



SISTEMAS DE INFORMAÇÃO E BASE DE DADOS

MEEC

Relatório do Projeto – Parte 3

Grupo N.º 2

Autores:

Leandro Almeida – n.º 84112
Sofia Estrela – n.º 84186
Vasco Candeias – n.º 84196

Turno:

Segunda-feira 15h30-17h

7 de Dezembro de 2018

1 Páginas Web

As páginas criadas, a que este relatório se refere, podem ser testadas em <http://web.tecnico.ulisboa.pt/ist425496/check.php>.

1.1 Clientes e animais

```
<html>
<body>
  <h1>Welcome to Veterinary Hospital</h1>
  <form action="animals.php" method="post">
    <h3>Input animal and client information</h3>
    <p>VAT client: <input type="text" name="VAT_client" required/></p>
    <p>Animal name: <input type="text" name="animal_name" required
      /></p>
    <p>Owner name: <input type="text" name="owner_name" required/></p>
    <p><input type="submit" value="Submit"/></p>
  </form>
</body>
</html>
```

Código 1: check.php

A página inicial corresponde ao ficheiro `check.php`. Esta página contém uma barra de procura onde se insere o VAT do cliente que leva o animal à consulta, o nome do animal, e o nome, ou parte, do dono do animal. Realça-se que todos estes parâmetros têm de ser preenchidos obrigatoriamente. O botão *Submit* reencaminha para a página dos animais, passando a informação do formulário para esta página pelo método POST. Na Figura 1 observa-se a página inicial.

Welcome to Veterinary Hospital

Input animal and client information

VAT client:

Animal name:

Owner name:

Figura 1: Página inicial: check.php

```
<html>
<body>
  <h3>Animals</h3>
  <?php
    $host = "db.tecnico.ulisboa.pt";
    $user = "istxxxxxx";
    $pass = "xxxxxxx";
    $dsn = "mysql:host=$host;dbname=$user";
    try
    {
      $connection = new PDO($dsn, $user, $pass);
```

```

}
catch(PDOException $exception)
{
    echo("<p>Error: ");
    echo($exception->getMessage());
    echo("</p>");
    exit();
}

$VAT_client = $_REQUEST['VAT_client'];
$name = $_REQUEST['animal_name'];
$owner_name = $_REQUEST['owner_name'];

$sqls = $connection->prepare("SELECT VAT FROM client WHERE VAT=
:VAT_client");
if ($sqls == 0) {
    $info = $sqls->errorInfo();
    echo("<p>Error: {$info[2]}</p>");
    exit();
}

$sqls->execute([':VAT_client'=> $VAT_client]);

$rows = $sqls->rowCount();
if ($rows == 0)
{
    echo("<p>There is no client with this VAT.</p>");
}
else
{
    echo("<p>You chose VAT Client: $VAT_client, Animal Name: $name,
    Owner Name: $owner_name </p>");
    $sqlw = $connection->prepare("SELECT distinct animal.name as
    an_name,
    animal.VAT, person.name as name_owner, species_name, colour,
    gender, birth_year, age
    FROM animal inner join consult on animal.name=consult.name
    and animal.VAT=consult.VAT_owner inner join person using(VAT)
    WHERE VAT_client = :VAT_client and animal.name = :animal_name
    and person.name like CONCAT('%',:owner_name,'%')");
    if ($sqlw == FALSE) {
        $info = $sqlw->errorInfo();
        echo("<p>Error: {$info[2]}</p>");
        exit();
    }

    $test = $sqlw->execute([':animal_name' => $name,
                           ':owner_name' => $owner_name,
                           ':VAT_client' => $VAT_client]);
    if ($test == FALSE) {
        $info = $sqlw->errorInfo();
        echo("<p>Error: {$info[2]}</p>");
        exit();
    }

    $result_w = $sqlw->fetchAll();

```

```

$nrows_an = $sqlw->rowCount();
if ($nrows_an == 0)
{
    ?>
    <p>The intersection of the 3 parameters is empty!</p>
    <form action='insertanimal.php' method='post'>
        <fieldset style='max-width:400px'><legend> Input the Animal
            information </legend>
            <input type='hidden' name='name' value='<?=$name?>' />
            <input type='hidden' name='VAT_client'
                value='<?=$VAT_client?>' />
            <p>VAT owner = <?=$VAT_client?></p>
            <p>Colour: <input type='text' name='colour' required/></p>
            <p>Gender: <input type='text' name='gender' required/></p>
            <p>Birthday: <input type='date' name='birth_year'
                max='<?=$date('Y-m-d')?>' required/></p>
            <p>Species name: <select name="species_name">
                <option value="---">---</option>
                <?php
                    $sql_aux = "SELECT distinct species_name FROM animal
                        order by species_name";
                    $result_aux = $connection->query($sql_aux);
                    if ($result_aux == FALSE)
                    {
                        $info = $connection->errorInfo();
                        echo("<p>Error: {$info[2]}</p>");
                        exit();
                    }

                    foreach($result_aux as $row)
                    {
                        $species_name=$row['species_name'];
                        echo("<option value=
                            \"\$species_name\">\$species_name</option>");
                    }

                ?>
            </select> </p>
            <p><input type='submit' value='Submit'></p>
        </fieldset>
    </form>

    <?php

    $sqlx = $connection->prepare("SELECT distinct animal.name as
        an_name,
        animal.VAT, person.name as name_owner, species_name,
        colour, gender, birth_year, age
    FROM animal inner join person using(VAT)
    WHERE animal.name = :animal_name and person.name like
        CONCAT('%',:owner_name,'%')
    UNION
    SELECT distinct animal.name as an_name, animal.VAT,
        person.name as name_owner, species_name, colour, gender,
        birth_year, age

```

```

FROM animal
inner join consult on animal.name = consult.name and
    animal.VAT = consult.VAT_owner
inner join person on animal.VAT = person.VAT WHERE
    VAT_client = :VAT_client");
if ($sqlx == FALSE) {
    $info = $sqlx->errorInfo();
    echo("<p>Error: {$info[2]}</p>");
    exit();
}

$sqlx->execute([':animal_name' => $name,
    ':owner_name' => $owner_name,
    ':VAT_client' => $VAT_client ]);

$result_w=$sqlx->fetchAll();
echo("<p>Here are some alternatives where the animal and
    owner match the input or animals this client took to
    consults.</p>");
}
else
{
    echo("<p>Results of the intersection of the 3
        parameters:</p>");
}

echo("<table border=\"1\" cellpadding=\"4\">");
echo("<tr><td>Owner name</td><td>VAT Owner</td><td>Animal
    name</td><td>Species</td><td>Colour</td><td>Gender
    </td><td>Birthday</td><td>Age</td></tr>");
foreach($result_w as $row)
{
    echo("<tr><td>");
    echo($row['name_owner']);
    echo("</td><td>");
    echo($row['VAT']);
    echo("</td><td>");
    echo($row['an_name']);
    echo("</td><td>");
    echo($row['species_name']);
    echo("</td><td>");
    echo($row['colour']);
    echo("</td><td>");
    echo($row['gender']);
    echo("</td><td>");
    echo($row['birth_year']);
    echo("</td><td>");
    echo($row['age']);
    echo("</td></td>");
    echo("<td><a href=\"consults.php?name=\"");
    echo($row['an_name']);
    echo("&VAT_owner=");
    echo($row['VAT']);
    echo("&VAT_client=");
    echo($VAT_client);
    echo("</a></td></tr>\n");
}

```

```

        echo("</table>");
    }
    $connection = null;
?>
<form action='check.php' method='post'>
    <h3>Go back to homepage</h3>
    <p><input type='submit' value='Homepage' /></p>
</form>
</body>
</html>

```

Código 2: animals.php

O código do `animals.php` é executado após o preenchimento do formulário inicial. A primeira verificação é se o cliente introduzido existe. Não havendo a possibilidade de inserir novos clientes, no caso de se introduzir um cliente inválido obtém-se o resultado da Figura 2.

