EJERCICIOS RESUELTOS TEMA 10

10.1. Crea una base de datos "ejercicio10", en la que guardaremos información sobre selecciones nacionales de baloncesto. Para ello tendremos: una tabla "PAISES" y una tabla "JUGADORES", unidas por una relación 1:M (cada país podrá tener muchos jugadores y cada jugador sólo podrá formar parte -en un instante dado- de la selección de un país). De cada país guardaremos el nombre (por ejemplo, "España") y un código que actuará como clave primaria (por ejemplo, "ESP). De cada jugador anotaremos código, nombre, apellidos, posición y, como resultado de esa relación 1:M, código de la selección a la que pertenece.

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

namespace _10._1
{
    class Program
    {
        static void Main()
        {
            SQLiteConnection conexion = new SQLiteConnection("Data Source =
..\\..\\..\\ejercicio10.sqlite; Version = 3; New = True; Compress = True");
            conexion.Open();

            string crear = "CREATE TABLE paises (nombre varchar (15), codigo
varchar (5) PRIMARY KEY);" +
                "CREATE TABLE jugadores (codigo varchar (8), nombre varchar (15),
apellidos varchar (15), " +
                "posicion varchar (12), codigoSeleccion varchar (8))";
            SQLiteCommand cmd = new SQLiteCommand(crear, conexion);
            cmd.ExecuteNonQuery();

            conexion.Close();
            Console.WriteLine("Tablas creadas con éxito.");
            Console.ReadLine();
        }
    }
}
```

10.2a. Añade los países:

- ESP, España
- ARG, Argentina
- AUS, Australia
- LIT, Lituania

10.2b. Añade los jugadores:

- RUB, Ricky, Rubio, Base (España)
- NAV, Juan Carlos, Navarro, Alero (España)
- SCO, Luis, Scola, Ala-Pivot (Argentina)
- DEL, Carlos, Delfino, Escolta (Argentina)
- MAC, Jonas, Maciulis, Alero (Lituania)
- BOG, Andrew, Bogut, Pivot (Australia)

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

namespace _10._2
{
    class Program
    {
        static void Main()
        {
            SQLiteConnection conexion = new SQLiteConnection("Data Source =
..\\..\\..\\ejercicio10.sqlite; Version = 3; New = False; Compress = True");
            conexion.Open();

            string insertar = "INSERT INTO paises VALUES ('España', 'ESP');
INSERT INTO paises VALUES ('Argentina', 'ARG'); " +
                "INSERT INTO paises VALUES ('Australia', 'AUS'); INSERT INTO
paises VALUES ('Lituania', 'LIT'); " +
                "INSERT INTO jugadores VALUES ('RUB', 'Ricky', 'Rubio', 'Base',
'España');" +
                "INSERT INTO jugadores VALUES ('NAV', 'Juan Carlos', 'Navarro',
'Alero', 'España');" +
                "INSERT INTO jugadores VALUES ('SCO', 'Luis', 'Scola', 'Ala-
Pivot', 'Argentina');" +
                "INSERT INTO jugadores VALUES ('DEL', 'Carlos', 'Delfino',
'Escolta', 'Argentina');" +
                "INSERT INTO jugadores VALUES ('MAC', 'Jonas', 'Maciulis',
'Alero', 'Lituania');" +
                "INSERT INTO jugadores VALUES ('BOG', 'Andrew', 'Bogut', 'Pivot',
'Australia');";
            SQLiteCommand cmd = new SQLiteCommand(insertar, conexion);
            cmd.ExecuteNonQuery();

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

10.3. Muestra los nombres y apellidos de todos los jugadores, en mayúsculas, ordenados por apellido y nombre.

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

namespace _10._3
{
    class Program
    {
        static void Main()
        {
            SQLiteConnection conexion = new SQLiteConnection("Data Source =
..\\..\\..\\ejercicio10.sqlite; " +
                "Version = 3; New = False; Compress = True");
            conexion.Open();

            string mostrar = "SELECT UPPER (apellidos), UPPER (nombre) FROM
jugadores";
            SQLiteCommand cmd = new SQLiteCommand(mostrar, conexion);
```

```
            SQLiteDataReader reader = cmd.ExecuteReader();

            while (reader.Read())
            {
                string apellidos = Convert.ToString(reader[0]);
                string nombre = Convert.ToString(reader[1]);
                Console.WriteLine("Apellidos: {0}, Nombre: {1}", apellidos,
nombre);
            }

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

10.4. Muestra el nombre y apellidos del jugador o jugadores cuyo apellido es el más largo (formado por más letras).

EN SQLITE NO HAY CHAR_LENGTH, SINO LENGTH. ESTA FUNCIÓN DEVUELVE EL NÚMERO DE CARACTERES, NO DE BYTES COMO EN MYSQL.

