EJERCICIOS RESUELTOS TEMA 7

7.1. Crea una base de datos "ejercicio7", con una única tabla "ciudades". Cada ciudad tendrá un código (clave primaria), un nombre (no nulo) y una cantidad de habitantes (que sí podrá tener valores nulos).

```
using System;
using System.Data.SQLite;

namespace _7._1
{
    class Program
    {
        static void Main()
        {
            SQLiteConnection conexion = new SQLiteConnection("Data Source =
..\\..\\..\\ejercicio7.sqlite; Version = 3; New = True; Compress = True");
            conexion.Open();

            string creacion = "CREATE TABLE ciudades (codigo varchar (8) PRIMARY
KEY, nombre varchar (15) NOT NULL, cantidadHabitantes numeric (7))";
            SQLiteCommand cmd = new SQLiteCommand(creacion, conexion);
            cmd.ExecuteNonQuery();

            conexion.Close();
            Console.WriteLine("Tabla creada con éxito.");
            Console.ReadLine();
        }
    }
}
```

7.2. Añade las ciudades: Boston (código BO, 4.180.000 habitantes) y Kyoto (código KY, sin indicar el número de habitantes).

```
using System;
using System.Data.SQLite;

namespace _7._2
{
    class Program
    {
        static void Main()
        {
            SQLiteConnection conexion = new SQLiteConnection("Data Source =
..\\..\\..\\ejercicio7.sqlite; Version = 3; New = False; Compress = True");
            conexion.Open();

            string insertar = "INSERT INTO ciudades (nombre, codigo,
cantidadHabitantes) VALUES ('Boston', 'BO', 4180000); " +
                "INSERT INTO ciudades (nombre, codigo, cantidadHabitantes) VALUES
('Kyoto', 'KY', NULL)";
            SQLiteCommand cmd = new SQLiteCommand(insertar, conexion);
            cmd.ExecuteNonQuery();

            Console.WriteLine("Datos insertados con éxito.");
            Console.ReadLine();
```

```
            string mostrar = "SELECT * FROM ciudades";
            cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader datos = cmd.ExecuteReader();

            while (datos.Read())
            {
                string codigo = Convert.ToString(datos[0]);
                string nombre = Convert.ToString(datos[1]);
                string habitantes = Convert.ToString(datos[2]);
                Console.WriteLine("Código: {0}, Nombre: {1}, Número de
habitantes: {2}", codigo, nombre, habitantes);
            }

            conexion.Close();
            Console.ReadLine();
        }
    }
}
```

7.3. Intenta añadir la ciudad Astana, sin código.

```
using System;
using System.Data.SQLite;

namespace _7._3
{
    class Program
    {
        static void Main()
        {
            SQLiteConnection conexion = new SQLiteConnection("Data Source =
..\\..\\..\\ejercicio7.sqlite; Version = 3; New = False; Compress = True");
            conexion.Open();

            string añadir = "INSERT INTO ciudades VALUES (NULL, 'Astana', NULL)";
//Funcionaría la primera vez,
            //pero la segunda daría error porque no se pueden insertar dos claves
primarias iguales.
            SQLiteCommand cmd = new SQLiteCommand(añadir, conexion);
            cmd.ExecuteNonQuery();

            conexion.Close();
            Console.Write("Datos insertados con éxito.");
            Console.ReadLine();
        }
    }
}
```

7.4. Intenta añadir la ciudad de código ELX pero de la que aún no sabemos el nombre.

```
using System;
using System.Data.SQLite;

namespace _7._4
{
    class Program
    {
```

```
        static void Main()
        {
            SQLiteConnection conexion = new SQLiteConnection("Data Source =
..\\..\\..\\ejercicio7.sqlite; Version = 3; New = False; Compress = True");
            conexion.Open();

            string añadir = "INSERT INTO ciudades VALUES ('ELX', NULL, NULL)";
//Salta la excepción porque el campo nombre no puede ser nulo.
            SQLiteCommand cmd = new SQLiteCommand(añadir, conexion);
            cmd.ExecuteNonQuery();

            conexion.Close();
            Console.Write("Datos insertados con éxito.");
            Console.ReadLine();
        }
    }
}
```

