Practica 1: Bases De Datos Relacionales

UNIVERSIDAD
DE MÁLAGA

Alejandro Silva Rodríguez

Marta Cuevas Rodríguez

Almacenes De Datos
Universidad de Málaga

Septiembre 2024

Índice

1. Introducción

En esta práctica, se diseñará y creará una base de datos para gestionar información sobre parques naturales en Andalucía y la liga española de fútbol de primera división. El objetivo es elaborar un modelo entidad-relación y un modelo relacional que muestren las características y relaciones de las entidades involucradas, utilizando herramientas de diseño de bases de datos. Además, se generará el DDL (Data Definition Language) para SQL Server, que permitirá crear la base de datos y las tablas necesarias. Se incluirán datos sobre los parques naturales, su gestión y las especies que los habitan, así como información sobre los equipos de fútbol, sus jugadores y los partidos. Este ejercicio ayudará a aplicar los conceptos de bases de datos y ofrecerá una experiencia práctica en el uso de herramientas y lenguajes de consulta.

2. Parques Naturales

En esta sección se explicara como se diseñó e implementó la base de datos de parques naturales y las consultas sobre la misma.

2.1. Requisitos De Datos

La Junta de Andalucía desea mantener la información sobre los parques naturales que hay en su comunidad autónoma. En particular sería necesario conocer el nombre del parque (que es único), su teléfono, dirección administrativa, una dirección web, un correo electrónico, su fecha de declaración como parque natural, la extensión (en hectáreas) de cada zona protegida, las especies animales que contiene, la población estimada de cada una de ellas y la dirección gestora del parque. Esta dirección gestora del parque está coordinada por un presidente y un número no determinado de consejeros. De todos estos miembros de la dirección gestora se desea conocer el DNI, nombre, fecha de nacimiento, dirección y teléfono de cada uno de ellos. Cada persona puede ser a lo sumo consejero en un parque y presidente de otro. De las especies guardamos su nombre científico y el común (ambos son únicos), el número de años de vida media.

Con objeto de poder determinar el estado de salud del parque necesitamos información sobre la interacción del hombre con el entorno. Para ello se almacenan datos sobre los municipios donde está ubicado el parque: número de municipios que abarca, nombre de cada uno de ellos, enlace a su web, fichero con la foto de su escudo, partido que gobierna en la alcaldía, número de habitantes y gasto de agua medio por habitante.

Las especies tienen una extensión (en hectáreas) necesaria para desarrollarse en libertad, dato que aparece en los estudios generales sobre cada especie. Sin embargo, el dato de si la especie está superpoblada en cada parque se guardará explícitamente, porque puede depender de factores como si el parque es montañoso, si tiene acuíferos, etc y por tanto precisa de la opinión última de un experto. Tenga en cuenta que cada municipio abarca a lo sumo un sólo parque.

2.2. Diseño Lógico

Para crear el diseño lógico de la base de datos en Oracle Datamodeler [2] empezaremos representando las figuras más relevantes como entidades, entre ellas encontramos a especie animal, municipio, parque natural y persona (figura 1). Cada entidad tendrá atributos que solo pertenecen a ella y no se relacionan con ninguna entidad más excepto del atributo numero de municipios que abarca de un parque natural, tendrá que ir actualizandose mediante un trigger cada vez que se le añade o elimina una relación municipio.Tambien tendremos una entidad intermedia entre especie animal y parque natural que almacenará la relacion entre estas y los atributos extensión, población estimada y is_superpoblado (booleano).

Por ultimo, tenemos relaciones entre parque natural y municipio y dos entre parque natural y persona. Una relación 1:N representará los consejeros del parque natural y otra 1:1 representara el presidente del parque.