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

namespace _10._4
{
    class Program
    {
        static void Main()
        {
            SQLiteConnection conexion = new SQLiteConnection("Data Source =
..\\..\\..\\ejercicio10.sqlite; " +
                "Version = 3; New = False; Compress = True");
            conexion.Open();

            string mostrar = @"SELECT apellidos, nombre FROM jugadores WHERE
LENGTH (apellidos) =
                                (SELECT MAX(longitud) FROM(SELECT
LENGTH(apellidos) as longitud FROM jugadores))";
            SQLiteCommand cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader reader = cmd.ExecuteReader();

            while (reader.Read())
            {
                string apellidos = Convert.ToString(reader[0]);
                string nombre = Convert.ToString(reader[1]);
                Console.WriteLine("Apellidos: {0}, Nombre: {1}", apellidos,
nombre);
            }

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

10.5. Muestra el apellido, una coma, un espacio y después el nombre de todos los jugadores de "España" (aparecerán datos como "Rubio, Ricky"). Para ello, usa la función "CONCAT". Los resultados deben aparecer como si se tratase de un campo llamado "nombreJug".

NO EXISTE LA FUNCIÓN CONCAT EN SQLITE. SE USA || PARA CONCATENAR.

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

namespace _10._5
{
    class Program
    {
        static void Main()
        {
            SQLiteConnection conexion = new SQLiteConnection("Data Source =
..\\..\\..\\ejercicio10.sqlite; " +
                "Version = 3; New = False; Compress = True");
            conexion.Open();

            string mostrar = "SELECT apellidos || ', ' || nombre as nombreJug
FROM jugadores WHERE codigoSeleccion = 'España'";
            SQLiteCommand cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader reader = cmd.ExecuteReader();

            while (reader.Read())
            {
                string nombreJug = Convert.ToString(reader[0]);
                Console.WriteLine("NombreJug: {0}", nombreJug);
            }

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

10.6. Muestra las 4 primeras letras de los apellidos de los jugadores que tenemos anotados de "Argentina", ordenados de forma descendente.

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

namespace _10._6
{
    class Program
    {
        static void Main()
        {
            SQLiteConnection conexion = new SQLiteConnection("Data Source =
..\\..\\..\\ejercicio10.sqlite; " +
               "Version = 3; New = False; Compress = True");
            conexion.Open();

            string mostrar = "SELECT SUBSTR (apellidos, 1, 4) FROM jugadores
WHERE codigoSeleccion = 'Argentina' ORDER BY apellidos DESC";
            SQLiteCommand cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader reader = cmd.ExecuteReader();

            while (reader.Read())
```

```
            {
                string apellidos = Convert.ToString(reader[0]);
                Console.WriteLine("Apellidos: {0}", apellidos);
            }

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

10.7. Muestra los nombres de todos los jugadores, reemplazando "Ricky" por "Ricard".

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

namespace _10._7
{
    class Program
    {
        static void Main()
        {
            SQLiteConnection conexion = new SQLiteConnection("Data Source =
..\\..\\..\\ejercicio10.sqlite; " +
              "Version = 3; New = False; Compress = True");
            conexion.Open();

            string mostrar = "SELECT REPLACE (nombre, 'Ricky', 'Ricard') FROM
jugadores";
            SQLiteCommand cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader reader = cmd.ExecuteReader();

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

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

10.8. Muestra "Don " seguido del nombre y del apellido de los jugadores (aparecerán datos como "Don Andrew Bogut"), usando "CONCAT" e "INSERT" para crear al vuelo un nuevo campo llamado "nombreJug".

NO EXISTEN LAS FUNCIONES CONCAT NI INSERT. HAY QUE HACER EL EJERCICIO CONCATENANDO CON ||

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

namespace _10._8
{
    class Program
    {
        static void Main()
```

```
        {
            SQLiteConnection conexion = new SQLiteConnection("Data Source =
..\\..\\..\\ejercicio10.sqlite; " +
              "Version = 3; New = False; Compress = True");
            conexion.Open();

            string mostrar = "SELECT 'Don ' || nombre || ' ' || apellidos as
nombreJug FROM jugadores";
            SQLiteCommand cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader reader = cmd.ExecuteReader();

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

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

10.9. Muestra el nombre y apellidos de todos los jugadores cuyo país contenga una N en el nombre. Debes eliminar los espacios iniciales y finales de ambos campos, en caso de que existan.

