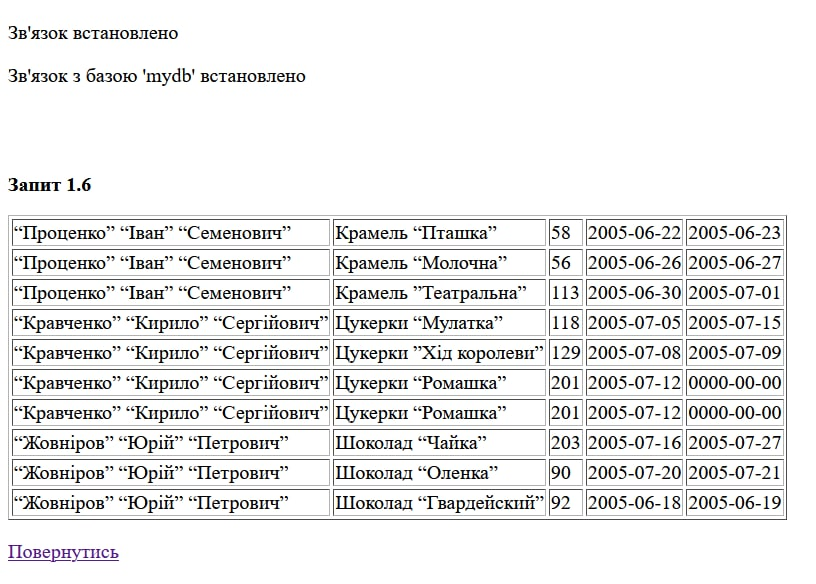
echo "<P> </P>";

}

?>

Результат роботи другої процедури:





Тепер до скрипту під третю процедуру:

<form method="get">

Production\_name:<br>

<input type="text" name="myprod"><br>

<br>

<input type="submit" value="GO">

</form>

<br> <br>

<?php

mysqli\_report(MYSQLI\_REPORT\_ERROR | MYSQLI\_REPORT\_STRICT);

if(isset($\_GET['myprod']))

{

include ("config.php");

$query = "call proc2\_1 ('$\_GET[myprod]')";

echo "<br><br>";

$ver=mysqli\_query($dbcon,$query);

if(!$ver){

echo "<P>Не вдалося виконати запит</P>";

exit(mysqli\_error($dbcon));

}

echo "<P><B> Запит 2.1 </B></P>";

echo "<table border=1>";

while(list($NAME, $fullAMOUNT, $fullREALISATION) = mysqli\_fetch\_row($ver))

{

echo "<tr>

<td> $NAME </td>

<td> $fullAMOUNT </td>

<td> $fullREALISATION </td>

</td>";

}

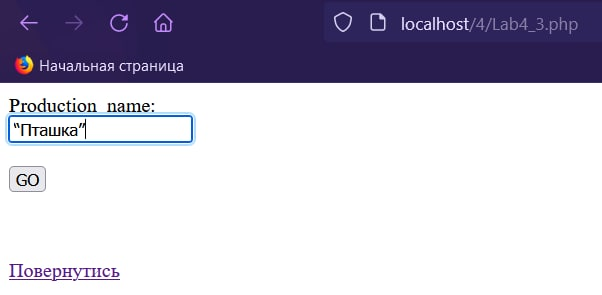
echo "</table>";

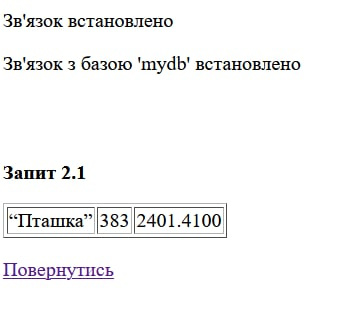
echo "<P> </P>";

}

?>

Результат роботи цього скрипту:





І останній скрипт для четвертої процедури:

<form method="get">

MONTHL:<br>

<input type="text" name="month"><br>

<br>

YEAR:<br>

<input type="text" name="year"><br>

<br>

<input type="submit" value="GO">

</form>

<br> <br>

<?php

mysqli\_report(MYSQLI\_REPORT\_ERROR | MYSQLI\_REPORT\_STRICT);

if(isset($\_GET['month'], $\_GET['year']))

{

include ("config.php");

$query = "call proc2\_2 ('$\_GET'month]', '$\_GET[year]')";

echo "<br><br>";