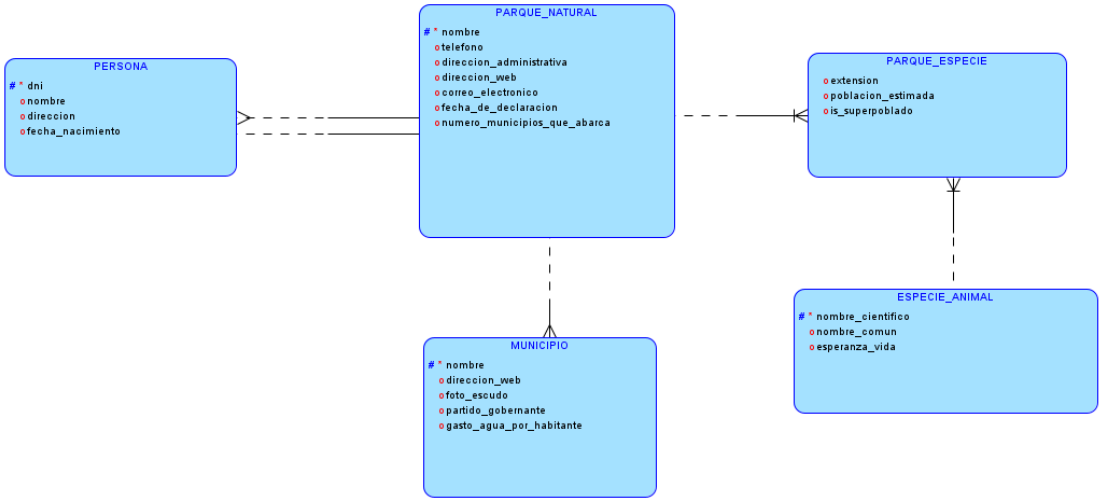

Figura 1: Diagrama Lógico De Parques Naturales

2.3. Diseño Entidad Relación

En el diseño entidad relación tenemos que tener en cuenta que las claves y restricciones están donde se necesitan para poder hacer consultas en la base de datos. Observamos (figura 2) que parque natural tiene una clave foranea dni que representa al unico presidente de esta entidad, mientras que persona tiene una clave foranea nombre de parque natural que relaciona a los consejeros con esta. Municipio obtiene la clave foranea de parque natural debido a su particion en la parte N de la relación. La entidad intermedia de especie animal y parque natural obtiene sus dos claves foraneas que pasan a ser su clave principal.