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

namespace _10._9
{
    class Program
    {
        static void Main()
        {
            SQLiteConnection conexion = new SQLiteConnection("Data Source =
..\\..\\..\\ejercicio10.sqlite; " +
             "Version = 3; New = False; Compress = True");
            conexion.Open();

            string mostrar = "SELECT TRIM (nombre), TRIM (apellidos) FROM
jugadores WHERE codigoSeleccion LIKE '%n%'";
            SQLiteCommand cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader reader = cmd.ExecuteReader();

            while (reader.Read())
            {
                string nombre = Convert.ToString(reader[0]);
                string apellidos = Convert.ToString(reader[1]);
                Console.WriteLine("Nombre: {0}, Apellidos: {1}", nombre,
apellidos);
            }

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

10.10. Muestra al revés el apellido de los jugadores de Australia que tenemos en nuestra base de datos.

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

namespace _10._10
{
    class Program
    {
        static void Main()
        {
            SQLiteConnection conexion = new SQLiteConnection("Data Source =
..\\..\\..\\ejercicio10.sqlite; " +
             "Version = 3; New = False; Compress = True");
            conexion.Open();

            string mostrar = "SELECT REVERSE (apellidos) FROM jugadores WHERE
codigoSeleccion = 'Australia'";
            SQLiteCommand cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader reader = cmd.ExecuteReader();

            while (reader.Read())
            {
                string apellidos = Convert.ToString(reader[0]);
                Console.WriteLine("Apellido al revés: {0}", apellidos);
            }

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

10.11. Muestra una cadena formada por 10 guiones, 10 espacios y otros 10 guiones.

NO EXISTE LA FUNCIÓN REPEAT EN SQLITE. HABRÍA QUE HACERLO CONCATENANDO.

10.12. Unifica todos los ejercicios en la misma clase.

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

namespace _10._12
{
    class Program
    {
        static SQLiteConnection conexion;

        static void Main()
        {
            EstablecerConexion();
            CrearBBDD();
            Console.WriteLine("Tablas creadas con éxito. Pulsa Intro para
continuar...");
            Console.ReadLine();
            InsertarDatos();
            Console.WriteLine("Datos insertados con éxito. Pulsa Intro para
continuar...");
            Console.ReadLine();
```

```
            Console.WriteLine("Mostrar en mayúsculas los nombres de los
jugadores: ");
            PrimeraConsulta();
            Console.WriteLine("Pulsa Intro para continuar...");
            Console.ReadLine();
            Console.WriteLine("Mostrar el nombre del jugador que tiene el
apellido más largo: ");
            SegundaConsulta();
            Console.WriteLine("Pulsa Intro para continuar...");
            Console.ReadLine();
            Console.WriteLine("Mostrar los nombres de los jugadores españoles:
");
            TerceraConsulta();
            Console.WriteLine("Pulsa Intro para continuar...");
            Console.ReadLine();
            Console.WriteLine("Mostrar las 4 primeras letras de los jugadores
argentinos: ");
            CuartaConsulta();
            Console.WriteLine("Pulsa Intro para continuar...");
            Console.ReadLine();
            Console.WriteLine("Mostrar todos los jugadores reemplazando 'Ricky'
por 'Ricard': ");
            QuintaConsulta();
            Console.WriteLine("Pulsa Intro para continuar...");
            Console.ReadLine();
            Console.WriteLine("Insertar 'Don' en los nombres de todos los
jugadores: ");
            SextaConsulta();
            Console.WriteLine("Pulsa Intro para continuar...");
            Console.ReadLine();
            Console.WriteLine("Mostrar los nombres de los jugadores cuyo país
tenga una 'N': ");
            SeptimaConsulta();
            Console.WriteLine("Pulsa Intro para continuar...");
            Console.ReadLine();
            Console.WriteLine("Mostrar al revés los nombres de los jugadores
australianos: ");
            OctavaConsulta();
            Console.WriteLine("Pulsa Intro para acabar...");
            Console.ReadLine();
            CerrarBBDD();
        }

        static void EstablecerConexion()
        {
            conexion = new SQLiteConnection(@"Data Source =
..\\..\\..\\ejercicio10.sqlite;
               Version = 3; New = True; Compress = True");
            conexion.Open();
        }

        static void CrearBBDD()
        {
            string crear = "CREATE TABLE paises (nombre varchar (15), codigo
varchar (5) PRIMARY KEY);" +
                "CREATE TABLE jugadores (codigo varchar (8), nombre varchar (15),
apellidos varchar (15), " +
                "posicion varchar (12), codigoSeleccion varchar (8))";
            SQLiteCommand cmd = new SQLiteCommand(crear, conexion);