$ver=mysqli\_query($dbcon,$query);

if(!$ver){

echo "<P>Не вдалося виконати запит</P>";

exit(mysqli\_error($dbcon));

}

echo "<P><B> Запит 1.4 </B></P>";

echo "<table border=1>";

while(list($NAME, $sec, $third, $forth) = mysqli\_fetch\_row($ver))

{

echo "<tr>

<td> $NAME </td>

<td> $sec </td>

<td> $third </td>

<td> $forth </td>

</td>";

}

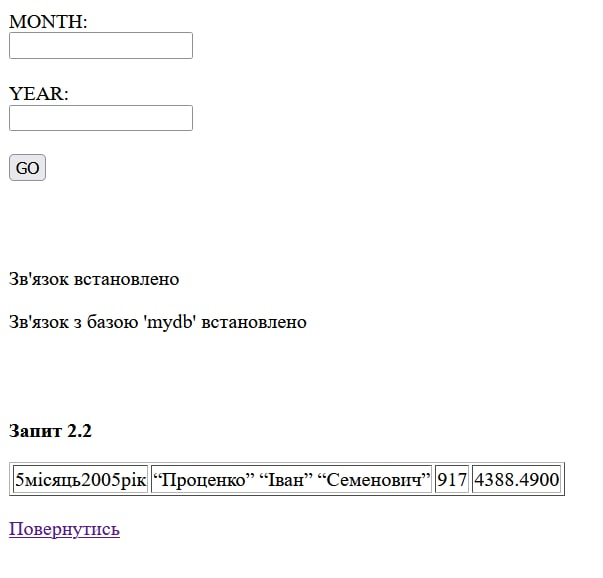
echo "</table>";

echo "<P> </P>";

}

?>

Результат його роботи:



**Висновок:** у ході даної лабораторної роботи я набула навичок захисту інтерфейсу користувача від SQL-ін’єкцій за допомогою створення процедур, а також покращила свої навичку у створюванні графічних інтерфейсів для користувачів за допомогою php-скриптів.

**Контрольні питання:**

1. ***Що таке SQL-ін’єкція?***

SQL ін'єкція — один з поширених способів злому сайтів та програм, що працюють з базами даних, заснований на впровадженні в запит довільного SQL-коду.

1. ***Які види та способи SQL-ін’єкцій ви знаєте?***

Використання UNION, Використання UNION + group\_concat(), Екранування хвоста запиту, розщеплення запиту.

1. ***Особливості методів POST та GET? Який з цих методів більш захищений від SQL-ін’єкцій?***

Метод POST використовується для запису даних із елементів графічного інтерфейсу.

Метод GET відображає записані дані з параметрами у пошуковому рядку, що саме по собі є не дуже безпечним.

1. ***Яке призначення функції mysqli\_escape\_string()?***

Ця функція допомагає уникати спеціальних символів у рядках.

1. ***Що таке регулярні вирази?***

Регулярні вирази це спеціальні правила що визначають символи, які можуть з’являтися у виразах.

1. ***Що таке інтерфейс користувача.***

Інтерфейс користувача́ — засіб зручної взаємодії користувача з інформаційною системою.

1. ***Основні елементи керування інтерфейсу користувача.***

Текстові поля, кнопки, чекбокси, радіо-кнопки, випадаючі менюшки

1. ***Значення тегу <form> …</form>.***

Тег для створення форми

1. ***Значення тегу <table> …</table>.***

Тег для створення таблиці.

1. ***Що таке повнотекстний пошук?***

Повнотекстовий пошук - це комплексний метод пошуку, який порівнює кожне слово запиту пошуку з кожним словом у документі чи базі даних.

1. ***Що таке релевантність?***

Релевантність — міра відповідності отримуваного результату бажаному.

1. ***Види повнотекстного пошуку.***

Існує три типи повнотекстового пошуку:

► Повнотекстовий пошук природною мовою.

► Логічний повнотекстовий пошук.

► Пошуки розширення запитів.

1. ***Значення оператора MATCH (…) AGAINST (…).***

Для повнотекстового пошуку в MySQL використовується конструкція MATCH (filelds) ... AGAINST (words). Вона може працювати в різних режимах, які досить сильно відрізняються між собою. Для всіх діє таке правило: дана конструкція повертає умовну релевантність, але спосіб обчислення якої може бути різним залежно від режиму.