InstantOffice

Projecto de Bases de Dados – LEIC-A 2016/2017

Parte 3 – Relatório de Aplicação Web

Profº Responsável: Gabriel Pestana

Turno: BD225179L09 – Grupo 2

81045	Rui Ventura	Tempo dedicado: 13h
81586	Diogo Freitas	Tempo dedicado: 3h
81700	Sara Azinhal	Tempo dedicado: 13h

Introdução

Consta neste relatório o conjunto de consultas(*queries*), juntamente com *triggers* que permitem definir as restrições de integridade pretendidas. Foram omitidos os ficheiros correspondentes ao esquema e população da base de dados, tendo sido utilizados os ficheiros fornecidos pelo corpo docente.

SQL

a) Quais os espaços com postos que nunca foram alugados?

```
SELECT DISTINCT morada, codigo_espaco
FROM Posto
WHERE (morada, codigo) NOT IN (
  SELECT DISTINCT morada, codigo
  FROM Aluga
);
```

b) Quais os edifícios com um número de reservas superior à média?

```
SELECT morada
FROM (
  SELECT morada, COUNT(1) AS c1
  FROM Aluga
  GROUP BY morada
  HAVING c1 > (
    SELECT AVG(c2)
    FROM (
      SELECT morada, COUNT(1) AS c2
      FROM Aluga
      GROUP BY morada
    ) AS Avg
  )
) AS R;
```

c) Quais os utilizadores cujos alugáveis foram fiscalizados sempre pelo mesmo fiscal?

```
SELECT nif, nome
FROM User
  NATURAL JOIN (
    SELECT nif, COUNT(1) AS c
    FROM (
      SELECT DISTINCT nif, id
      FROM Arrenda NATURAL JOIN Fiscaliza
    ) AS J
  GROUP BY nif
  HAVING c = 1
) AS R;
```

d) Qual o montante total realizado (pago) por cada espaço durante o ano de 2016, assumindo que a tarifa indicada na oferta é diária? (Considerado os casos em que o espaço foi alugado totalmente ou por postos)

```
SELECT morada, codigo_espaco,
  SUM(tarifa * DATEDIFF(data_fim, data_inicio)) AS montante_total
FROM Oferta
  NATURAL JOIN Aluga
  NATURAL JOIN Paga
  NATURAL JOIN (
    SELECT morada, codigo, codigo_espaco
    FROM Alugavel NATURAL JOIN Posto
    UNION
    SELECT morada, codigo, codigo AS codigo_espaco
    FROM Alugavel NATURAL JOIN Espaco
  ) AS A
WHERE YEAR(data) = 2016
GROUP BY morada, codigo_espaco;
```

e) Quais os espaços de trabalhos cujos postos nele contidos foram todos alugados, entendendo-se por alugado um posto de trabalho que tenha pelo menos uma oferta aceite, independentemente das suas datas?

```
SELECT DISTINCT morada, codigo_espaco AS codigo
FROM (
  SELECT morada, codigo_espaco, COUNT(1) AS c
  FROM Posto
    NATURAL JOIN Aluga
    NATURAL JOIN (
      SELECT DISTINCT numero
      FROM Estado
      WHERE estado = 'Aceite'
    ) AS R
  GROUP BY morada, codigo_espaco
) AS A
  NATURAL JOIN (
    SELECT morada, codigo_espaco COUNT(1) AS c
    FROM Posto
    GROUP BY morada, codigo_espaco
) AS T;
```

Restrições de Integridade

a) RI-1: “Não podem existir ofertas com datas sobrepostas”

```
DELIMITER $

CREATE TRIGGER TRG_Oferta_dates_overlap
BEFORE INSERT ON Oferta
FOR EACH ROW
  BEGIN

    IF New.data_inicio > New.data_fim
      THEN CALL exception_begining_date_gt_end_date ();
    END IF;

    IF EXISTS (
      SELECT 1
      FROM Oferta
      WHERE New.morada = morada
        AND New.codigo = codigo
        AND data_inicio < New.data_fim
        AND New.data_inicio < data_fim
      )
      THEN CALL exception_dates_overlap ();
      END IF;

  END $

DELIMITER ;
```

b) RI-2: “A data de pagamento de uma reserva paga tem que ser superior ao timestamp do último estado da reserva”

```
DELIMITER $

CREATE TRIGGER TRG_Paga_state_timestamp
BEFORE INSERT ON Paga
FOR EACH ROW
 BEGIN

 IF EXISTS (
 SELECT 1
 FROM Estado
 WHERE New.numero = numero
 AND (New.data < time_stamp
 OR estado = 'Paga')
 )
 THEN
 CALL exception_paid_or_date_lt_last_state_timestamp ();
 ELSE
 INSERT INTO Estado(numero, estado) VALUES(New.numero, 'Paga');
 END IF;

 END $

DELIMITER ;
```

Aplicação

Abaixo segue-se um exemplo de interação com a base de dados, incluindo ligação à base de dados e inserção de um registo que requer *input* do utilizador usando *prepared statements* e transações. Tomámos a decisão de, na remoção de registos, não tomar proveito destes mecanismos, uma vez que se tratam de transações simples, pelo que as *queries* são feitas directamente.

Exemplo – Criar Reserva (Formulário)

```
try {

  $db = new PDO("mysql:host=$dbhost;dbname=$dbname;", $dbuser, $dbpass);
  $db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

```
  if ($_SERVER['REQUEST_METHOD'] === "POST") {
    try {
      $db->beginTransaction();

      $stmt = $db->prepare("INSERT INTO Aluga VALUES (?, ?, ?, ?, ?)");
      $stmt->execute(
        array($_REQUEST['morada'], $_REQUEST['codigo'],
        $_REQUEST['data_inicio'], $_REQUEST['nif'], $_REQUEST['numero'])
      );
      $stmt = null;

      echo "<p>Inserção feita com sucesso!</p>";

      $db->commit();
    } catch (PDOException $e) {
      $db->rollBack();
      echo "<p>{$e->getMessage()}</p>";
    }
  } else {
    echo "
      <form method=\"post\">
        <p>Numero: <input type=\"text\" name=\"numero\" required /></p>
        <p>Nif: <input type=\"text\" name=\"nif\" required /></p>
        <input type=\"hidden\" name=\"morada\" value=\"".$_REQUEST['morada']."\"/>
        <input type=\"hidden\" name=\"codigo\" value=\"".$_REQUEST['codigo']."\"/>
        <input type=\"hidden\" name=\"data_inicio\" value=\"".
          $_REQUEST['data_inicio']."\"/>
        <input type=\"submit\" value=\"Inserir\"/>
      </form>";
  }

  $db = null;

} catch (PDOException $e) {
  echo "<p>PDOException: {$e->getMessage()}</p>";
}
```

De modo a atingir este formulário, existem páginas que permitem a navegação sem requerir input directo do teclado por parte do utilizador para, de algum modo, atenuar os erros que o utilizador possa causar e/ou simplificar as consultas ou ações sobre bases de dados necessárias para concluir uma ação.

São apresentadas as tabelas relevantes à acção, como, por exemplo, neste caso, a tabela com as ofertas disponíveis de modo a criar uma reserva, podendo o utilizador clicar numa ligação adjacente à oferta sobre a qual deseja criar a oferta.