Stored Procedures

Class	BDatos 6102
Created	@Apr 27, 2020 2:57 PM
KPI	
Reviewed	☐
Type	

Primera parte

Materiales

Agrega material dentro de tabla Materiales

```
IF EXISTS (SELECT name FROM sysobjects
                      WHERE name = 'creaMaterial' AND type = 'P')
            DROP PROCEDURE creaMaterial
            GO

            CREATE PROCEDURE creaMaterial
                @uclave NUMERIC(5,0),
                @udescripcion VARCHAR(50),
                @ucosto NUMERIC(8,2),
                @uimpuesto NUMERIC(6,2)
            AS
                INSERT INTO Materiales VALUES(@uclave, @udescripcion, @ucosto, @uimpuesto)
            GO
```

Traduccin a MySQL Workbench

```
DELIMITER //
DROP PROCEDURE IF EXISTS creaMaterial;

  CREATE PROCEDURE creaMaterial
    (
    IN uclave NUMERIC(5,0),
    IN udescripcion VARCHAR(50),
    IN ucosto NUMERIC(8,2),
    IN uimpuesto NUMERIC(6,2)
  ) BEGIN
    INSERT INTO Materiales VALUES(uclave, udescripcion, ucosto, uimpuesto);
  END
//
DELIMITER ;
```

- ¿Qué hace el primer bloque del código (bloque del IF)?

Verifica que no exista ya un stored procedure con ese mismo nombre. Si es el caso, lo elimina

- ¿Para qué sirve la instrucción GO?

Es un delimitador de un conjunto de operaciones. Es decir, una vez que se hayan ejecutado correctamente todas las operaciones anteriores correctamente, manda la instruccion de ejecutarlo de manera permanente.

Analogamente, en otros DBMS tenemos el uso de un simple punto y coma.

- ¿Explica que recibe como parámetro este Procedimiento y qué tabla modifica?

Recibe los parametros uclave, udescripcion, ucosto, uimpuesto, insertandolos en la tabla de Materiales para agregar un nuevo material dentro de la tabla.

Modifica material

```
DELIMITER //
DROP PROCEDURE IF EXISTS modificaMaterial;

  CREATE PROCEDURE modificaMaterial
    (
    IN uclave NUMERIC(5,0),
    IN udescripcion VARCHAR(50),
    IN ucosto NUMERIC(8,2),
    IN uimpuesto NUMERIC(6,2)
  ) BEGIN
    UPDATE Materiales set descripcion = udescripcion, costo = ucosto, impuesto = uimpuesto
where clave = uclave
  END
//
DELIMITER ;
```

Elimina material

```
DELIMITER //
DROP PROCEDURE IF EXISTS eliminaaMaterial;

  CREATE PROCEDURE eliminaaMaterial
    (
    IN uclave NUMERIC(5,0),
  ) BEGIN
    DELETE FROM Materiales where clave = uclave
  END
//
DELIMITER ;
```

Proyecto

Agrega Proyecto

```
DELIMITER //
DROP PROCEDURE IF EXISTS agregaProyecto;

  CREATE PROCEDURE agregaProyecto
    (
    IN unumero NUMERIC(5,0),
    IN udenominacion VARCHAR(50)
  ) BEGIN
    INSERT INTO Proyectos VALUES(unumero, udenominacion);
  END
//
DELIMITER ;
```

Modifica proyecto

```
DELIMITER //
DROP PROCEDURE IF EXISTS modificaProyecto;

  CREATE PROCEDURE modificaProyecto
    (
    IN unumero NUMERIC(5,0),
    IN udenominacion VARCHAR(50)
  ) BEGIN
    UPDATE Proyectos set denominacion = udenominacion
where numero = unumero;
  END
//
DELIMITER ;
```

Elimina proyecto

```
DELIMITER //
DROP PROCEDURE IF EXISTS eliminaProyecto;

  CREATE PROCEDURE eliminaProyecto
    (
    IN unumero NUMERIC(5,0)
  ) BEGIN
    DELETE FROM Proyectos where numero = unumero;
  END
//
DELIMITER ;
```

