GBD06- Solución a la TAREA

CONSTRUCCIÓN DE GUIONES.

Para realizar las actividades de esta tarea necesitas tener instalada la base de datos **TalleresFaber**. Para ello puedes descargarte del apartado Recursos de esta unidad el recurso GBD06_CONT_R29_talleresfaber.zip, que contiene los script SQL para crear la base de datos talleresfaber e insrtar los datos en ella.

Creación: bd_talleresfaber.sql
Insertar: insert_talleresfaber.sql

EJERCICIO 1

Crear un **procedimiento** que muestre los vehículos (marca, modelo y color) que no estén reparados y los datos de los clientes y vehículos que han entrado a reparar hoy. (En nuestro caso ninguno).

EJERCICIO 2

- Realizar un **procedimiento** que recibe la matrícula de un vehículo y escriba las características del automóvil y el número de reparaciones que ha sufrido ese automóvil, los empleados que han realizado esas reparaciones y los datos de los vehículos de la misma marca.
- Hacer una llamada al procedimiento creado.

EJERCICIO 3

Modifica el procedimiento anterior añadiendo un **HANDLER** que controle que si esa matrícula no está en la base de datos, el resto de instrucciones no se ejecuten.

EJERCICIO 4

Crear una **función** que actualice el estado de las reparaciones que estén finalizadas en una fecha que se indique y que devuelva cuantas reparaciones han finalizado en esa fecha.

EJERCICIO 5

Crear un **procedimiento** para **dar de alta una nueva reparación** para un vehículo y un cliente que no tenemos registrados. Llama al procedimiento ReparacionClienteNuevo.
Incluye un **HANDLER** que controle que si insertamos un cliente y/o un vehículo que ya existen, el resto de sentencias continúen ejecutándose, y se añade como mínimo la nueva reparación.
Para probar el procedimiento toma como referencia los datos siguientes: (tomados de un ejercicio de la unidad anterior).
Un cliente nuevo nos ha traído su vehículo al taller el día 03/03/2011. En recepción se registran los siguientes datos:

1. *Del **cliente**.- Código: 00011, Nombre y apellidos: Tomás Gómez Calle, Teléfono: 22334455.*
2. *Del **vehículo**.- Matrícula: 3131 FGH, Modelo: Renault Scénic, matriculado el 17/03/2009, 105.000 km;*
3. *De la **reparación**.- Sustitución de las lámparas delanteras.*

EJERCICIO 6

Creación de **funciones**:

1. Diseña una función que calcule el importe de los recambios sustituidos en una reparación.
2. Crear una función que devuelva el importe de las actuaciones que se llevan a cabo en una reparación (para calcular el importe multiplica las horas por el importe de cada actuación).

En ambas funciones Pasar como variable el Id de la reparación.

3. Hacer una consulta que calcule el importe total (mano de obra y recambios) de las reparaciones que se le hayan realizado al vehículo de matrícula '1313 DEF'.

EJERCICIO 7

Crear una **función** que reciba como parámetro de entrada el número correspondiente a un mes y devuelva el importe total facturado ese mes. Utiliza para ello las dos funciones obtenidas en la práctica anterior.
NOTAS:

- Por ejemplo para Enero, número del mes 1.
- Utilizar un cursor para recorrer cada fila de la consulta de los IdReparacion que se obtengan en ese mes.
- Controla mediante un HANDLER que la consulta haya devuelto alguna fila.

EJERCICIO 8

Crea un **trigger** que, antes de insertar una fila en la tabla **Incluyen**, compruebe si existen unidades en Stock en la tabla RECAMBIOS llevando a cabo las siguientes acciones:

- Si hay suficientes unidades actualiza el Stock restando las unidades que se van a insertar.
- Si no hay suficientes unidades en Stock cancela la inserción de las unidades.

EJERCICIO 9

1. Crea una tabla denominada **PedidoRecambios** que contenga 3 columnas:
   - IdRecambio,
   - Descripcion
   - Stock

   Con los mismos tipos de datos que tienen esas columnas en la tabla RECAMBIOS.