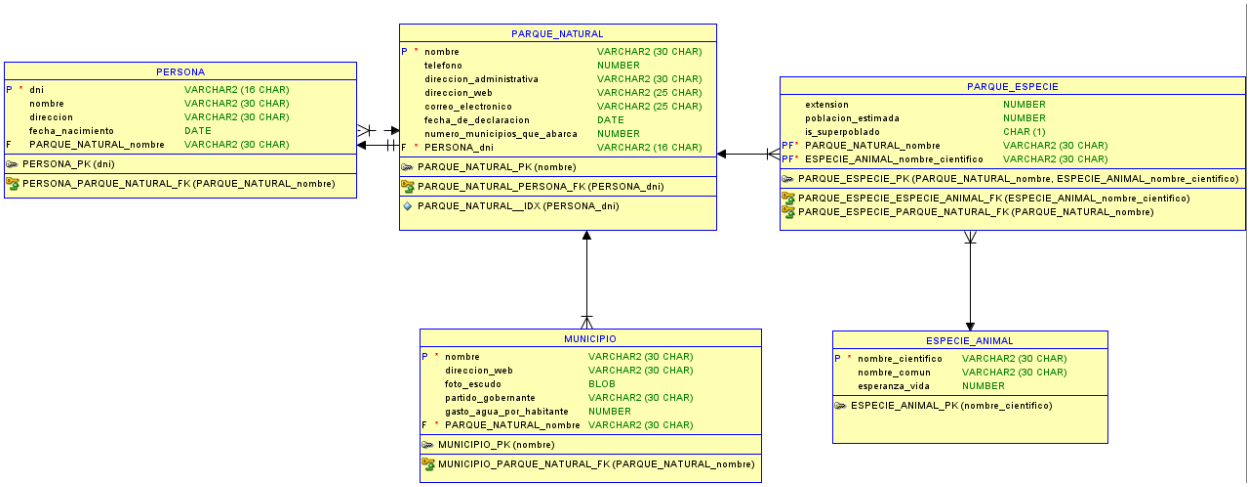

Figura 2: Diagrama Relacional De Parques Naturales

2.4. Implementación

En la implementación de la base de datos generaremos el ddl (data definition languaje) de la base de datos e insertaremos tuplas en la misma. En el listing 1 encontramos una pequeña parte del mismo donde se crea la tabla de especie animal. Esto se ejecutará en Microsoft SQL Server [1] donde se alojará la base de datos.

```
1
2       CREATE TABLE ESPECIE_ANIMAL
3       (
4       nombre_cientifico VARCHAR (30) NOT NULL ,
5       nombre_comun VARCHAR (30) ,
6       esperanza_vida NUMERIC (28)
7       )
8       GO
9
10      ALTER TABLE ESPECIE_ANIMAL ADD CONSTRAINT ESPECIE_ANIMAL_PK PRIMARY KEY
            CLUSTERED (nombre_cientifico)
11      WITH (
12      ALLOW_PAGE_LOCKS = ON ,
13      ALLOW_ROW_LOCKS = ON )
14      GO
```

Listing 1: Definicion De Datos De Parques Naturales

En el listing 2 se observan ejemplos de las inserciones que se hacen sobre la base de datos y el trigger que añadimos para la actualización del número de municipios. Es crucial seguir un orden logico a la hora de la inserción ya que algunas entidades dependen de otras.

```
        -- Insertar especies animales
        INSERT INTO ESPECIE_ANIMAL (nombre_cientifico, nombre_comun,
            esperanza_vida) VALUES
        ('Ursus arctos', 'Oso pardo', 25),
        -- Insertar personas (presidentes) - un presidente por parque natural
        INSERT INTO PERSONA (dni, nombre, direccion, fecha_nacimiento) VALUES
        ('34567890C', 'Luis Torres', 'Calle Larga 789', '1985-02-20'),  -- Cabo de
            Gata
        -- Insertar parques naturales
        INSERT INTO PARQUE_NATURAL (nombre, telefono, direccion_administrativa,
            direccion_web, correo_electronico, fecha_de_declaracion,
            numero_municipios_que_abarca, PERSONA_dni) VALUES
        ('Sierra Nevada', 123456789, 'Granada', 'www.sierranevada.com',
        ('Picos de Europa', 321321321, 'Asturias', 'www.picosdeeuropa.com', '
            contacto@picos.com', '1999-10-05', 0, '45678901D');
        -- Insertar consejeros
        INSERT INTO PERSONA (dni, nombre, direccion, fecha_nacimiento,
            PARQUE_NATURAL_nombre) VALUES
        ('23456789K', 'Valeria Torres', 'Avenida Costa 234', '1988-09-17', 'Cabo
            de Gata'),
        go
        -- Actualizar el numero de municipios del parque al insertar un municipio
            al que pertenece
        CREATE TRIGGER actualizar_numero_municipios_insert
        ON MUNICIPIO
        AFTER INSERT
        AS
        BEGIN
        -- Sumar 1 a cada parque natural por cada municipio insertado
        UPDATE PN
        SET PN.numero_municipios_que_abarca = PN.numero_municipios_que_abarca + (
        SELECT COUNT(*)
        FROM inserted I
        WHERE I.PARQUE_NATURAL_nombre = PN.nombre
        )
        FROM PARQUE_NATURAL PN
        WHERE EXISTS (
        SELECT 1 FROM inserted I WHERE I.PARQUE_NATURAL_nombre = PN.nombre
        );
        END;
        GO
        -- Insertar municipios
        INSERT INTO MUNICIPIO (nombre, direccion_web, foto_escudo,
            partido_gobernante, gasto_agua_por_habitante, PARQUE_NATURAL_nombre)
            VALUES
        ('Granada', 'www.granada.es', NULL, 'PSOE', 50.25, 'Sierra Nevada'),
        -- Insertar especies en parques
        INSERT INTO PARQUE_ESPECIE (extension, poblacion_estimada, is_superpoblado
            , PARQUE_NATURAL_nombre, ESPECIE_ANIMAL_nombre_cientifico) VALUES
        (10000, 200, 0, 'Sierra Nevada', 'Ursus arctos'),
```