            cmd.ExecuteNonQuery();
        }
```

```
        static void InsertarDatos()
        {
            string insertar = @"INSERT INTO paises VALUES ('España', 'ESP');
INSERT INTO paises VALUES ('Argentina', 'ARG'),
                                ('Australia', 'AUS'), ('Lituania', 'LIT');
                                INSERT INTO jugadores VALUES ('RUB', 'Ricky',
'Rubio', 'Base', 'España'),
                                ('NAV', 'Juan Carlos', 'Navarro', 'Alero',
'España'),
                                ('SCO', 'Luis', 'Scola', 'Ala-Pivot',
'Argentina'),
                                ('DEL', 'Carlos', 'Delfino', 'Escolta',
'Argentina'),
                                ('MAC', 'Jonas', 'Maciulis', 'Alero',
'Lituania'),
                                ('BOG', 'Andrew', 'Bogut', 'Pivot',
'Australia')";
            SQLiteCommand cmd = new SQLiteCommand(insertar, conexion);
            cmd.ExecuteNonQuery();
        }

        static void PrimeraConsulta()
        {
            string mostrar = "SELECT UPPER (apellidos), UPPER (nombre) FROM
jugadores";
            SQLiteCommand cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader reader = cmd.ExecuteReader();

            while (reader.Read())
            {
                string apellidos = Convert.ToString(reader[0]);
                string nombre = Convert.ToString(reader[1]);
                Console.WriteLine("Apellidos: {0}, Nombre: {1}", apellidos,
nombre);
            }
        }

        static void SegundaConsulta()
        {
            string mostrar = @"SELECT apellidos, nombre FROM jugadores WHERE
LENGTH (apellidos) =
                                (SELECT MAX(longitud) FROM(SELECT
LENGTH(apellidos) as longitud FROM jugadores))";
            SQLiteCommand cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader reader = cmd.ExecuteReader();

            while (reader.Read())
            {
                string apellidos = Convert.ToString(reader[0]);
                string nombre = Convert.ToString(reader[1]);
                Console.WriteLine("Apellidos: {0}, Nombre: {1}", apellidos,
nombre);
            }
        }

        static void TerceraConsulta()
        {
            string mostrar = "SELECT apellidos || ', ' || nombre as nombreJug
FROM jugadores WHERE codigoSeleccion = 'España'";
            SQLiteCommand cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader reader = cmd.ExecuteReader();
```

```
            while (reader.Read())
            {
                string nombreJug = Convert.ToString(reader[0]);
                Console.WriteLine("NombreJug: {0}", nombreJug);
            }
        }

        static void CuartaConsulta()
        {
            string mostrar = "SELECT SUBSTR (apellidos, 1, 4) FROM jugadores
WHERE codigoSeleccion = 'Argentina' ORDER BY apellidos DESC";
            SQLiteCommand cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader reader = cmd.ExecuteReader();

            while (reader.Read())
            {
                string apellidos = Convert.ToString(reader[0]);
                Console.WriteLine("Apellidos: {0}", apellidos);
            }
        }

        static void QuintaConsulta()
        {
            string mostrar = "SELECT REPLACE (nombre, 'Ricky', 'Ricard') FROM
jugadores";
            SQLiteCommand cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader reader = cmd.ExecuteReader();

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

        static void SextaConsulta()
        {
            string mostrar = "SELECT 'Don ' || nombre || ' ' || apellidos as
nombreJug FROM jugadores";
            SQLiteCommand cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader reader = cmd.ExecuteReader();

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

        static void SeptimaConsulta()
        {
            string mostrar = "SELECT TRIM (nombre), TRIM (apellidos) FROM
jugadores WHERE codigoSeleccion LIKE '%n%'";
            SQLiteCommand cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader reader = cmd.ExecuteReader();

            while (reader.Read())
            {
                string nombre = Convert.ToString(reader[0]);
                string apellidos = Convert.ToString(reader[1]);
                Console.WriteLine("Nombre: {0}, Apellidos: {1}", nombre,
apellidos);
```

```
            }
        }

        static void OctavaConsulta()
        {
            string mostrar = "SELECT REVERSE (apellidos) FROM jugadores WHERE
codigoSeleccion = 'Australia'";
            SQLiteCommand cmd = new SQLiteCommand(mostrar, conexion);
            SQLiteDataReader reader = cmd.ExecuteReader();

            while (reader.Read())
            {
                string apellidos = Convert.ToString(reader[0]);
                Console.WriteLine("Apellido al revés: {0}", apellidos);
            }
        }

        static void CerrarBBDD()
        {
            conexion.Close();
        }
    }
}
```