Animals

There is no client with this VAT.

Go back to homepage

Homepage

Figura 2: Caso em que não existe o cliente inserido

Caso o cliente introduzido já tenha ido a consultas com animais que tenham o nome introduzido no formulário, assim como um dono com uma porção do seu nome igual à introduzida, obtém-se uma tabela com toda a informação acerca dos animais em questão – Figura 3. De notar que podem aparecer vários animais, sendo todos eles apresentados.

Animals

You chose VAT Client: 00000002, Animal Name: Blue, Owner Name: o

Results of the intersection of the 3 parameters:

Owner name	VAT Owner	Animal name	Species	Colour	Gender	Birthday	Age	
John Doe	00000002	Blue	Bird	Blue	Female	2009-02-05	7	View animal
Oliver Watts	00000003	Blue	Bird	Blue	Female	2009-02-05	7	View animal

Go back to homepage

Homepage

Figura 3: Caso em que se encontram dois animais distintos que o cliente introduzido já levou a consulta

Se a interseção dos três parâmetros iniciais for nula (ou seja, se este cliente não foi a nenhuma consulta com o animal inserido ou o animal inserido não existe na base de dados), é necessário introduzir um novo animal na base de dados e, para isso, é apresentado um formulário, para preencher com as características do animal. Assume-se que o dono do animal

é o cliente que o leva à consulta (pelo que este parâmetro não é pedido no formulário mas sim logo apresentado). Os parâmetros passados por POST da página inicial para a dos animais, são apresentados logo no início.

Realça-se que apesar de não ter sido encontrado nenhum resultado, apresenta-se uma tabela com sugestões de animais. Esta tabela apresenta todos os animais levados a uma consulta pelo cliente inserido, assim como os animais que satisfazem os restantes parâmetros (nome e parte do nome do dono). Esta lista tem como objetivo permitir a seleção quando o cliente se engana num dos nomes mas já levou o animal ou quando o animal está corretamente identificado mas nunca foi acompanhado por este cliente.

Animals

You chose VAT Client: 00000000, Animal Name: Puma, Owner Name: John

The intersection of the 3 parameters is empty!

Input the Animal information

VAT owner = 00000000

Colour:

Gender:

Birthday:

Species name:

Submit

Here are some alternatives where the animal and owner match the input or animals this client took to consults.

Owner name	VAT Owner	Animal name	Species	Colour	Gender	Birthday	Age	
John Smith	00000000	Puma	Bulldog	Black	Male	1997-12-18	19	View animal
John Smith	00000000	Theo	Siamese	Grey	Female	2000-12-19	4	View animal

Go back to homepage

Homepage

Figura 4: Caso em que o cliente não foi a uma consulta com o animal inserido ou o animal inserido não existe na base de dados

Após preenchido o formulário, executa-se o código do ficheiro `insertanimal.php`, que efetua a inserção do animal na base de dados. Aqui, considerou-se que o único parâmetro não obrigatório era a espécie do animal e teve-se em atenção que a data de nascimento não pode ser superior à atual. Todas as restrições possíveis de fazer pelo formulário (como o tipo de dados) foram tidas em conta, sendo o único parâmetro possível de ficar vazio a espécie do animal (pode não existir na base de dados ou ser desconhecida, por exemplo).

```

<html>
<body>
    <?php
        $host = "db.tecnico.ulisboa.pt";
        $user = "istxxxxxx";
        $pass = "xxxxxxx";
        $dsn = "mysql:host=$host;dbname=$user";
        try
        {
            $connection = new PDO($dsn, $user, $pass);
        }
        catch(PDOException $exception)
        {
            echo("<p>Error: ");
            echo($exception->getMessage());
            echo("</p>");
            exit();
        }
        $name = $_REQUEST['name'];
        $VAT_client=$_REQUEST['VAT_client'];
        $species_name = $_REQUEST['species_name'];
        if($species_name == "---")
        {
            $species_name = NULL;
        }
        $colour = $_REQUEST['colour'];
        $gender= $_REQUEST['gender'];
        $birth_year = $_REQUEST['birth_year'];
        $age=0;

        $sql = $connection->prepare("INSERT into animal values(:name,
            :VAT_owner, :species_name, :colour, :gender, :birth_year,
            $age)");
        if ($sql == FALSE) {
            $info = $sql->errorInfo();
            echo("<p>Error: {$info[2]}</p>");
            exit();
        }

        $test = $sql->execute([':name' => $name,
                                ':species_name'=>$species_name,
                                ':VAT_owner'=>$VAT_client,
                                ':colour'=>$colour,
                                ':gender'=>$gender,
                                ':birth_year'=>$birth_year]);

        if ($test == FALSE) {
            echo("<p>Are you trying to insert an existing animal?</p>");
            $info = $sql->errorInfo();
            echo("<p>Error: {$info[2]}</p>");
        }
        else
        {
            header("Location:
                consults.php?name=$name&VAT_owner=$VAT_client&VAT_client=$VAT_client");
        }
    ?>
    <form action='check.php' method='post'>

```



```

<h3>Go back to homepage</h3>
<p><input type='submit' value='Homepage' /></p>
</form>
</body>
</html>

```

Código 3: insertanimal.php

Caso o animal seja bem inserido, o utilizador é imediatamente redirecionado para a página de consultas: a ver na próxima secção. Por se registar uma consulta de seguida, a idade do animal começa por ser 0, uma vez que vai ser atualizada após a inserção da consulta.

Salienta-se que, caso se tente inserir um animal já existente (isto pode acontecer se, por exemplo, após a inserção se retroceder na página e se voltar a submeter ou se o animal que se tenta inserir já tiver sido inserido mas não foi a nenhuma consulta com o seu dono, que agora é o cliente), aparece uma página de erro controlado – Figura 5. Esta é a única página em que não se deve voltar atrás. No entanto, não será esperado o utilizador tentar esta ação pois, a seguir a inserir o animal, deverá registar uma nova consulta.

Are you trying to insert an existing animal?
 Error: Duplicate entry 'Puma-00000000' for key 'PRIMARY'

Figura 5: Resultado de uma inserção mal realizada

Ao longo de todo o projeto, de forma a prevenir *SQL Injection* recorreu-se ao uso de *Prepared Statements*. Com a função *Prepare*, o programa prepara o tipo de *query* a receber, pelo que desta forma o envio de comandos indesejados para a base de dados (que podiam, por exemplo, eliminar tabelas) é impossibilitado.