Listing 2: Carga de datos

2.5. Consultas

Para probar el correcto funcionamiento de la base de datos realizaremos consultas (listing 3) que involucren varias tablas.

```
1
2       -- Consulta 1: Seleccionar los animales que se pueden encontrar en Sierra
            Nevada
3       SELECT e.nombre_cientifico, e.nombre_comun, p.nombre
4       FROM PARQUE_NATURAL p , PARQUE_ESPECIE pe, ESPECIE_ANIMAL e
5       WHERE e.nombre_cientifico=pe.ESPECIE_ANIMAL_nombre_cientifico and pe.
            PARQUE_NATURAL_nombre=p.nombre and p.nombre='Sierra Nevada';
6
7       -- Consulta 2: Seleccionar el nombre de los consejeros de Do ana
8
9       SELECT p.nombre, c.nombre
10      FROM PARQUE_NATURAL p, PERSONA c
11      WHERE c.PARQUE_NATURAL_nombre=p.nombre and p.nombre='Do ana';
12
13      -- Consulta 3: Seleccionar el nombre del presidente de Do ana
14
15      SELECT p.nombre, c.nombre
16      FROM PARQUE_NATURAL p, PERSONA c
17      WHERE c.dni=p.PERSONA_dni and p.nombre='Do ana';
18
19      -- Consulta 4: Seleccionar el nombre de los municipios que abarca do ana
            junto al numero de municipios que abarca
20
21      SELECT m.nombre, p.nombre, p.numero_municipios_que_abarca
22      FROM PARQUE_NATURAL p, MUNICIPIO m
23      WHERE p.nombre=m.PARQUE_NATURAL_nombre and p.nombre='Do ana';
24
25      -- Consulta 5: Seleccionar el nombre de los presidentes de paruqes en los
            que se encuentra algun tipo de oso
26
27      SELECT e.nombre_comun, presi.nombre, p.nombre
28      FROM PERSONA presi, ESPECIE_ANIMAL e, PARQUE_NATURAL p, PARQUE_ESPECIE pe
29      WHERE presi.dni=p.PERSONA_dni and p.nombre=pe.PARQUE_NATURAL_nombre and pe
            .ESPECIE_ANIMAL_nombre_cientifico=e.nombre_cientifico and e.
            nombre_comun LIKE 'Oso%';
```

Listing 3: Consultas Sobre Parques Naturales

3. Liga De Fútbol

En esta sección se describirá en detalle como se ha llevado a cabo el diseño y la implementación de la base de datos sobre la liga de fútbol, además de las correspondientes consultas.

3.1. Requisitos De Datos

Se quiere crear una base de datos que almacene información sobre la liga española de primera división. Esta información es anual (sólo datos de la liga en curso) y se recolectan los datos sobre los equipos que militan ese año en la categoría, su plantilla, cuerpo técnico y directivos, partidos en los que se enfrentan y resultados (parciales y globales de la liga).

Los equipos están identificados por su nombre y guardaremos además su número de socios, el nombre de su campo, la ciudad a la que pertenecen, el año de fundación, el número de años continuados en primera división y el nombre de su fundador.

Los equipos están compuestos por un entrenador, un médico, un preparador físico, un director deportivo, un entrenador de porteros, un presidente del club, varios consejeros y la plantilla de jugadores. De todo este personal se guarda su nombre, fecha de nacimiento, DNI, teléfono, dirección y sueldo. Además, de los jugadores se guarda el apodo o alias, el puesto en el equipo, los años para el fin del contrato, la cuantía para la claúsula de rescisión y el número de años en el equipo.