Proveedores

Agrega Proveedor

```
DELIMITER //
DROP PROCEDURE IF EXISTS agregaProveedor;

  CREATE PROCEDURE agregaProveedor
    (
    IN urfc VARCHAR(9),
    IN urazonsocial VARCHAR(50)
  ) BEGIN
    INSERT INTO Proveedor VALUES(urfc, urazonsocial);
  END
//
DELIMITER ;
```

Modifica Proveedor

```
DELIMITER //
DROP PROCEDURE IF EXISTS modificaProveedor;

  CREATE PROCEDURE modificaProveedor
    (
    IN urfc VARCHAR(9),
    IN urazonsocial VARCHAR(50)
  ) BEGIN
    UPDATE Proveedor set RazonSocial = urazonsocial
where RFC = urfc;
  END
//
DELIMITER ;
```

Elimina Proveedor

```
DELIMITER //
DROP PROCEDURE IF EXISTS eliminaProveedor;

  CREATE PROCEDURE eliminaProveedor
    (
    IN urfc VARCHAR(9)
  ) BEGIN
    DELETE FROM Proyectos where RFC = urfc;
  END
//
DELIMITER ;
```

Entregan

Agrega entrega

```
DELIMITER //
DROP PROCEDURE IF EXISTS agregaEntrega;
```

```
  CREATE PROCEDURE agregaEntrega
    (
    IN uclave NUMERIC(5,0),
    IN urfc VARCHAR(13),
    IN unumero NUMERIC(5,0),
    IN ufecha datetime,
    IN ucantidad NUMERIC(8,2)
  ) BEGIN
    INSERT INTO Entregan VALUES(uclave, urfc, unumero, ufecha, ucantidad);
  END
//
DELIMITER ;
```

Modifica entrega

```
DELIMITER //
DROP PROCEDURE IF EXISTS modificaEntrega;

  CREATE PROCEDURE modificaEntrega
    (
    IN uclave NUMERIC(5,0),
    IN urfc VARCHAR(13),
    IN unumero NUMERIC(5,0),
    IN ufecha datetime,
    IN ucantidad NUMERIC(8,2)
  ) BEGIN
    UPDATE Entregan set RFC = urfc, Numero = unumero, Fecha = ufecha
where clave = uclave;
  END
//
DELIMITER ;
```

Elimina entrega

```
DELIMITER //
DROP PROCEDURE IF EXISTS eliminaEntrega;

  CREATE PROCEDURE eliminaEntrega
    (
    IN uclave NUMERIC(5,0),
    IN urfc VARCHAR(13),
    IN unumero NUMERIC(5,0),
    IN ucantidad NUMERIC(8,2)
  ) BEGIN
    DELETE FROM Entregan where clave = uclave and RFC = urfc and Numero = unumero and Cantidad = ucantidad;
  END
//
DELIMITER ;
```

Segunda Parte

Define el siguiente en tu base de datos

```
IF EXISTS (SELECT name FROM sysobjects
                WHERE name = 'queryMaterial' AND type = 'P')
          DROP PROCEDURE queryMaterial
      GO

      CREATE PROCEDURE queryMaterial
          @udescripcion VARCHAR(50),
          @ucosto NUMERIC(8,2)

      AS
          SELECT * FROM Materiales WHERE descripcion
          LIKE '%'+@udescripcion+'%' AND costo > @ucosto
      GO
```

Traduccion a MySQL Workbench

```
DELIMITER //
DROP PROCEDURE IF EXISTS queryMaterial;
  CREATE PROCEDURE queryMaterial
    (
      IN udescripcion VARCHAR(50),
      IN ucosto NUMERIC(8,2)
    ) BEGIN
        SELECT * FROM Materiales WHERE descripcion
          LIKE '%'+udescripcion+'%' AND costo > ucosto
      END

//
DELIMITER ;
```

Qué recibe como parámetro este procedimiento y qué hace?

