UNIVERSIDAD DE MÁLAGA
Departamento de Lenguajes y Ciencias de la Computación
UNIVERSIDAD DE MÁLAGA

Gestión de la Información
Práctica 2

SQL - Server

Vamos a crear y a trabajar con una base de datos de reservas de habitaciones de un hotel. El objetivo de esta práctica es profundizar en las sentencias T-SQL y crear funciones definidas por el usuario y procedimientos almacenados. Toda la práctica se trabajará desde una consola de comandos `sqlcmd`.

Lo primero que vamos a hacer es crear una base de datos llamada "Hotel" y crear una tabla de usuarios del sistema. Dicha tabla tendrá los campos `Nombre varchar(20) NOT NULL, pwd varchar(8) NOT NULL, rol int` y la clave primaria será el nombre de usuario.

```
Create database hotel;
use hotel;

CREATE TABLE  usuario  (
  Nombre  varchar(20) NOT NULL,
  pwd     varchar(8) NOT NULL,
  rol     int,
  PRIMARY KEY  (Nombre)
) ;
```

Posteriormente vamos a insertar a los usuarios admin, user y invitado:
INSERT INTO usuario VALUES ('admin','admin',2);
INSERT INTO usuario VALUES ('user','user',1);
INSERT INTO usuario VALUES ('invitado','',0);

Haremos un select de toda la tabla para comprobar que se han insertado correctamente y posteriormente vamos a insertar otra vez al usuario 'invitado' para asegurarnos que la restricción de la clave funciona bien.

INSERT INTO usuario VALUES ('invitado','hola',0);

Ahora debemos crear la tabla tipo_servicio con un único atributo (NombreSERVICIO varchar(10) NOT NULL) que será la clave de la tabla, e insertar en dicha tabla los valores 'COMEDOR' y 'LAVANDERIA'.

A continuación debemos crear la tabla paises con un único atributo (pais varchar(20) NOT NULL) que será la clave de la tabla, e insertar en dicha tabla los valores 'ALEMANIA', 'ESPAÑA', 'FRANCIA' y 'PORTUGAL'.

A continuación debemos crear la tabla clientes con los siguientes atributos (Identificacion varchar(12) NOT NULL, Pais varchar (20) NOT NULL, Nombre varchar (12) NOT NULL, Apellido1 varchar (12) NOT NULL, Apellido2 varchar (12) , Direccion varchar (30) NOT NULL, Telefono varchar (12) NOT NULL, Observaciones varchar (50)). La clave primaria de dicha tabla será el campo Identificacion y el campo país será clave externa de la tabla países (FOREIGN KEY (Pais) references paises (pais)).

Insertar en dicha tabla los valores de clientes
('12345', 'ESPAÑA', 'Felipe', 'Iglesias', 'López', 'Avda Los Castros, 44', '942344444', 'Buen cliente');
('44444', 'ESPAÑA', 'Luis', 'García', 'García', 'Calle Mayor, 67 ', '942456444', null);
('456789', 'FRANCIA', 'Ludovic', 'Giuly', 'Bourquin', '18 avenue Alsacen Cour', '37890194', null);

A continuación debemos crear la tabla tipo_habitacion con los siguientes atributos (Categoria int NOT NULL , Camas int NOT NULL, Exterior char(2) NOT NULL check (Exterior in ('SI', 'NO')), Salon varchar(2) NOT NULL check (Salon in ('SI', 'NO')), Terraza varchar (2) NOT NULL check (Terraza in ('SI', 'NO'))) y cuya clave primaria será el campo Categoria.

Insertar en dicha tabla los valores
(1, 1, 'SI', 'NO', 'NO');
(2, 2, 'SI', 'NO', 'NO');
(3, 3, 'SI', 'NO', 'NO');
(4, 1, 'SI', 'SI', 'NO');

A continuación debemos crear la tabla habitaciones con los siguientes atributos (NumHABITACION int NOT NULL, TIPO_HABITACION int NOT NULL), donde la clave primaria será el número de habitación y el tipo de habitación será clave externa que referencia al campo categoría de la tabla tipo_habitacion.

