# Transaccións

Cando traballamos con bases de datos pode suceder que nalgunhas ocasións queiramos que X operacións se executen como un bloque, isto é, que ou ben se executen todas correctamente, ou, se algunha falla, non queremos que se rexistre ningún cambio na base de datos. Para isto debemos usar *transaccións*.

Os motores de almacenamento de tablas en MySQL son MyISAM e InnoDB. O motor MyISAM, que exclúe a seguridade proporcionada pola integridade referencial, non permite transaccións. Por isto, para traballar con transaccións debemos traballar co motor InnoDB, que é o motor por defecto.

Por defecto, MySQL execútase en modo de execución automática (autocommit), o que significa que cada consulta individual inclúese dentro da súa propia transacción. O que conleva que tan pronto como se executa unha sentenza, se modifica a táboa de MySQL, facendo imposible a volta atrás. Para poder revertir os cambios das consultas (rollback) usando transaccións, debemos desctivar o modo de execución automática, o que iniciará a transacción.

```
$db->autocommit(false);
```

Ou co estilo procedemental:

```
mysqli_autocommit($db, false);
```

Devuelve **true** en caso de éxito o **false** en caso de error.

```
Estilo orientado a objetos
<?php
$mysqli = new mysqli("localhost", "mi_usuario",
"mi_contraseña", "world");
if (mysqli_connect_errno()) {
printf("Fallo la conexión: %s\n", mysqli_connect_error());
exit();
}
/* activar la autoconsigna */
$mysqli->autocommit(TRUE);

if ($resultado = $mysqli->query("SELECT @@autocommit")) {
$fila = $resultado->fetch_row();
printf("El estado de la autoconsigna es %s\n", $fila[0]);
$resultado->free();
}
/* Cerrar conexión */
$mysqli->close();
?>



Estilo por procedimientos
```

```php
<?php
$enlace = mysqli_connect("localhost", "mi_usuario",
"mi_contraseña", "world");

if (!$enlace) {
printf("Imposible conectar a localhost. Error: %s\n",
mysqli_connect_error());
exit();
}

/* activar la autoconsigna */
mysqli_autocommit($enlace, TRUE);

if ($resultado = mysqli_query($enlace, "SELECT
@@autocommit")) {
$fila = mysqli_fetch_row($resultado);
printf("El estado de la autoconsigna es %s\n", $fila[0]);
mysqli_free_result($resultado);
}

/* close connection */
mysqli_close($enlace);
?>
```

El resultado de los ejemplos sería: El estado de la autoconsigna es 1

O importante cando traballamos con transaccións é que o programador ten o control de cando facer efectiva a modificación ou de revertir as modificacións ocasionadas polas consultas.

Para facer efectiva as modificacións chamaremos a función `commit` coa interface orientada a obxectos e `mysqli_commit (mysqli $link)` coa interface procedimental. E para revertir a transacción actual chamaremos á función rollback coa interface procedimental e `mysqli_rollback (mysqli $link)` coa interface procedimental.

A continuación imos ver un exemplo de transacción coa interface orientada a obxectos para insertar un novo chef do que non estaba dada de alta a provincia:

```
$db = new mysqli("localhost", "alumno", "abc123.", "receitas");
if ($db->connect_error) {
    echo "Erro na conexión a base de datos";
    exit;
} else {
    $bandeira=true;
    $db->autocommit(FALSE);
    $sql1="INSERT INTO PROVINCIA (CODIGO, NOME) VALUES
('45','Toledo')";
    $sql2="INSERT INTO CHEF (CODIGO, NOME, APELIDO1, SEXO,
LOCALIDADE,
```

```
            COD_PROVINCIA) VALUES
        (12,'ANTÍA','NOGUEIRA','M','Toledo','45')";
    $result = $db->query($sql1);
    if ($db->errno) {
        $bandeira = false;
        echo '<br/>ERRO na primeira operación ('.$db->errno.')-
>'.$db->error;
    }
   $result = $db->query($sql2);
    if ($db->errno) {
        $bandeira = false;
        echo '<br/>ERRO na segunda operación ('.$db->errno.')->'.$db-
>error;
    }
     if ($bandeira == true){
         $db->commit();
         echo 'Transacción executada con éxito!.';
    } else {
         $db->rollback();
         echo '<div>Erro nalgún punto da transacción.</div>';
    }
    $db->close();
}
```