2. Crear un **trigger** asociado a la tabla **RECAMBIOS** que después de actualizar el Stock de un recambio, si el número de unidades en Stock del recambio modificado es inferior a 4 unidades, inserte una fila en la tabla PedidoRecambios con los datos resultantes.

EJERCICIO 10

Utilizando funciones de librerías disponibles en MySQL obtener:

1. Un listado con dos columnas: en la primera, en mayúsculas apellidos y nombre de todos los clientes (entre los apellidos y el nombre incluir una coma como separador) y en la segunda, la ciudad en la que cada cliente tiene su domicilio (únicamente la ciudad, no la dirección).
2. Un listado con 2 columnas: en la primera la fecha de alta de los empleados con el formato dd/mm/aaaa y en la segunda aparecerá 'Contrato temporal ' para aquellos empleados que lleven contratados en el taller menos de 2 años, y 'Contrato fijo' para el resto.

Nota para el tutor:

La solución propuesta a continuación es orientativa. Algunas consultas pueden resolverse de distintas formas.

SOLUCIÓN EJERCICIO 1:

```
DELIMITER |
DROP PROCEDURE IF EXISTS listadoHoy |
CREATE PROCEDURE listadoHoy()
BEGIN
        SELECT VEHICULOS.Matricula, Marca, Modelo, Color FROM (VEHICULOS INNER JOIN
        REPARACIONES ON VEHICULOS.Matricula=REPARACIONES.Matricula)
        INNER JOIN CLIENTES ON CLIENTES.CodCliente=VEHICULOS.CodCliente WHERE NOT Reparado;
        SELECT Nombre, Apellidos, REPARACIONES.Matricula, Marca, Modelo FROM
        (VEHICULOS INNER JOIN       REPARACIONES ON
        VEHICULOS.Matricula=REPARACIONES.Matricula)
        INNER JOIN CLIENTES ON CLIENTES.CodCliente=VEHICULOS.CodCliente WHERE CURDATE() =
        FechaEntrada;
END |
DELIMITER ;
```

SOLUCIÓN EJERCICIO 2:

```
DELIMITER |
DROP PROCEDURE IF EXISTS DatosVehiculo |
CREATE PROCEDURE DatosVehiculo (IN Matri CHAR(8), OUT a INT)
BEGIN
        DECLARE m VARCHAR(15);
        SELECT Marca, Modelo, Color FROM VEHICULOS WHERE Matricula=Matri;
        SELECT COUNT(*) INTO a FROM REPARACIONES WHERE Matricula=Matri;
        SELECT DISTINCT Nombre, Apellidos FROM (EMPLEADOS INNER JOIN Intervienen ON
        EMPLEADOS.CodEmpleado=Intervienen.CodEmpleado) INNER JOIN REPARACIONES ON
        Intervienen.IdReparacion=REPARACIONES.IdReparacion WHERE Matricula=Matri;
        SET m=(SELECT Marca FROM VEHICULOS WHERE Matricula=Matri);
        SELECT Matricula, Modelo, Color FROM VEHICULOS WHERE Marca=m;
END |
DELIMITER ;
```

Para llamar al procedimiento:

```
SET @Num=0;
CALL DatosVehiculo ('1313 DEF', @Num);
SELECT @NUM;
```

SOLUCIÓN EJERCICIO 3:

```
DELIMITER |
DROP PROCEDURE IF EXISTS DatosVehiculo |
CREATE PROCEDURE DatosVehiculo (IN Matri CHAR(8), OUT a INT)
BEGIN
        DECLARE m VARCHAR(15);
        DECLARE ma varchar(8);
        DECLARE EXIT HANDLER FOR SQLSTATE '02000' SELECT 'La matricula no existe';
        SELECT Matricula INTO ma FROM VEHICULOS WHERE Matricula=Matri;
    BEGIN
        SELECT Marca, Modelo, Color FROM VEHICULOS WHERE Matricula=Matri;
        SELECT COUNT(*) INTO a FROM REPARACIONES WHERE Matricula=Matri;
        SELECT DISTINCT Nombre, Apellidos FROM (EMPLEADOS INNER JOIN Intervienen ON
        EMPLEADOS.CodEmpleado=Intervienen.CodEmpleado) INNER JOIN REPARACIONES ON
        Intervienen.IdReparacion=REPARACIONES.IdReparacion WHERE Matricula=Matri;
        SET m=(SELECT Marca FROM VEHICULOS WHERE Matricula=Matri);
        SELECT Matricula, Modelo, Color FROM VEHICULOS WHERE Marca=m;
    END ;
END |
DELIMITER ;
```