El campeonato de liga está compuesto por una serie de jornadas que se identifican con un número. Cada jornada está formada por un conjunto de partidos, que son enfrentamientos entre una pareja de equipos y se juegan en el campo de uno de los dos. Queremos tener asociados los partidos a cada jornada y deseamos conocer su resultado (5-0, 3-1, 0-0, etc.), la fecha y hora en que se celebraron, la recaudación por taquilla, el número de espectadores y las personas que forman el equipo arbitral (un árbitro principal, dos jueces de línea y un cuarto árbitro) . Además guardamos para cada jornada el total de goles marcados de cabeza, en propia meta y de penalti y la recaudación obtenida por medio de las quinielas de esa jornada.

Los colegiados (árbitros y jueces de línea) son seleccionados al principio de temporada para participar en esa categoría. De ellos se almacena el número de colegiado (que identifica a cada uno), nombre, DNI, antigüedad en la categoría y categoría en la que participó el año anterior. En cada temporada no son intercambiables los papeles de árbitro y juez de línea (un juez de línea no puede actuar como árbitro ni al revés). De los jueces de línea, además de los datos antes mencionados guardamos un dato que indique las posibilidades de desempeñar funciones de árbitro en la temporada siguiente y edad, y de los árbitros si ha sido o no internacional y si fue futbolista anteriormente.

3.2. Diseño Lógico

Para la creación del diseño lógico de nuestra base de datos utilizando Oracle Datamodeler [2], para ello comenzaremos representando los elementos más significativos, las entidades. En nuestra caso, como se muestra en la figura 3, encontramos las entidades jornada, partido, equipo, personal (que cuenta con un subtipo jugador) y colegiado que a su vez cuenta con dos subtipos, arbitro y juez de linea. Cada una cuenta con atributos propios en su mayoría, pero existen casos como los supertipos que compartirán clave principal con sus subtipos correspondientes.

El diagrama cuenta con seis relaciones entre equipo y personal, correspondientes a los diferentes trabajadores nombrados en los requisitos (tales como médicos, entrenadores, preparadores...) siendo la primera de estas la utilizada para consejeros ya que requiere una relación 1:N. Además existe una relación de dependencia entre jornada y partido, dos relaciones 1:1 entre partido y equipo referentes a los equipos local y visitante, otra entre equipo y los jugadores que participan en él; y por último, dos relaciones asociadas a los colegiados que arbitran el partido (árbitro principal y asistente), y otras dos para los jueces de línea.