7.5. Muestra el nombre y población de todas las ciudades para las que sabemos la cantidad de habitantes, ordenadas de la menos poblada a la más poblada.

```
using System;
using System.Data.SQLite;

namespace _7._5
{
    class Program
    {
        static void Main()
        {
            SQLiteConnection conexion = new SQLiteConnection("Data Source =
..\\..\\..\\ejercicio7.sqlite; Version = 3; New = False; Compress = True");
            conexion.Open();

            string mostrar = "SELECT nombre, cantidadHabitantes FROM ciudades
WHERE cantidadHabitantes IS NOT NULL ORDER BY cantidadHabitantes";
            SQLiteCommand cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader datos = cmd.ExecuteReader();

            while (datos.Read())
            {
                string nombre = Convert.ToString(datos[0]);
                int habitantes = Convert.ToInt32(datos[1]);
                Console.WriteLine("Nombre: {0}, habitantes: {1}", nombre,
habitantes);
            }

            conexion.Close();
            Console.ReadLine();
        }
    }
}
```

7.6. Muestra el nombre de las ciudades que tienen 0 habitantes, ordenadas alfabéticamente.

```
using System;
using System.Data.SQLite;

namespace _7._6
{
    class Program
    {
        static void Main()
        {
            SQLiteConnection conexion = new SQLiteConnection("Data Source =
..\\..\\..\\ejercicio7.sqlite; Version = 3; New = False; Compress = True");
            conexion.Open();

            string mostrar = "SELECT nombre FROM ciudades WHERE
cantidadHabitantes = 0 ORDER BY nombre";
            SQLiteCommand cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader datos = cmd.ExecuteReader();

            while (datos.Read())
            {
                string nombre = Convert.ToString(datos[0]);
                Console.WriteLine("Nombre: {0}", nombre);
            }

            conexion.Close();
            Console.ReadLine();
        }
    }
}
```

7.7. Muestra el nombre de las ciudades para las que no conocemos la cantidad de habitantes, ordenadas alfabéticamente.

```
using System;
using System.Data.SQLite;

namespace _7._7
{
    class Program
    {
        static void Main()
        {
            SQLiteConnection conexion = new SQLiteConnection("Data Source =
..\\..\\..\\ejercicio7.sqlite; Version = 3; New = False; Compress = True");
            conexion.Open();

            string mostrar = "SELECT nombre FROM ciudades WHERE
cantidadHabitantes IS NULL ORDER BY nombre";
            SQLiteCommand cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader datos = cmd.ExecuteReader();

            while (datos.Read())
            {
                string nombre = Convert.ToString(datos[0]);
                Console.WriteLine("Nombre: {0}", nombre);
            }

            conexion.Close();
            Console.ReadLine();
```

```
        }
    }
}
```

7.8. Unifica todos los ejercicios en la misma clase.

```
using System;
using System.Data.SQLite;

namespace _7._2
{
    class Program
    {
        static void Main()
        {
            SQLiteConnection conexion = new SQLiteConnection("Data Source =
..\\..\\..\\ejercicio7.sqlite; Version = 3; New = False; Compress = True");
            conexion.Open();

            string insertar = "INSERT INTO ciudades (nombre, codigo,
cantidadHabitantes) VALUES ('Boston', 'BO', 4180000); " +
                "INSERT INTO ciudades (nombre, codigo, cantidadHabitantes) VALUES
('Kyoto', 'KY', NULL)";
            SQLiteCommand cmd = new SQLiteCommand(insertar, conexion);
            cmd.ExecuteNonQuery();

            Console.WriteLine("Datos insertados con éxito.");
            Console.ReadLine();

            string mostrar = "SELECT * FROM ciudades";
            cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader datos = cmd.ExecuteReader();

            while (datos.Read())
            {
                string codigo = Convert.ToString(datos[0]);
                string nombre = Convert.ToString(datos[1]);
                string habitantes = Convert.ToString(datos[2]);
                Console.WriteLine("Código: {0}, Nombre: {1}, Número de
habitantes: {2}", codigo, nombre, habitantes);
            }

            conexion.Close();
            Console.ReadLine();
        }
    }
}
```