SOLUCIÓN EJERCICIO 4:

```
DELIMITER |
DROP FUNCTION IF EXISTS DatosFecha |
CREATE FUNCTION DatosFecha (Fecha DATE)
RETURNS INT
BEGIN
        DECLARE a INT DEFAULT 0;
        DECLARE HANDLER
        SELECT COUNT(*) INTO a FROM REPARACIONES WHERE FechaSalida=Fecha;
        UPDATE REPARACIONES SET Reparado=1 WHERE FechaSalida=Fecha;
        RETURN a;
END |
DELIMITER ;
```

Para llamar a la función:

```
SELECT DatosFecha(CURDATE());
SET @num=DatosFecha('2011-01-03');
```

SOLUCIÓN EJERCICIO 5:

```
DELIMITER |
DROP PROCEDURE IF EXISTS RegistrarAlta|
CREATE PROCEDURE RegistrarAlta
(IN Cod VARCHAR(5),
IN Dn VARCHAR(10),
IN Ape VARCHAR(30),
IN Nom VARCHAR(25),
IN Dir VARCHAR(50),
IN Telef VARCHAR(9),
IN Matri VARCHAR(8),
IN Mar VARCHAR(25),
IN Model VARCHAR(50),
IN Col VARCHAR(5),
IN Fmat DATE,
IN Kms INT,
IN FEn DATE,
IN Descri VARCHAR(100))

BEGIN
DECLARE CONTINUE HANDLER FOR SQLSTATE '23000' SELECT 'El Cliente y/o Vehículo ya
existen.Reparación registrada' AS 'Clave Duplicada';
INSERT INTO CLIENTES  (CodCliente, DNI, Apellidos, Nombre, Direccion, Telefono)
VALUES (Cod, Dn, Ape, Nom, Dir, Telef);

INSERT INTO VEHICULOS (Matricula, Marca, Modelo, Color, FechaMatriculacion, CodCliente)
VALUES (Matri, Mar, Model, Col, Fmat, Cod);

INSERT INTO REPARACIONES
VALUES (NULL, Matri, FEn, Kms, Descri, NULL, 0, NULL);
END|
DELIMITER ;
```

Para llamar al procedimiento:

```
CALL RegistrarAlta('00011', 0, 'Gómez Calle', 'Tomás', Null, '22334455',
'3131 FGH', 'Renault', 'Scénic', NULL, '2009-03-17', 105000, '2011-03-03','Sustitución de lámparas delanteras');
```

SOLUCIÓN EJERCICIO 6:

```
DELIMITER |
DROP FUNCTION  IF EXISTS ImporteRecambios |
CREATE FUNCTION ImporteRecambios (IdRep INT)
RETURNS FLOAT
BEGIN
   DECLARE Resultado FLOAT Default 0;
   SELECT  SUM(Unidades*PrecioReferencia) INTO Resultado
   FROM  Incluyen INNER JOIN RECAMBIOS REC ON REC.IdRecambio = Incluyen.IdRecambio
   WHERE  Incluyen.IdReparacion = IdRep;
RETURN Resultado;
END |
DELIMITER ;


DELIMITER |
DROP FUNCTION  IF EXISTS ImporteActuaciones |
CREATE FUNCTION ImporteActuaciones (IdRep INT)
RETURNS FLOAT
BEGIN
   DECLARE Resultado FLOAT Default 0;
   SELECT SUM(Horas*Importe) INTO Resultado
   FROM Realizan Rz INNER JOIN ACTUACIONES AC
   ON Rz.Referencia = AC.Referencia
   AND IdReparacion = IdRep;
   RETURN Resultado;
END |
DELIMITER ;
```