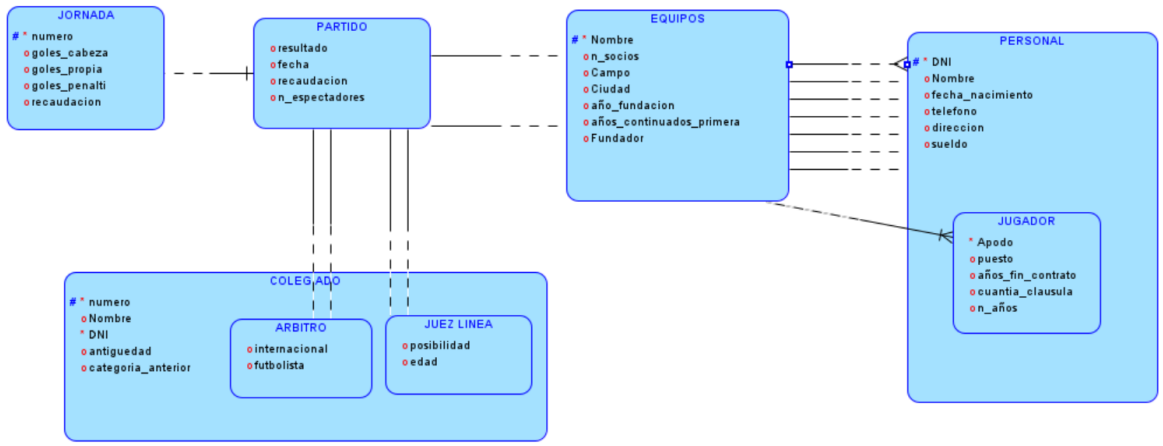

Figura 3: Diagrama Lógico De La Liga de Fútbol

3.3. Diseño Entidad Relación

Una vez acabado el diseño lógico, pasamos a la entidad relación. En este se muestran las claves migrantes que pasarán a otras entidades debido a las relaciones entre ellas (Figura 4), como por ejemplo los DNIs heredados por equipo debido a sus múltiples relaciones con personal, o la clave foránea de jugadores (DNI de nuevo) heredada de su supertipo, al igual que lo hacen los subtipos de colegiado. En esta sección podemos observar que se ha creado un nuevo atributo en colegiado llamado COLEGIADO TYPE que indicará si se trata de un árbitro o de un juez de línea. Tenemos también en partido los IDs de los jueces y árbitros que participarán en él y el número de jornada al que corresponde. Jugador indicará además el nombre del equipo al que pertenece.

Los consejeros al ser relación 1:N guardarán información sobre el equipo al que pertenecen dentro de personal.