1.2 Consultas

```

<html>
<body>
  <h3>Consults</h3>
  <?php
    $host = "db.tecnico.ulisboa.pt";
    $user = "istxxxxxx";
    $pass = "xxxxxxx";
    $dsn = "mysql:host=$host;dbname=$user";
    try
    {
      $connection = new PDO($dsn, $user, $pass);
    }
    catch(PDOException $exception)
    {
      echo("<p>Error: ");
      echo($exception->getMessage());
      echo("</p>");
      exit();
    }

    $name = $_REQUEST['name'];
    $VAT_owner = $_REQUEST['VAT_owner'];
    $VAT_client = $_REQUEST['VAT_client'];
  
```

```

$date_timestamp = date('Y-m-d H:i:s', time());

$sql = $connection->prepare("SELECT date_timestamp FROM consult
    WHERE name = :name AND VAT_owner = :VAT_owner;");
if($sql == FALSE){
    $info = $connection->errorInfo();
    echo("<p>Error: {$info[2]}</p>");
    exit();
}

$sql->execute([':name' => $name, ':VAT_owner'=>$VAT_owner]);
$result=$sql->fetchAll();

echo("<table border=\"0\" cellspacing=\"5\">\n");
foreach($result as $row)
{
    $date = $row['date_timestamp'];
    echo("<tr>\n");
    echo("<td>{$row['date_timestamp']}</td>\n");
    echo("<td><a href=\"consultdetails.php?name=$name
        &VAT_owner=$VAT_owner&date_timestamp=$date\">
    echo("<\>View consult</a></td>\n");
    echo("<td><a href=\"test.php?name=$name
        &VAT_owner=$VAT_owner&date_timestamp=$date\">
    echo("<\>Add blood test</a></td>\n");
    echo("</tr>\n");
}
echo("</table>\n");
?>
<fieldset style='max-width:300px'><legend> Add another consult
    </legend>
<p>Name: <?=$name?></p>
<p>Owner VAT: <?=$VAT_owner?></p>
<p>Date and time: <?=$date_timestamp?></p>
<form action='addconsult.php' method='post'>
    <p>VAT veterinary doctor:
        <select name="VAT_vet">
            <?php
                $sql = "SELECT VAT FROM veterinary ORDER BY VAT";
                $result = $connection->query($sql);
                if ($result == FALSE)
                {
                    $info = $connection->errorInfo();
                    echo("<p>Error: {$info[2]}</p>");
                    exit();
                }
                foreach($result as $row)
                {
                    $VAT_vet = $row['VAT'];
                    echo("<option value=\" $VAT_vet \">$VAT_vet</option>");
                }
            ?>
        </select>
    </p>
    <input type='hidden' name='name' value='<?=$name?>' />
    <input type='hidden' name='date_timestamp'
        value='<?=$date_timestamp?>' />

```

```

<input type='hidden' name='VAT_owner' value='<?=$VAT_owner?>' />
<input type='hidden' name='VAT_client' value='<?=$VAT_client?>' />
<p>Weight: <input type='number' min="0.01" step="0.01"
name='weight' style="width:60px;" required /> kg</p>
<p>Subjective:</p>
<p><textarea type='text' style="width:250px;height:100px;"
name='s'></textarea></p>
<p>Objective:</p>
<p><textarea type='text' style="width:250px;height:100px;"
name='o'></textarea></p>
<p>Assessment:</p>
<p><textarea type='text' style="width:250px;height:100px;"
name='a'></textarea></p>
<p>Plan:</p>
<p><textarea type='text' style="width:250px;height:100px;"
name='p'></textarea></p>
<p>Assistants:</p>
<?php
$sql = "SELECT * FROM assistant";
$result = $connection->query($sql);
if ($result == FALSE)
{
    $info = $connection->errorInfo();
    echo("<p>Error: {$info[2]}</p>");
    exit();
}

foreach($result as $row)
{
    $vat = $row['VAT'];
    echo("<input type='checkbox' name='assistants[]'
value='$vat'>$vat<br/>");
}
?>
<p>Diagnosis:</p>
<?php
$sql = "SELECT * FROM diagnosis_code";
$result = $connection->query($sql);
if ($result == FALSE)
{
    $info = $connection->errorInfo();
    echo("<p>Error: {$info[2]}</p>");
    exit();
}

foreach($result as $row)
{
    $code = $row['code'];
    $dname = $row['name'];
    echo("<input type='checkbox' name='diagnosis[]'
value='$code'>$dname<br/>");
}

$connection = null;
?>
<p><input type='submit' value='Submit'></p>
</fieldset>

```

```

</form>
<form action='check.php' method='post'>
<h3>Go back to homepage</h3>
<p><input type='submit' value='Homepage' /></p>
</body>
</html>

```

Código 4: consults.php

Para mostrar a lista de consultas associadas a um animal, recebem-se os seus parâmetros através de um GET (pela hiperligação), sendo estes utilizados para executar uma *query* à base de dados que lista as datas de todas as consultas, tal como apresentado na Figura 6, tendo-se utilizado o segundo animal da Figura 3.

Consults

```

2016-11-29 09:00:00 View consult Add blood test
2018-12-06 13:53:03 View consult Add blood test

```

Figura 6: Listagem das consultas de um animal

```

<html>
<body>
  <?php
    $host = "db.tecnico.ulisboa.pt";
    $user = "istxxxxxx";
    $pass = "xxxxxxx";
    $dsn = "mysql:host=$host;dbname=$user";
    try
    {
      $connection = new PDO($dsn, $user, $pass);
    }
    catch(PDOException $exception)
    {
      echo("<p>Error: ");
      echo($exception->getMessage());
      echo("</p>");
      exit();
    }

    $name = $_REQUEST['name'];
    $date_timestamp= $_REQUEST['date_timestamp'];
    $VAT_vet = $_REQUEST['VAT_vet'];
    $VAT_client=$_REQUEST['VAT_client'];
    $VAT_owner=$_REQUEST['VAT_owner'];
    $weight= $_REQUEST['weight'];
    $s = $_REQUEST['s'];
    $o=$_REQUEST['o'];
    $a=$_REQUEST['a'];
    $p=$_REQUEST['p'];
    $diagnosis= $_REQUEST['diagnosis'];
    $assistants= $_REQUEST['assistants'];

    $sql = $connection->prepare("INSERT into consult values(:name,
      :VAT_owner, :date_timestamp, :s, :o, :a, :p, :VAT_client,
      :VAT_vet, :weight);");