Para probar la función:

```
SELECT IdReparacion, ImporteRecambios(IdReparacion)+ImporteActuaciones(IdReparacion)
        FROM REPARACIONES WHERE Matricula='1313 DEF;
```

SOLUCIÓN EJERCICIO 7:

```
DELIMITER |
DROP FUNCTION  IF EXISTS FacturacionMes |
CREATE FUNCTION FacturacionMes (Mes INT)
RETURNS FLOAT
BEGIN
   DECLARE Existe INT DEFAULT 1;
   DECLARE Tot FLOAT Default 0;
   DECLARE k INT;
   DECLARE cur_1 CURSOR  FOR SELECT IdReparacion FROM REPARACIONES WHERE
  Mes=MONTH(FechaSalida);
   DECLARE CONTINUE HANDLER FOR SQLSTATE '02000' SET Existe = 0;
   SET Tot=0;
   OPEN cur_1;
   FETCH cur_1 INTO k;
   WHILE Existe=1 DO
      SET Tot=Tot+ImporteRecambios(Mes)+ImporteActuaciones(Mes);
      FETCH Cur_1 INTO k;
   END WHILE;
CLOSE Cur_1;
RETURN Tot;
END |
DELIMITER ;
```

SOLUCIÓN EJERCICIO 8:

```
DELIMITER |
DROP TRIGGER IF EXISTS ComprobarStock |
CREATE TRIGGER ComprobarStock BEFORE INSERT ON Incluyen FOR EACH ROW
BEGIN
DECLARE SK INT;
SELECT Stock INTO SK FROM RECAMBIOS WHERE IdRecambio=New.IdRecambio;
IF SK>=New.Unidades
   THEN UPDATE RECAMBIOS SET Stock=Stock-New.Unidades WHERE IdRecambio=New.IdRecambio;
      ELSE
   INSERT INTO RECAMBIOS VALUES ('Provocar un error');
END IF;
END |
DELIMITER ;
```

SOLUCIÓN EJERCICIO 9:

```
CREATE TABLE `PedidoRecambios` (
  `IdRecambio` varchar(10) NOT NULL,
  `Descripcion` varchar(100) default NULL,
  `Stock` smallint(6) default NULL,
  PRIMARY KEY  (`IdRecambio`)
) ENGINE=InnoDB;

DELIMITER |
DROP TRIGGER IF EXISTS PedidoRecambios |
CREATE TRIGGER PedidoRecambios AFTER UPDATE ON RECAMBIOS FOR EACH ROW
BEGIN
DECLARE SK INT;
SELECT New.Stock INTO SK FROM RECAMBIOS WHERE IdRecambio=New.IdRecambio;
IF SK<4
   THEN INSERT INTO PedidoRecambios VALUES (New.IdRecambio, New.Descripcion, New.Stock);
END IF;
END |
DELIMITER ;
```

SOLUCIÓN EJERCICIO 10:

Ejercicio 1

```
SELECT UPPER(CONCAT_WS(', ',Apellidos, Nombre)) AS 'APELLIDOS Y NOMBRE',
RIGHT(Direccion,INSTR(REVERSE(Direccion),' ')) AS CIUDAD FROM CLIENTES;
```

Otra solución:

```
SELECT UPPER(CONCAT_WS(', ',Apellidos, Nombre)) AS 'APELLIDOS Y NOMBRE',
SUBSTRING_INDEX(Direccion, ',' , -1) AS CIUDAD FROM CLIENTES;
```

Ejercicio 2

```
SELECT DATE_FORMAT(FechaAlta, '%d/%m/%Y') AS 'FECHA DE ALTA',
if((YEAR(CURDATE())-YEAR(FechaAlta))>=2, 'Contrato fijo', 'Contrato temporal') AS 'TIPO DE CONTRATO'
FROM EMPLEADOS;
```