O mesmo exemplo coa interface procedemental sería:

```
$db = mysqli_connect("localhost", "alumno", "abc123.", "receitas");
if (mysqli_connect_error($db)) {
            echo "Erro na conexión a base de datos";
    exit;
} else {
    $bandeira=true;
    mysqli_autocommit($db, FALSE);
    $sql1="INSERT INTO PROVINCIA (CODIGO, NOME) VALUES
('45','Toledo')";
    $sql2="INSERT INTO CHEF (CODIGO, NOME, APELIDO1, SEXO,
LOCALIDADE,
    COD_PROVINCIA) VALUES
(12,'ANTÍA','NOGUEIRA','M','Toledo','45')";
    $result = mysqli_query($db, $sql1);
    if (mysqli_errno($db) != 0) {
        $bandeira = false;
        echo '<br/>ERRO na primeira operación ('.mysqli_errno($db).
             ') ->'. mysqli_error($db);
    }
    $result = mysqli_query($db, $sql2);
    if (mysqli_errno($db) != 0) {
        $bandeira = false;
        echo '<br/>ERRO na segunda operación ('. mysqli_errno($db).
             ') ->'. mysqli_error($db);
    }

    if ($bandeira == true){
```

```
        mysqli_commit($db);
        echo 'Transacción executada con éxito!.';
    } else {
       mysqli_rollback($db);
        echo '<div>Erro nalgún punto da transacción.</div>';
    }
    mysqli_close($db);
}
```

É importante ter en conta algunhas cousas cando se traballa con transaccións:

- O `rollback` unicamente afecta as operacións de manipulación de datos, é dicir, `DELETE`, `INSERT` e `UPDATE`. Non afecta a `CREATE`, `DROP` ou `ALTER`.
- Se temos un campo `auto_increment` e durante a transacción se fai algún `insert` con éxito, e posteriormente falla algunha operación realizada na mesma transacción, dando lugar á execución dun `rollback`, o campo `auto_increment` terá o contador como se os inserts sí houbesen tido lugar. O `rollback` non o restaura ao valor inicial. Para facelo deberiamos executar un `ALTER TABLE` para establecer o valor ao seu estado actual.
- Para deshabilitar implícitamente el modo de confirmación automática para una única serie de declaraciones:

  https://www.php.net/manual/es/mysqli.begin-transaction.php

```
Ejemplo: Estamos realizando una transferencia de fondos entre dos
cuentas bancarias y queremos asegurarnos de que ambas actualizaciones
se realicen correctamente o ninguna.
 <?php
 // Conexión a la base de datos (ajusta según tus credenciales)
 $conexion = new mysqli("localhost", "usuario", "contraseña", "basededatos");

 // Verificar conexión
 if ($conexion->connect_error) {
     die("Conexión fallida: " . $conexion->connect_error);
 }
  // Iniciar la transacción
 $conexion->begin_transaction();

 try {
     // Consultas SQL de la transacción
     $conexion->query("UPDATE cuentas SET saldo = saldo - 100 WHERE
id_cuenta = 123");
     $conexion->query("UPDATE cuentas SET saldo = saldo + 100 WHERE
id_cuenta = 456");

     // Confirmar la transacción
     $conexion->commit();

     echo "Transacción completada con éxito.";
 } catch (Exception $e) {
     // Revertir la transacción en caso de error
     $conexion->rollback();
```

```
    echo "Error durante la transacción: " . $e->getMessage();
}

// Cerrar la conexión
$conexion->close();
?>
```

En este ejemplo, la transacción se inicia con $conexion->begin_transaction(), se realizan las consultas SQL y, si todo va bien, se confirma la transacción con $conexion->commit(). Si ocurre algún error, la transacción se revierte con $conexion->rollback().

## Tarefa 6. Realización de operacións que implican transaccións

O chef con nome artístico SEOANE creou unha nova receita de nome 'ENSALADA ESPECIAL' catalogada como 'ENTRANTE' e con dificultade 'MEDIA'. Esta receta consta dos seguintes ingredientes:

- Garavanzos: 200 gramos
- Pementos: 70 gramos
- Cebola:40 gramos
- Sal: 1 pizca.

Garda esta información na base de datos tendo en conta que non queremos que a receta quede almacenada 'a medias'. Ademáis, o programa debe indicar, no caso de terse producido algún erro, a operación na que se produciu ese erro e cal é a descrición deste.