```

```

if($sql == FALSE){
    $info = $connection->errorInfo();
    echo("<p>Error: {$info[2]}</p>");
    exit();
}

$test = $sql->execute([':name' => $name,
    ':VAT_owner'=>$VAT_owner,
    ':date_timestamp'=>$date_timestamp,
    ':s'=>$s,
    ':o'=>$o,
    ':a'=>$a,
    ':p'=>$p,
    ':VAT_client'=>$VAT_client,
    ':VAT_vet'=>$VAT_vet,
    ':weight'=>$weight]);

if($test == FALSE){
    $info = $connection->errorInfo();
    echo("<p>Error: {$info[2]}</p>");
    exit();
}

if(!empty($assistants)){

    $sql = $connection->prepare("INSERT into participation
        values(:name, :VAT_owner, :date_timestamp, :VAT_assistant);");
    if($sql == FALSE){
        $info = $connection->errorInfo();
        echo("<p>Error: {$info[2]}</p>");
        exit();
    }

    foreach ($assistants as $vat) {
        $test = $sql->execute([':name' => $name,
            ':VAT_owner'=>$VAT_owner,
            ':date_timestamp'=>$date_timestamp,
            ':VAT_assistant'=>$vat]);

        if($test == FALSE){
            $info = $connection->errorInfo();
            echo("<p>Error: {$info[2]}</p>");
            exit();
        }
    }
}

if(!empty($diagnosis)){

    $sql = $connection->prepare("INSERT into consult_diagnosis
        values(:code,:name,:VAT_owner,:date_timestamp);");
    if($sql == FALSE){
        $info = $connection->errorInfo();
        echo("<p>Error: {$info[2]}</p>");
        exit();
    }

    foreach ($diagnosis as $code) {

```

```

        $test = $sql->execute([':name' => $name,
                                ':VAT_owner'=>$VAT_owner,
                                ':date_timestamp'=>$date_timestamp,
                                ':code'=>$code]);

        if($test == FALSE){
            $info = $connection->errorInfo();
            echo("<p>Error: {$info[2]}</p>");
            exit();
        }
    }
}

if($sql == TRUE && $test == TRUE){
    header("Location:
        consults.php?name=$name&VAT_owner=$VAT_owner&VAT_client=$VAT_client");
}
$connection = null;
?>
</body>
</html>

```

Código 5: addconsult.php

Caso não houvesse nenhum registo de consultas deste animal, esta lista não seria apresentada. Em todo o caso, é sempre possível inserir uma nova consulta, a partir do formulário da Figura 7, no qual apenas é necessário preencher o VAT do veterinário e o peso medido, sendo a data considerada a atual.

Add another consult

Name: Blue

Owner VAT: 00000003

Date and time: 2018-12-06 13:53:19

VAT veterinary doctor:

Weight: kg

Subjective:

Objective:

Assessment:

Plan:

Assistants:

☐ 00000010

☐ 00000012

☐ 00000016

☐ 00000019

Diagnosis:

☐ High Blood Pressure

☐ Kidney Failure

☐ Cron disease

☐ Flu

☐ Fever

☐ Broken Nose

☐ End-stage renal disease

Go back to homepage

(a) Início da página
(b) Fim da página

Figura 7: Formulário para nova consulta

Após a inserção da consulta, a página é recarregada, tendo a lista sido atualizada por forma a apresentar a nova consulta, tal como se pode ver pela Figura 8

Consults

2016-11-29 09:00:00 [View consult](#) [Add blood test](#)
 2018-12-06 13:53:03 [View consult](#) [Add blood test](#)
 2018-12-06 13:53:19 [View consult](#) [Add blood test](#)

Figura 8: Listagem das consultas de um animal após uma inserção

Finalmente, poderia ainda ser adicionado aqui um outro *link* para aceder aos procedimentos associados a uma certa consulta. No entanto, como esta funcionalidade não é pedida, optou-se por apenas ter esta informação na base de dados por, por exemplo, poder ser consultada apenas pelo médico.

```
<html>
<body>
  <h3>Consult: <?=$_REQUEST['date_timestamp']?></h3>
  <?php
    $host = "db.tecnico.ulisboa.pt";
    $user = "istxxxxxx";
    $pass = "xxxxxxx";
    $dsn = "mysql:host=$host;dbname=$user";
    try
    {
      $connection = new PDO($dsn, $user, $pass);
    }
    catch(PDOException $exception)
    {
      echo("<p>Error: ");
      echo($exception->getMessage());
      echo("</p>");
      exit();
    }

    $name = $_REQUEST['name'];
    $VAT_owner = $_REQUEST['VAT_owner'];
    $date_timestamp = $_REQUEST['date_timestamp'];

    $sql = $connection->prepare("SELECT * FROM animal WHERE name =
      :name AND VAT = :VAT_owner;");
    if($sql == FALSE){
      $info = $connection->errorInfo();
      echo("<p>Error: {$info[2]}</p>");
      exit();
    }
    $sql->execute([':name' => $name, ':VAT_owner'=>$VAT_owner]);
    $row=$sql->fetch();
    ?>

    <table border="0" cellspacing="5">
    <tr><td>Name: <?=$row['name']?></td></tr>
    <tr><td>Owner's VAT: <?=$row['VAT']?></td></tr>
    <tr><td>Species: <?=$row['species_name']?></td></tr>
    <tr><td>Gender: <?=$row['gender']?></td></tr>
```

```
|  |  |  |  |  |  |  |  |  |  | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Colour: <?=$row['colour']?></td></tr> |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | Age: <?=$row['age']?></td></tr> |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | Birthday: <?=$row['birth_year']?></td></tr>  <?php $sql = $connection->prepare("SELECT s, o, a, p, VAT_client,     VAT_vet, weight     FROM consult WHERE name = :name AND VAT_owner = :VAT_owner     AND date_timestamp = :date_timestamp;"); if($sql == FALSE){     $info = $connection->errorInfo();     echo("<p>Error: {$info[2]}</p>");     exit(); } $sql->execute([':name' => $name, ':VAT_owner'=>$VAT_owner, '     :date_timestamp'=>$date_timestamp]); $row=$sql->fetch(); ?>  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | Vet's VAT: <?=$row['VAT_vet']?></td></tr> |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Client's VAT: <?=$row['VAT_client']?></td></tr> |  |  |  |  |  | | --- | --- | --- | --- | --- | | Weight: <?=$row['weight']?></td></tr> </table> <h4>SOAP notes</h4> <table border="0" cellspacing="5"> |  |  |  |  | | --- | --- | --- | --- | | Subjective: <?=$row['s']?></td></tr> |  |  |  | | --- | --- | --- | | Objective: <?=$row['o']?></td></tr> |  |  | | --- | --- | | Assessment: <?=$row['a']?></td></tr> |  | | --- | | Plan: <?=$row['p']?></td></tr> </table>  <?php $sql = $connection->prepare("SELECT VAT_assistant FROM participation     WHERE name = :name AND VAT_owner = :VAT_owner AND date_timestamp     = :date_timestamp;"); if($sql == FALSE){     $info = $connection->errorInfo();     echo("<p>Error: {$info[2]}</p>");     exit(); }  $sql->execute([':name' => $name, ':VAT_owner'=>$VAT_owner, '     :date_timestamp'=>$date_timestamp]);  $result = $sql->fetchAll();  if ($result != FALSE) {     echo("<h4>Assistants:</h4>\n");     foreach($result as $row)     {         echo("<table border=\"0\" cellspacing=\"5\">\n");         echo("<tr><td>{$row['VAT_assistant']}</td></tr>\n");     }     echo("</table>\n"); } | | | | | | | | | |