Insertar las siguientes habitaciones: (101, 1), (102, 1), (103, 1), (104, 2), (105, 2), (106, 3), (107, 4)

A continuación debemos crear la tabla servicios con los siguientes atributos (idSERVICIOS int NOT NULL , NombreSERVICIO varchar(10) NOT NULL, Descripcion varchar (30) NOT NULL, Precio money NOT NULL, IVA numeric (5,2) NOT NULL, Fecha date NOT NULL). La clave principal de la tabla será idSERVICIOS y (nombreServicio será clave externa de la tabla tipo_servicio.

Estableceremos el formato de fecha como año-mes-día (set dateformat ymd;) e insertaremos los siguientes servicios:
(1, 'COMEDOR', '1 menu del Dia', 10, 7, '2009-01-01')
(2, 'LAVANDERIA', 'Lavado de Camisa', 2, 7, '2009-01-01')
(3, 'LAVANDERIA', 'Lavado de pantalon', 1, 7, '2009-01-01')

A continuación debemos crear la tabla temporada con los siguientes atributos (TEMPORADA int NOT NULL , FechaINICIO date NOT NULL, FechaFINAL date NOT NULL, Tipo varchar(1) not null check (tipo in ('B','M','A'))) y cuya clave primaria es el campo TEMPORADA.

Estableceremos el formato de fecha como año-mes-día (set dateformat ymd;) e insertaremos los siguientes valores:
(1, '2009-01-01', '2009-03-31', 'B');
(2, '2009-04-01', '2009-05-31', 'M');
(3, '2009-06-01', '2009-08-31', 'A');
(4, '2009-09-01', '2009-10-31', 'M');
(5, '2009-11-01', '2009-12-31', 'B');

A continuación debemos crear la tabla precio_habitacion con los siguientes atributos (idPrecio int NOT NULL , Precio money NOT NULL, TEMPORADA int NOT NULL, TIPO_HABITACION int NOT NULL) donde la clave principal será idPrecio y temporada será clave externa de la tabla temporada y tipo_habitacion será clave externa de la tabla tipo_habitacion.

Insertaremos los siguientes valores: (1, 30, 1, 1), (2, 35, 2, 1), (3, 40, 3, 1), (4, 35, 4, 1), (5, 30, 5, 1), (6, 35, 1, 2), (7, 40, 2, 2) (8, 45, 3, 2), (9, 40, 4, 2), (10, 35, 5, 2), (11, 40, 1, 3), (12, 45, 2, 3), (13, 50, 3, 3), (14, 45, 4, 3), (15, 40, 5, 3), (16, 50, 1, 4), (17, 55, 2, 4), (18, 60, 3, 4), (19, 55, 4, 4), (20, 50, 5, 4);

A continuación debemos crear la tabla reserva_habitac con los siguientes atributos (idRESERVA numeric identity(1,1) NOT NULL , FechaENTRADA date NOT NULL, FechaSALIDA date NOT NULL, IVA numeric(5,2) NOT NULL, NumHABITACION int NOT NULL, CLIENTE varchar(12) NOT NULL) donde la clave primaria es idRESERVA, el campo cliente es clave externa de la tabla CLIENTES y numHabitacion es clave externa de la tabla HABITACIONES.

Crear la Función de usuario getRol que toma como parámetros el nombre de usuario y la contraseña y devuelve el entero que representa a su rol. En caso de que el usuario no esté registrado deberá devolver 0 (Invitado).

Crear el Procedimiento almacenado InsertaUsuario que recibe como parámetros el nombre y la contraseña del usuario que quiere realizar la inserción, y el nombre, contraseña y rol del nuevo usuario. Si el usuario que quiere hacer la inserción es administrador (rol 2) se realizará la inserción y en otro caso se ignorará.

Ejecutar las siguientes instrucciones y ver su resultado:

```
exec Insertausuario 'admin', 'admin', 'user2', 'user2', 1;
select * from usuario;

exec Insertausuario 'admin', 'admin', 'user2', '2', 1;
select * from usuario;

exec Insertausuario 'admin', 'hola', 'user3', 'user3', 1;
select * from usuario;

exec Insertausuario 'user', 'user', 'user3', 'user3', 1;
select * from usuario;
```

TRANSACCIONES

Una transacción es un conjunto de operaciones que van a ser tratadas como una única unidad. Estas transacciones deben cumplir 4 propiedades fundamentales comúnmente conocidas como ACID (atomicidad, coherencia, asilamiento y durabilidad).
La transacción más simple en SQL Server es una única sentencia SQL. Por ejemplo una sentencia como esta: UPDATE Products SET UnitPrice=20 WHERE ProductName =’Chai’ Es una transacción.
Esta es una transacción ‘autocommit’, una transacción autocompletada.