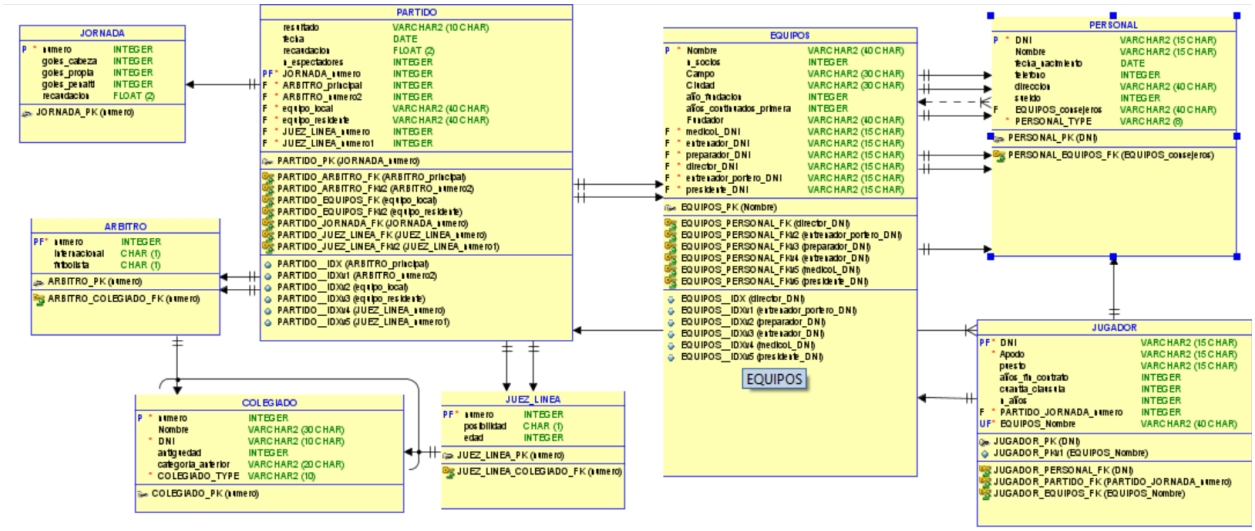

Figura 4: Diagrama Relacional De La Liga de Fútbol

3.4. Implementación

En la implementación de la base de datos generaremos el ddl (data definition languaje) e insertaremos tuplas en la misma. En el listing 4 encontramos una pequeña parte del mismo donde se crea la tabla equipos con sus claves de las relaciones. Esto se ejecutará en Microsoft SQL Server [1] donde se alojará la base de datos.

```
1
2         CREATE TABLE EQUIPOS
3         (
4         Nombre VARCHAR (40) NOT NULL ,
5         n_socios INTEGER ,
6         Campo VARCHAR (30) ,
7         Ciudad VARCHAR (30) ,
8         a o_fundacion INTEGER ,
9         a os_continuados_primera INTEGER ,
10        Fundador VARCHAR (40) ,
11        medicoL_DNI VARCHAR (15) NOT NULL ,
12        entrenador_DNI VARCHAR (15) NOT NULL ,
13        preparador_DNI VARCHAR (15) NOT NULL ,
14        director_DNI VARCHAR (15) NOT NULL ,
15        entrenador_portero_DNI VARCHAR (15) NOT NULL ,
16        presidente_DNI VARCHAR (15) NOT NULL
17        )
18        GO
19
20        CREATE UNIQUE NONCLUSTERED INDEX
21        EQUIPOS__IDX ON EQUIPOS
22        (
23        director_DNI
24        )
25        GO
26
27        CREATE UNIQUE NONCLUSTERED INDEX
28        EQUIPOS__IDXv1 ON EQUIPOS
29        (
30        entrenador_portero_DNI
31        )
32        GO
33
34        CREATE UNIQUE NONCLUSTERED INDEX
35        EQUIPOS__IDXv2 ON EQUIPOS
36        (
37        preparador_DNI
38        )
39        GO
40
41        CREATE UNIQUE NONCLUSTERED INDEX
42        EQUIPOS__IDXv3 ON EQUIPOS
43        (
44        entrenador_DNI
45        )
46        GO
47
48        CREATE UNIQUE NONCLUSTERED INDEX
49        EQUIPOS__IDXv4 ON EQUIPOS
50        (
```

```
        medicoL_DNI
        )
        GO

        CREATE UNIQUE NONCLUSTERED INDEX
        EQUIPOS__IDXv5 ON EQUIPOS
        (
        presidente_DNI
        )
        GO

        ALTER TABLE EQUIPOS ADD CONSTRAINT EQUIPOS_PK PRIMARY KEY CLUSTERED (
            Nombre)
        WITH (
        ALLOW_PAGE_LOCKS = ON ,
        ALLOW_ROW_LOCKS = ON )
        GO
```

Listing 4: Definicion De Datos De La Liga de Fútbol

En el listing 5 se observan ejemplos de las inserciones que se hacen sobre la base de datos, siguiendo un orden lógico para la correcta inserción teniendo en cuenta ciertas dependencias.

```
        -- Insercion de colegiados (incluyendo jueces de l nea)
        INSERT INTO COLEGIADO (numero, Nombre, DNI, antiguedad, categoria_anterior
            , COLEGIADO_TYPE)
        VALUES
        (1, 'Antonio Mateu', '12345678A', 10, 'Segunda Divisi n', 'ARBITRO'),
        (2, 'Carlos del Cerro', '23456789B', 12, 'Segunda Divisi n B', 'ARBITRO')
            ,
        (3, 'Jos  Luis Munuera', '34567890C', 8, 'Segunda Divisi n', 'ARBITRO'),
        (4, 'Alejandro Hern ndez', '45678901D', 9, 'Segunda Divisi n', 'ARBITRO'
            ),
        (5, 'Jes s Gil Manzano', '56789012E', 11, 'Segunda Divisi n', 'ARBITRO')
            ,
        (6, 'Luis Mart nez', '67890123F', 7, 'Segunda Divisi n', 'JUEZ_LINEA'),
        (7, 'Javier L pez', '78901234G', 6, 'Segunda Divisi n', 'JUEZ_LINEA'),
        (8, 'Fernando D az', '89012345H', 5, 'Segunda Divisi n', 'JUEZ_LINEA');

        -- Inserci n de arbitros
        INSERT INTO ARBITRO (numero, internacional, futbolista)
        VALUES
        (1, 1, 0),
        (2, 0, 0),
        (3, 1, 0),
        (4, 0, 1),
        (5, 1, 0);
```