```



```

$sql = $connection->prepare("SELECT code, d.name FROM
    consult_diagnosis
    inner join diagnosis_code as d using(code) WHERE
        consult_diagnosis.name = :name
    AND VAT_owner = :VAT_owner AND date_timestamp =
        :date_timestamp;");
if($sql == FALSE){
    $info = $connection->errorInfo();
    echo("<p>Error: {$info[2]}</p>");
    exit();
}
$sql->execute([':name' => $name, ':VAT_owner'=>$VAT_owner, '
    :date_timestamp'=>$date_timestamp]);

$result=$sql->fetchAll();

if ($result != FALSE)
{
    echo("<h4>Diagnosis codes:</h4>\n");
    foreach($result as $row)
    {
        echo("<table border=\"0\" cellpadding=\"5\">\n");
        echo("<tr><td>{$row['code']} &ndash;
            {$row['name']}</td></tr>\n");
    }
    echo("</table>\n");
}

$sql = $connection->prepare("SELECT code, name_med, lab, dosage,
    regime
    FROM prescription WHERE name = :name
    AND VAT_owner = :VAT_owner AND date_timestamp =
        :date_timestamp;");
if($sql == FALSE){
    $info = $connection->errorInfo();
    echo("<p>Error: {$info[2]}</p>");
    exit();
}
$sql->execute([':name' => $name, ':VAT_owner'=>$VAT_owner, '
    :date_timestamp'=>$date_timestamp]);
$result=$sql->fetchAll();

if ($result != FALSE)
{
    echo("<h4>Prescriptions for each diagnosis:</h4>\n");
    foreach($result as $row)
    {
        echo("<table border=\"0\" cellpadding=\"5\">\n");
        echo("<tr><td>{$row['code']}: {$row['name_med']} |
            {$row['lab']} | {$row['dosage']} |
            {$row['regime']}</td></tr>\n");
    }
    echo("</table>\n");
}

$connection = null;
?>

```

```

<form action='test.php' method='post'>
</form>
<form action='check.php' method='post'>
<h3>Go back to homepage</h3>
<p><input type='submit' value='Homepage' /></p>
</body>
</html>

```

Código 6: consultdetails.php

Finalmente, pode-se selecionar, por exemplo, o *link View consult* da primeira opção da Figura 8. Assim, obtém-se a página que apresenta todos os detalhes dessa consulta, apresentada na Figura 9, passando a informação do animal e a data da consulta através de um GET.

Consult: 2016-11-29 09:00:00	Assistants:
Name: Blue	00000010
Owner's VAT: 00000003	00000012
Species: Bird	
Gender: Female	Diagnosis codes:
Colour: Blue	1111 – Kidney Failure
Age: 7	2222 – Cron disease
Birthday: 2009-02-05	5555 – Broken Nose
Vet's VAT: 00000001	
Client's VAT: 00000020	Prescriptions for each diagnosis:
Weight: 1.00	1111: Ben-u-ron Ben-u-ron Inc. 1000.00 After eating.
SOAP notes	2222: Gaviscom Gaviscom Inc. 800.00 One in the morning.
Subjective: Subjective	Go back to homepage
Objective: Objective	<input type="button" value="Homepage"/>
Assessment: Assessment	
Plan: Plan	

(a) Início da página

(b) Fim da página

Figura 9: Detalhes da primeira consulta do Blue

Quando se observa uma consulta sem assistentes, diagnósticos ou prescrições associados, as respectivas secções desta página são suprimidas, o que se pode ver se se abrir a outra consulta deste animal – Figura 10.

Consult: 2016-12-30 09:00:00

Name: Blue
Owner's VAT: 00000003
Species: Bird
Gender: Female
Colour: Blue
Age: 7
Birthday: 2009-02-05
Vet's VAT: 00000001
Client's VAT: 00000002
Weight: 1.00

SOAP notes

Subjective: s
Objective: o
Assessment: a
Plan: p

Go back to homepage

[Homepage](#)

Figura 10: Listagem das consultas de um animal após uma inserção

1.3 Teste de sangue

```
<html>
<body>
  <?php
    $host = "db.tecnico.ulisboa.pt";
    $user = "istxxxxxx";
    $pass = "xxxxxxx";
    $dsn = "mysql:host=$host;dbname=$user";
    try
    {
      $connection = new PDO($dsn, $user, $pass);
    }
    catch(PDOException $exception)
    {
      echo("<p>Error: ");
      echo($exception->getMessage());
      echo("</p>");
      exit();
    }

    $VAT_owner= $_REQUEST['VAT_owner'];
    $name= $_REQUEST['name'];
    $date_timestamp= $_REQUEST['date_timestamp'];

    $sql="SELECT VAT FROM assistant";
    $result=$connection->query($sql);
```

```

if ($result == FALSE)
{
    $info = $connection->errorInfo();
    echo("<p>Error: {$info[2]}</p>");
    exit();
}
$assistants=$result->fetchAll();

$sql="SELECT name, units FROM indicator";
$result=$connection->query($sql);
if ($result == FALSE)
{
    $info = $connection->errorInfo();
    echo("<p>Error: {$info[2]}</p>");
    exit();
}
$indicators=$result->fetchAll();

$connection = null;
?>
<h3>Blood Test</h3><br>
<form action="act.php" method="post">
<input type="hidden" id="vatowner" name="vatowner"
    value='<?=$VAT_owner?>'>
<input type="hidden" id="name" name="name" value='<?=$name?>'>
<input type="hidden" id="date" name="date"
    value='<?=$date_timestamp?>'>
<fieldset style='max-width:400px'><legend> Results of the Blood
    Tests </legend>
<p><label for="vatassistant"> VAT of the assistant:</label></p>
<?php
foreach($assistants as $row){
    $vat=$row["VAT"];
    echo("<input type='checkbox' name='assistants[]'
        value='$vat'>$vat<br/>");
}
echo("<p>Description:</p><p><textarea type='text'
    style='width:250px;height:100px;'
    name='description'></textarea></p>");
foreach($indicators as $row){
    $indicator = $row["name"];
    $units = $row["units"];
    echo("<p>$indicator: <input type='number' min='0'
        style='width:60px;' name='indicator[$indicator]'>
        $units</p>");
}
?>
<input type="submit" value="Submit">
</fieldset>
</form>
<form action='check.php' method='post'>
<h3>Go back to homepage</h3>
<p><input type='submit' value='Homepage'></p>
</body>
</html>

```

Código 7: test.php

Para se inserir um teste de sangue, o utilizador terá de carregar no *link Add blood test*, da página da Figura 8, correspondente à consulta associada aos testes. Ao clicar neste *link*, é-se redirecionado para a página `test.php` (passando os dados da consulta correspondente por GET), onde existe um formulário para preencher os dados do teste de sangue. O resultado da entrada nesta página pode ser visto na Figura 11.