Este stored procedure recibe la descripcion de un Material y el costo del mismo, para despues seleccionar todos las filas dentro de la tabla de Materiales que cumplan con lo parametros ingresados dentro del stored procedure.

Ejecutar Stored Procedures desde la aplicación cliente (al menos 3)

Eliminar proyecto desde cliente-servidor

En este codigo primero se hace una funcion para crear el SP eliminar entrega, despues otra funcion que ejecutara dicho stored procedure

```
function crear_sp_eliminar_entrega($Clave, $RFC, $Numero, $Cantidad) {
    $conexion_bd = connectBD();

    if (!$mysqli->query("DROP PROCEDURE IF EXISTS eliminaProveedor") ||
        !$mysqli->query("CREATE PROCEDURE eliminaProveedor(IN urfc VARCHAR(9)) IN urfc VARCHAR(9); END;")) {
        echo "Falló la creación del procedimiento almacenado: (" . $mysqli->errno . ") " . $mysqli->error;
    }
```

```
    if (!$mysqli->query("CALL eliminaProveedor(1)")) {
        echo "Falló CALL: (" . $mysqli->errno . ") " . $mysqli->error;
    }

    disconnectBD($conexion_bd);
    return 0;
  }



function sp_eliminar_entrega ($RFC) {
    $conexion_bd = connectBD();

    $dml = 'CALL eliminaProveedor(?);';

    if ( !($statement = $conexion_bd->prepare($dml)) ) {
        die("Error: (" . $conexion_bd->errno . ") " . $conexion_bd->error);
        return 1;
    }

    //Unir los parametros de la funcion con los parametros de la consulta
    if (!$statement->bind_param("s", $RFC)) {
        die("Error en vinculación: (" . $statement->errno . ") " . $statement->error);
        return 1;
    }

    //Executar la consulta
    if (!$statement->execute()) {
      die("Error en ejecución: (" . $statement->errno . ") " . $statement->error);
      return 1;
    }


    disconnectBD($conexion_bd);
    return 0;

}
```

```
function call_creaMaterial($idPersona, $idEstado){
      $conexion_bd = conectar_bd();

      //preparar consulta
      $dml_insertar = 'CALL creaMaterial(?,?)';
      if(!($statement = $conexion_bd->prepare($dml_insertar)))
      {
        die("Error: (".$conexion_bd->errno.") ".$conexion_bd->error);
        return 0;
      }

      //unir parámetros de la función con la consulta
      //el primer arg es el formato de cada parámetro
      if(!$statement->bind_param("ii", $idPersona, $idEstado))
      {
        die("Error en vinculación: (".$statement->errno.") ".$statement->error);
        return 0;
      }

      //Ejecutar inserción
      if(!$statement->execute())
      {
        die("Error en ejecución: (".$statement->errno.") ".$statement->error);
```

```
        return 0;
      }

      desconectar_bd($conexion_bd);
      return 1;
    }
```

PREGUNTAS

- ¿Qué ventajas tienen el utilizar Stored Procedures en una aplicación cliente-servidor?

Obtienes ventajas en tiempo y errores. Al crear SP, estos te permiten hacer funciones que reducen el tiempo en el que tienes que realizar una consulta o una ejecucion dentro de la base de datos. Seria lo equivalente a realizar una funcion dentro de cualquier lenguaje de programacion, pero para BD.

En cuanto a errores, debido a que el sp es, analogamente, una funcion, puedes evitar un codigo que reescribirias muchas veces en dado caso de que no lo tengas. Al momento de escribir la funcion una y otra vez, siempre esta la posibilidad de equivocarse, por lo que asi solo tienes que pasarle un parametro donde el error a cometer seria minimo. Y en dado caso, podrias anticiparlo dentro de la funcion

- ¿Qué ventajas tiene utilizar SP en un proyecto?

Debido a que reduce tiempo y errores, y los proyectos son en gran escala, van a facilitar la realizacion del mismo, reduciendo tiempos y pudiendo lograr metas mas grandes.