Listing 5: Carga de datos

A continuación se muestra en el listing 6 una alteración de la tabla jugador previa a la inserción de los datos, debido a una restricción que prohibía que dos jugadores pertenecieran a un mismo equipo.

```
ALTER TABLE JUGADOR
DROP CONSTRAINT JUGADOR_PKv1;

-- Insercion de jugadores
INSERT INTO JUGADOR (DNI, Apodo, puesto, a os_fin_contrato,
    cuantia_clausula, n_a os, PARTIDO_JORNADA_numero, EQUIPOS_Nombre)
VALUES
('DNI1001', 'Benzema', 'Delantero', 2025, 100000000, 5, 1, 'Real Madrid'),
('DNI1002', 'Modric', 'Centrocampista', 2024, 50000000, 8, 1, 'Real Madrid
    '),
('DNI1003', 'Ter Stegen', 'Portero', 2026, 80000000, 5, 2, 'FC Barcelona')
    ,
('DNI1004', 'Pedri', 'Centrocampista', 2026, 150000000, 6, 2, 'FC
    Barcelona'),
('DNI1005', 'Koke', 'Centrocampista', 2025, 60000000, 4, 3, 'Atletico
    Madrid'),
('DNI1006', 'Navas', 'Defensa', 2024, 30000000, 3, 4, 'Sevilla FC');
```

Listing 6: Corrección de restricción

3.5. Consultas

Una vez implementados correctamente todos los datos y rellenadas las tablas, procedemos a la realización de consultas (listing 7).

```
1
2         --CONSULTA 1: Salario medio de los entrenadores de los equipos
3         SELECT AVG(p.sueldo) as salario_medio_entrenadores
4         FROM EQUIPOS e, PERSONAL p
5         WHERE e.entrenador_DNI = p.dni;
6
7         --CONSULTA 2: Nombre, antiguedad y tipo de los colegiados con mas de 6
              a os de antiguedad
8         SELECT Nombre,antiguedad, COLEGIADO_TYPE
9         FROM COLEGIADO
10        WHERE antiguedad>6
11        ORDER BY antiguedad;
12
13        --CONSULTA 3: Muestra el nombre de los arbitros que han participado en al
              menos dos partidos y que son internacionales
14        SELECT distinct( c.Nombre)
15        FROM COLEGIADO c
16        JOIN ARBITRO a ON c.numero = a.numero
17        JOIN PARTIDO p ON p.ARBITRO_principal = c.numero OR p.ARBITRO_numero2 = c.
              numero
18        WHERE a.internacional = 1
19        GROUP BY c.Nombre
20        HAVING COUNT(c.numero) >= 2;
21
22        --CONSULTA 4: Muestra los goles totales de cada jornadas
23        SELECT numero, SUM(goles_cabeza + goles_propia + goles_penalti) as
              total_goles
24        FROM JORNADA
25        GROUP BY numero;
26
27        --CONSULTA 5: Recaudacion media por espectador en cada partido
28        SELECT equipo_local, equipo_residente, recaudacion/n_espectadores as
              recaudacion_media
29        FROM PARTIDO;
```

Listing 7: Consultas Sobre La Liga de Fútbol

4. Acceso al Repositorio

Toda la información adicional, incluyendo el código fuente y la documentación completa de este proyecto, está disponible en el repositorio de GitHub [3].

Referencias

[1] Microsoft Corporation. *Microsoft SQL Server 2022 Documentation*. Microsoft Corporation, Redmond, Washington, 2022. Versión SQL Server 2022.

[2] Oracle Corporation. *Oracle SQL Developer Data Modeler User's Guide*. Oracle Corporation, Redwood Shores, California, 2024. Versión 23.2.

[3] Alex Silva. Practica1_almacenes_de_datos. https://github.com/AlexSilvaa9/Practica1$_a lmacenes_d e_d atos$, 2024.