Blood Test

Results of the Blood Tests

VAT of the assistant:

☐ 00000010
☐ 00000012
☐ 00000016
☐ 00000019

Description:

Cholesterol:

milligrams

Creatinine level:

milligrams per deciliter

Erythrocytes:

grams

Fatty acids:

milligrams

Insulin:

milligrams

Red Blood Cells:

grams

White Blood Cells:

milligrams

Submit

Go back to homepage

[Homepage](#)

Figura 11: Inserção dos resultados

A interpretação dada ao enunciado, quando é dito que este formulário é fixo, foi que o número de indicadores não deve variar, isto é, não aparece um segundo indicador quando se preenche o primeiro. Como tal, considerou-se que neste formulário deveriam ser apresentados todos os indicadores existentes na base de dados, não sendo preenchidos os que não foram medidos.

Repare-se que tanto poderá não ser selecionado nenhum, um ou múltiplos assistentes e que nem todos os indicadores têm de ser preenchidos. Os indicadores em branco são enviados como parâmetros a *null* e posteriormente tratados como se não tivessem sido introduzidos no

formulário. A inserção do valor 0 é válida e é tratada como a inserção de um valor diferente de *null*, pois se considera que um indicador medido pode ser efetivamente zero, não podendo apenas ser negativo.

Quanto aos assistentes, é importante referir que são apresentados todos os assistentes do hospital porque se considera que no momento de inserção da consulta poderá ainda não se saber quem fará os exames (que poderão ocorrer apenas dias depois).

Todos os valores, ao serem submetidos, são enviado para a página `act.php` por POST, sendo a sua utilização analisada de seguida.

```
<html>
<body>
    <?php
        $host = "db.tecnico.ulisboa.pt";
        $user = "istxxxxxx";
        $pass = "xxxxxxx";
        $dsn = "mysql:host=$host;dbname=$user";
        try
        {
            $connection = new PDO($dsn, $user, $pass);
        }
        catch(PDOException $exception)
        {
            echo("<p>Error: ");
            echo($exception->getMessage());
            echo("</p>");
            exit();
        }

        $VAT_owner = $_REQUEST['vatowner'];
        $name = $_REQUEST['name'];
        $date_timestamp = $_REQUEST['date'];
        $assistants = $_REQUEST['assistants'];
        $indicators=$_REQUEST['indicator'];
        $description=$_REQUEST['description'];

        $connection->beginTransaction();

        $ok = 0;
        while($ok == 0){

            $sql = $connection->prepare("SELECT max(num) as max FROM proced
                WHERE name = :name AND VAT_owner = :VAT_owner AND
                date_timestamp = :date_timestamp;");
            if($sql == FALSE){
                break;
            }

            $test = $sql->execute([':name' => $name, '
                :VAT_owner'=>$VAT_owner, ':date_timestamp'=>$date_timestamp]);
            if($test == FALSE){
                break;
            }

            $result = $sql->fetch();
            if($result['max'] == NULL){
                $number = 1;
            }
        }
    }

```

```

else{
    $number = $result['max'] + 1;
}

$sql= $connection->prepare("INSERT INTO proced VALUES(:name,
:VAT_owner, :date_timestamp, :num, :description);");
if($sql == FALSE){
    break;
}

$test = $sql->execute([':name' => $name, '
:VAT_owner'=>$VAT_owner, ':date_timestamp'=>$date_timestamp, '
:num'=>$number, ':description'=>$description]);
if($test == FALSE){
    break;
}

$sql= $connection->prepare("INSERT INTO test_procedure
VALUES(:name, :VAT_owner, :date_timestamp, :num, 'blood');");
if($sql == FALSE){
    break;
}

$test = $sql->execute([':name' => $name, '
:VAT_owner'=>$VAT_owner, ':date_timestamp'=>$date_timestamp, '
:num'=>$number ]]);
if($test == FALSE){
    break;
}

$ninsert=0;
foreach($indicators as $nome => $value){
    echo("<p>");
    if($value != null){
        $sql = $connection->prepare("INSERT INTO produced_indicator
VALUES(:name, :VAT_owner, :date_timestamp, :num,
:indicator_name, :indicator_value);");
        if($sql == FALSE){
            break 2;
        }

        $test = $sql->execute([':name' => $name, '
:VAT_owner'=>$VAT_owner, '
:date_timestamp'=>$date_timestamp, ':num'=>$number, '
:indicator_name'=>$nome, ':indicator_value'=>$value]);
        if($test == FALSE){
            break 2;
        }
        $ninsert++;
    }
    echo("</p>");
}

if ($ninsert==0){
    echo("No data was given so nothing was inserted in the
database.");
    $connection->rollback;
}

```

```

        $ok = -1;
        break;
    }

    foreach ($assistants as $VAT_assistant){
        $sql = $connection->prepare("INSERT INTO performed
            VALUES(:name, :VAT_owner, :date_timestamp, :num,
                :VAT_assistant);");
        if($sql == FALSE){
            break;
        }

        $test = $sql->execute([':name' => $name, '
            :VAT_owner'=>$VAT_owner, ':date_timestamp'=>$date_timestamp,
            ':num'=>$number, ':VAT_assistant'=>$VAT_assistant]);
        if($test == FALSE)
            break;
    }
    $connection->commit();
    $ok = 1;
}

if($ok == 0){
    echo("<p>There was a problem with the insertion. Try again.</p>");
    echo("<p>Error: {$info[2]}</p>");
}else if($ok == 1){
    echo("<p>Successful insertion");
    $sql = $connection->prepare("SELECT indicator_name, value FROM
        produced_indicator
        WHERE name = :name AND VAT_owner = :VAT_owner AND
            date_timestamp = :date_timestamp AND num = :num;");
    if($sql == FALSE){
        echo(" but we are unable to show you the results.</p>");
        echo("<p>Error: {$info[2]}</p>");
    } else {
        $sql->execute([':name' => $name, ':VAT_owner'=>$VAT_owner, '
            :date_timestamp'=>$date_timestamp, ':num'=>$number]);

        echo(". Review your results:</p>");
        $results = $sql->fetchAll();
        echo("<table border='1' cellpadding='4'>");
        echo("<tr><td>Indicator</td><td>Value</td></tr>");
        foreach($results as $row)
        {
            echo("<tr><td>");
            echo($row['indicator_name']);
            echo("</td><td>");
            echo($row['value']);
            echo("</td></tr>\n");
        }
        echo("</table>");
    }
}
}
$connection = null;
?>
<form action='check.php' method='post'>
    <h3>Go back to homepage</h3>

```



```
<p><input type='submit' value='Homepage' /></p>
</body>
</html>
```

Código 8: `act.php`

Ao serem inseridos valores válidos, o utilizador é direcionado para a página `act.php`. Após serem passados os parâmetros recolhidos, inicia-se a transação e um ciclo `while`, que possibilita a utilização do comando `break` na eventualidade de haver algum erro e que nunca será percorrido mais do que uma vez.

Para escolher o número associado ao novo *procedure*, procura-se o maior número (da tabela *procedure*) associado a esta consulta, escolhendo-se o número seguinte. De seguida, são efetuadas as inserções nas tabelas `proced`, `test_procedure` e `produced_indicator`, que guardam, respetivamente, informação sobre procedimentos por consulta, informação sobre os tipos de procedimentos (neste caso são todos ao sangue) e os valores dos indicadores obtidos. É de notar que é na inserção dos indicadores que são verificados quantos indicadores foram de facto preenchidos na página anterior. Caso o formulário não tenha nenhum indicador preenchido, é anulada a operação (feito *rollback*), informando-se o utilizador – Figura^o12. Por fim, o programa inicia a adição dos assistentes selecionados na tabela `performed`.