Cuando enviamos esta sentencia al SQL Server se escribe en el fichero de transacciones lo que va a ocurrir y a continuación realiza los cambios necesarios en la base de datos. Si hay algún tipo de problema al hacer esta operación el SQL Server puede leer en el fichero de transacciones lo que se estaba haciendo y si es necesario puede devolver la base de datos al estado en el que se encontraba antes de recibir la sentencia.
Por supuesto este tipo de transacciones no requieren de nuestra intervención puesto que el sistema se encarga de todo.

Sin embargo si hay que realizar varias operaciones y queremos que sean tratadas como una unidad tenemos que crear esas transacciones de manera explícita.

Sentencias para una transacción

Como decíamos una transacción es un conjunto de operaciones tratadas como una sola. Este conjunto de operaciones debe marcarse como transacción para que todas las operaciones que la conforman tengan éxito o todas fracasen. La sentencia que se utiliza para indicar el comienzo de una transacción es ‘BEGIN TRAN’.
Si alguna de las operaciones de una transacción falla hay que deshacer la transacción en su totalidad para volver al estado inicial en el que estaba la base de datos antes de empezar. Esto se consigue con la sentencia ‘ROLLBACK TRAN’.
Si todas las operaciones de una transacción se completan con éxito hay que marcar el fin de una transacción para que la base de datos vuelva a estar en un estado consistente con la sentencia ‘COMMIT TRAN’.

Un ejemplo

```
DECLARE @Error int
--Declaramos una variable que utilizaremos para almacenar un posible código de error
BEGIN TRAN
--Iniciamos la transacción
UPDATE Products SET UnitPrice=20 WHERE ProductName =’Chai’
--Ejecutamos la primera sentencia
IF (@Error<>0)
BEGIN
PRINT ‘Ha ecorrido un error. Abortamos la transacción’
--Se lo comunicamos al usuario y deshacemos la transacción
--todo volverá a estar como si nada hubiera ocurrido
ROLLBACK TRAN
RETURN
END
--Si la primera sentencia se ejecuta con éxito, pasamos a la segunda
UPDATE Products SET UnitPrice=20 WHERE ProductName=’Chang’
SET @Error=@@ERROR
--Y si hay un error hacemos como antes
BEGIN
PRINT ‘Ha ecorrido un error. Abortamos la transacción’
--Se lo comunicamos al usuario y deshacemos la transacción
--todo volverá a estar como si nada hubiera ocurrido
ROLLBACK TRAN
RETURN
END
--No hubo errores
COMMIT TRAN
```

Crear un procedimiento almacendado llamado InsertaReserva con los siguientes parámetros: nombre del usuario que va a realizar la inserción, contraseña del usuario que va a realizar la inserción, Fecha de ENTRADA , Fecha de SALIDA , IVA , Numero de HABITACION y CLIENTE. Y que tiene el siguiente comportamiento: Sólo se permite insertar reservas a los administradores y usuarios, pero no a los invitados. La inserción deberá realizarse en una transacción de manera que si se produce algún error se deberá hacer un ROLLBACK e informar del error. Si todo fue correcto se deberá confirmar la transacción.

Ejecutar las siguientes llamadas:

```
EXEC InsertaReserva 'admin', 'admin', '2009-03-15', '2009-03-25', 0.07,101, '12345';
EXEC InsertaReserva 'admin', 'admin', '2009-03-15', '2009-03-25', 0.07,101, '11111';
EXEC InsertaReserva 'invitado', '', '2009-03-15', '2009-03-25', 0.07,101, '11111';
EXEC InsertaReserva 'user', 'user', '2009-03-15', '2009-03-25', 0.07, 102, '12345'
EXEC InsertaReserva 'user', 'user','2009-02-16', '2009-02-21', 0.07,103, '12345';
EXEC InsertaReserva 'user', 'user','2009-03-16', '2009-03-21', 0.07,104, '44444';
EXEC InsertaReserva 'user', 'user','2009-03-16', '2009-03-21', 0.07,105, '44444';
EXEC InsertaReserva 'user', 'user','2009-03-16', '2009-03-21', 0.07,106, '44444';
EXEC InsertaReserva 'user', 'user','2009-03-16', '2009-03-21', 0.07,107, '44444';
```