No data was given so nothing was inserted in the database.

Go back to homepage

Homepage

Figura 12: Nenhum indicador preenchido

Se o ciclo `while` chegar ao fim (com sucesso), faz-se uma *query* para apresentação dos valores inseridos na tabela *produced_indicator*. Assim o utilizador poderá rever os valores introduzidos. O resultado pode ser observado na Figura 13.

Successfull insertion. Review your results:

Indicator	Value
Cholesterol	4.00
Fatty acids	1.00
Red Blood Cells	5.00

Go back to homepage

Homepage

Figura 13: Confirmação dos resultados

2 *Triggers, functions e stored procedures*

2.1 Atualizar idade

```
drop trigger if exists update_age;
delimiter $$
create trigger update_age after insert on consult
for each row
begin
    declare max_date date;

    select LEAST(NOW(), max(date_timestamp)) into max_date
    from consult
    where name=new.name and VAT_owner=new.VAT_owner;

    update animal set age = timestampdiff(YEAR, birth_year, max_date),
        animal.name=animal.name, animal.VAT=animal.VAT
    where animal.name=new.name and animal.VAT=new.VAT_owner;

end$$
delimiter ;
```

Código 9: *Trigger* 1

Para a atualização da idade sempre que é introduzida uma nova consulta para um dado animal, teve-se em atenção o caso de se inserir uma data de consulta inferior à de outra já registada, calculando-se a data da consulta mais recente para fazer a atualização, e ainda se, por engano, se tivesse introduzido uma data superior à data atual numa consulta, fazendo-se o mínimo entre a data máxima das consultas e a data atual. Para o primeiro caso, tem de se considerar a consulta com a data mais recente. Para o segundo, considera-se a data atual. Este resultado é confirmado na Figura 14: o Bonzo inicialmente tinha 10 anos e, ao ser inserida uma consulta mais recente, esta é atualizada para 11. Destaca-se que a idade seria na verdade 12 anos, mas a consulta é de 2017 e não da data atual, pelo que se verifica que o *trigger* funciona corretamente.

```
MySQL [ist425496]> select name, age, birth_year
-> from animal where name = 'Bonzo';
+-----+-----+-----+
| name | age | birth_year |
+-----+-----+-----+
| Bonzo | 10 | 2006-05-29 |
+-----+-----+-----+
1 row in set (0.01 sec)

MySQL [ist425496]> insert into consult values
-> ('Bonzo', '00000002', '2017-12-6 10:30:00.75', '', '', '', ''),
-> ('00000020', '00000001', 35);
Query OK, 1 row affected (0.02 sec)

MySQL [ist425496]> select name, age, birth_year
-> from animal where name = 'Bonzo';
+-----+-----+-----+
| name | age | birth_year |
+-----+-----+-----+
| Bonzo | 11 | 2006-05-29 |
+-----+-----+-----+
1 row in set (0.01 sec)
```

Figura 14: Verificação do primeiro *trigger*

2.2 Apenas veterinário ou assistente

```
drop trigger if exists check_vet_insert;
delimiter $$
create trigger check_vet_insert before insert on veterinary
for each row
begin

    if (new.VAT in (select VAT from assistant)) then
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'This person is already
            an assistant';
    end if;
end$$
delimiter ;

drop trigger if exists check_vet_update;
delimiter $$
create trigger check_vet_update before update on veterinary
for each row
begin

    if new.VAT in (select VAT from assistant) then
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'This person is already
            an assistant';
    end if;
end$$
delimiter ;

drop trigger if exists check_assistant_insert;
delimiter $$
create trigger check_assistant_insert before insert on assistant
for each row
begin

    if new.VAT in (select VAT from veterinary) then
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'This person is already
            a veterinary';
    end if;
end$$
delimiter ;

drop trigger if exists check_assistant_update;
delimiter $$
create trigger check_assistant_update before update on assistant
for each row
begin

    if new.VAT in (select VAT from veterinary) then
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'This person is already
            a veterinary';
    end if;
end$$
delimiter ;
```

Código 10: *Trigger 2*

Para evitar a existência de veterinários que são assistentes, criaram-se quatro *triggers* simples, dois para cada tabela, dos quais um visa a correção de inserções e o outro a de atualizações.

Realça-se a importância dos *triggers* serem colocados antes dos comandos e não depois, por forma a ainda se conseguir evitar a inserção. Estes consistem simplesmente em verificar que o VAT a inserir/atualizar não estão já na outra tabela. Pode-se verificar facilmente que é emitido um erro quando se tenta fazer uma destas inserções – Figura 16 – verificando-se não só a mensagem de erro como a inalteração dos registos.

VAT_vet		VAT_assistant
00000001		
00000004		
00000005		
00000008		
00000009		
00000011		
00000013		
		00000010
		00000012
		00000016
		00000019

(a) Veterinários (b) Assistentes

Figura 15: Listas de veterinários e assistentes do hospital

```
MySQL [ist425496]> insert into assistant values ('00000001');
ERROR 1644 (45000): This person is already a veterinary
MySQL [ist425496]> select VAT from assistant;
```

VAT
00000010
00000012
00000016
00000019

4 rows in set (0.01 sec)

Figura 16: Verificação do segundo *trigger*

2.3 Telemóveis únicos

```
drop trigger if exists check_phone_insert;
delimiter $$
create trigger check_phone_insert before insert on phone_number
for each row
begin
    if new.phone in (select phone from phone_number) then
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Phone number already
        exists';
    end if;
end$$
delimiter ;

drop trigger if exists check_phone_update;
delimiter $$
create trigger check_phone_update before update on phone_number
for each row
```

```

begin

    if new.phone in (select phone from phone_number where VAT !=
        old.VAT) then
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Phone number already
            exists';
    end if;

end$$
delimiter ;

```

Código 11: *Trigger 3*

Este *trigger* é bastante semelhante ao anterior, agora realizado apenas para uma tabela, de forma a evitar a inserção múltipla do mesmo número de telemóvel. Realça-se que se teve em atenção o caso em que se pretende modificar um VAT sem se modificar o número de telemóvel (isto é, o telemóvel de uma pessoa passa a estar associado a outra). Para isso, não se conta com o número de telemóvel se esta for a entrada que se está a atualizar. Como se pode ver na Figura 17, percebe-se que, tentando inserir um número que já existia na tabela (o primeiro), é emitido um erro e a tabela permanece inalterada.

```

MySQL [ist425496]> select * from phone_number;

```

VAT	phone
00000000	934368287
00000001	963154632
00000002	926842214
00000003	913546541
00000005	933987048
00000006	966984516
00000007	925424700
00000008	917813568
00000009	933534684
00000010	927132454
00000010	938713541
00000010	968103912
00000013	926846513
00000014	936841324
00000015	961351153
00000019	918468344
00000020	910360480

```

17 rows in set (0.01 sec)

```

```

MySQL [ist425496]> insert into phone_number values
-> ('00000001', 934368287);
ERROR 1644 (45000): Phone number already exists
MySQL [ist425496]> select * from phone_number;
+-----+-----+
| VAT      | phone      |
+-----+-----+
| 00000000 | 934368287 |
| 00000001 | 963154632 |
| 00000002 | 926842214 |
| 00000003 | 913546541 |
| 00000005 | 933987048 |
| 00000006 | 966984516 |
| 00000007 | 925424700 |
| 00000008 | 917813568 |
| 00000009 | 933534684 |
| 00000010 | 927132454 |
| 00000010 | 938713541 |
| 00000010 | 968103912 |
| 00000013 | 926846513 |
| 00000014 | 936841324 |
| 00000015 | 961351153 |
| 00000019 | 918468344 |
| 00000020 | 910360480 |
+-----+-----+
17 rows in set (0.01 sec)

```

Figura 17: Verificação do terceiro *trigger*

2.4 Número total de consultas

```

drop function if exists total_nr_consults;

delimiter $$
create function total_nr_consults(a_name varchar(255), a_vat
    varchar(255), a_year integer)
returns integer
begin
    declare total integer;

    select count(*) into total
    from consult
    where year(date_timestamp) = a_year and name = a_name and VAT_owner
        = a_vat;

    return total;
end $$
delimiter ;

```

Código 12: *Function*

Para criar a função que conta o número de consultas de um animal num dado ano, basta fazer esse `select` à tabela de consultas, filtrando pelos parâmetros passados pelo utilizador e fazer a contagem das entradas. A Figura 18 confirmam o bom funcionamento da função, pois as contagens do número de consultas em 2016 para cada animal foram bem executadas.

name	VAT_owner	date_timestamp
Blue	00000002	2016-11-30 09:00:00
Severin	00000015	2005-07-31 09:00:00
Severin	00000015	2005-08-29 09:00:00
Severin	00000015	2016-07-28 11:00:00
Severin	00000015	2016-07-29 11:45:00
Severin	00000015	2018-07-30 09:00:00
Spark	00000003	2005-08-29 09:00:00
Spark	00000003	2017-08-29 11:30:00
Theo	00000000	2005-10-29 15:30:00
Toby	00000006	2005-09-29 17:40:00
Trojan	00000006	2005-10-29 16:30:00
Trojan	00000006	2017-10-29 09:30:00
Wanikiy	00000015	2005-02-27 09:00:00
Puma	00000000	2017-08-27 09:00:00
Puma	00000000	2017-07-27 10:30:00
Nugget	00000005	2018-01-10 14:00:00
Blue	00000003	2016-11-29 09:00:00
Blue	00000003	2016-12-30 09:00:00
Bolt	00000007	2005-07-29 09:00:00
Bonzo	00000002	2005-03-29 09:00:00
Bonzo	00000002	2017-03-29 10:30:00
Cally	00000014	2005-01-29 09:00:00
Doggy	00000014	2005-09-29 12:30:00
Fluffy	00000002	2005-07-29 09:00:00
Foxy	00000005	2018-01-10 11:00:00
Garfield	00000005	2005-07-29 09:00:00
Goofy	00000005	2018-01-10 12:30:00
Marlie	00000006	2016-11-29 19:00:00
Wanikiy	00000015	2018-02-27 09:00:00

(a) Lista de consultas

```
MySQL [ist425496]> select name, VAT,
-> total_nr_consults(name, VAT, 2016) as count
-> from animal;
```

name	VAT	count
Puma	00000000	0
Theo	00000000	0
Blue	00000002	1
Bonzo	00000002	0
Fluffy	00000002	0
Blue	00000003	2
Spark	00000003	0
Foxy	00000005	0
Garfield	00000005	0
Goofy	00000005	0
Nugget	00000005	0
Marlie	00000006	1
Toby	00000006	0
Trojan	00000006	0
Bolt	00000007	0
Cally	00000014	0
Doggy	00000014	0
Severin	00000015	2
Wanikiy	00000015	0

19 rows in set (0.01 sec)

(b) Contagem de consultas

Figura 18: Confirmação da função

2.5 Alterar indicadores

```
drop procedure if exists change_reference;

delimiter $$
create procedure change_reference()
begin
update indicator left outer join produced_indicator on
    produced_indicator.indicator_name = indicator.name
set reference_value = reference_value*0.1,
    units = 'centigrams',
    value = value*0.1,
    VAT_owner = VAT_owner,
    date_timestamp = date_timestamp,
    num = num,
    produced_indicator.name = produced_indicator.name,
    produced_indicator.indicator_name =
        produced_indicator.indicator_name
where units='milligrams';

end $$
delimiter ;
```

Código 13: *Stored procedure*

Para este *stored procedure*, basta juntar as tabelas de indicadores de referência e medidos e filtrar pelos que têm como unidades de medição *milligrams*. Depois, repõem-se os valores que já lá estavam e não são alterados, fazendo nos outros as alterações pedidas. Esta atualização

pode ser confirmada pela Figura 19. Destaca-se que os indicadores medidos em *milligrams per deciliter* se mantêm iguais.

```
MySQL [ist425496]> select p.name, VAT_owner, indicator.name, reference_value, units, value
-> from indicator left outer join produced_indicator as p
-> on p.indicator_name = indicator.name;
```

name	VAT_owner	name	reference_value	units	value
NULL	NULL	Cholesterol	200.00	milligrams	NULL
Doggy	00000014	Creatinine level	0.60	milligrams per deciliter	0.70
Puma	00000000	Creatinine level	0.60	milligrams per deciliter	1.10
NULL	NULL	Erythrocytes	500.00	grams	NULL
NULL	NULL	Fatty acids	110.00	milligrams	NULL
NULL	NULL	Insulin	350.00	milligrams	NULL
Puma	00000000	Red Blood Cells	50.00	grams	500.00
Puma	00000000	White Blood Cells	100.00	milligrams	1.30
Puma	00000000	White Blood Cells	100.00	milligrams	1.10
Trojan	00000006	White Blood Cells	100.00	milligrams	1.40

```
10 rows in set (0.01 sec)
```

```
MySQL [ist425496]> call change_reference();
Query OK, 7 rows affected (0.01 sec)
```

```
MySQL [ist425496]> select p.name, VAT_owner, indicator.name, reference_value, units, value
-> from indicator left outer join produced_indicator as p
-> on p.indicator_name = indicator.name;
```

name	VAT_owner	name	reference_value	units	value
NULL	NULL	Cholesterol	20.00	centigrams	NULL
Doggy	00000014	Creatinine level	0.60	milligrams per deciliter	0.70
Puma	00000000	Creatinine level	0.60	milligrams per deciliter	1.10
NULL	NULL	Erythrocytes	500.00	grams	NULL
NULL	NULL	Fatty acids	11.00	centigrams	NULL
NULL	NULL	Insulin	35.00	centigrams	NULL
Puma	00000000	Red Blood Cells	50.00	grams	500.00
Puma	00000000	White Blood Cells	10.00	centigrams	0.13
Puma	00000000	White Blood Cells	10.00	centigrams	0.11
Trojan	00000006	White Blood Cells	10.00	centigrams	0.14

```
10 rows in set (0.01 sec)
```

Figura 19: Verificação do *stored procedure*