En este documento se podrá ver paso a paso como se crea una base de datos sobre fútbol, incluyendo una serie de tablas como estadios, jugadores, equipos… El documento se divide en varios apartados, los cuales se pueden ver en el índice en la página siguiente.

Trabajo Bases de Datos 2ª evaluación

Base de Datos sobre fútbol

Antonio Salces Alcaraz y Sergii Butrii (1º DAM)

ÍNDICE

1. Introducción

Este documento detalla el proceso de diseño e implementación de una base de datos orientada a la gestión integral de información relacionada con el fútbol. Se abordan desde la definición de requerimientos y el modelado conceptual, hasta el diseño lógico y la implementación física en SQL Server. Este proyecto permite gestionar datos de estadios, jugadores, equipos, árbitros y partidos, garantizando la integridad referencial y optimizando las consultas mediante índices y restricciones específicas, creando también vistas para ofrecer mayor seguridad a la BB.DD.

El objetivo principal de este proyecto es desarrollar una base de datos robusta y eficiente que cumpla con los requerimientos funcionales y técnicos planteados. Con ello, se busca facilitar la consulta y administración de datos relacionados con el fútbol, proporcionando una guía didáctica que demuestre las buenas prácticas en el diseño y la gestión de bases de datos. El alcance del proyecto incluye desde la definición de la especificación hasta la implementación de transacciones y vistas para la consulta de información.

2. Escribir la especificación lo más detallada posible para el diseño de una base de datos de al menos tres tablas. Utilizando una lista de viñetas o numerada para la lista de requisitos que debe cumplir la aplicación

Se desea modelar una base de datos que almacena información sobre información relacionada a futbol, incluyendo estadios, jugadores, equipos… Las tablas serán las siguientes:

1. Estadio

- La aplicación debe almacenar la información de cada estadio.
- **Requisitos:**
  - Cada estadio tendrá un código único (CodEst) que se genera de forma autoincrementable.
  - Se almacenará el nombre del estadio (NomEst) y la ciudad (CiuEst).
  - La combinación de nombre y ciudad debe ser única para evitar duplicados (restricción UNIQUE).

2. Persona (Supertipo)

- La base de datos debe mantener la información de las personas relacionadas con el fútbol, que en este caso se utiliza para modelar a los árbitros.
- **Requisitos:**
  - Cada persona se identificará con un código autogenerado (CodPer).
  - Se almacenará el nombre (NomPer) y la nacionalidad (NacPer).

3. Árbitro (Subtipo de Persona)

- La aplicación debe distinguir a los árbitros como un subtipo de la entidad persona.
- **Requisitos:**
  - Cada árbitro se identifica por su código de persona (CodPer), que debe existir en la tabla persona.
  - Se almacenará la licencia del árbitro (LicArb).
  - Las acciones de actualización y borrado en la tabla persona deben propagarse a la tabla árbitro mediante restricciones en cascada.

4. Equipo

- La aplicación debe gestionar la información de cada equipo de fútbol.
- **Requisitos:**
  - Cada equipo tendrá un código único (CodEqu) autogenerado, un nombre (NomEqu) y la ciudad (CiuEqu).
  - Se almacenará, además, el código del jugador que actúa como capitán (CodCap), el cual se establecerá mediante una restricción de clave foránea hacia la tabla jugador. En caso de que se elimine el jugador, el valor se pondrá a NULL.

5. Jugador

- Se debe almacenar la información específica de cada jugador.
- **Requisitos:**
  - Cada jugador tendrá un código único (CodPer) autogenerado, que identifica al jugador de forma independiente (aunque en otros diseños se podría aprovechar el supertipo, en este caso se define en una tabla aparte).
  - Se registrará la posición del jugador (PosJug).
  - Se establecerá una relación con el equipo al que pertenece (CodEqu), garantizando la integridad referencial.

6. Partido

- La aplicación debe gestionar los encuentros o partidos de fútbol.
- **Requisitos:**
  - Cada partido tendrá un código único (CodPar) autogenerado.
  - Se registrará la fecha del partido (FecPar) y el resultado (ResPar), asignando por defecto el valor "Por definir" si aún no se conoce.
  - Se almacenará el estadio donde se juega (CodEst), el árbitro asignado (CodArb), y los equipos participantes: el equipo local (LocEqu) y el visitante (VisEqu).
  - Debe existir una restricción que asegure que el equipo local y el visitante sean distintos (CHECK).

7. Jugador_Partido (Tabla de Relación)

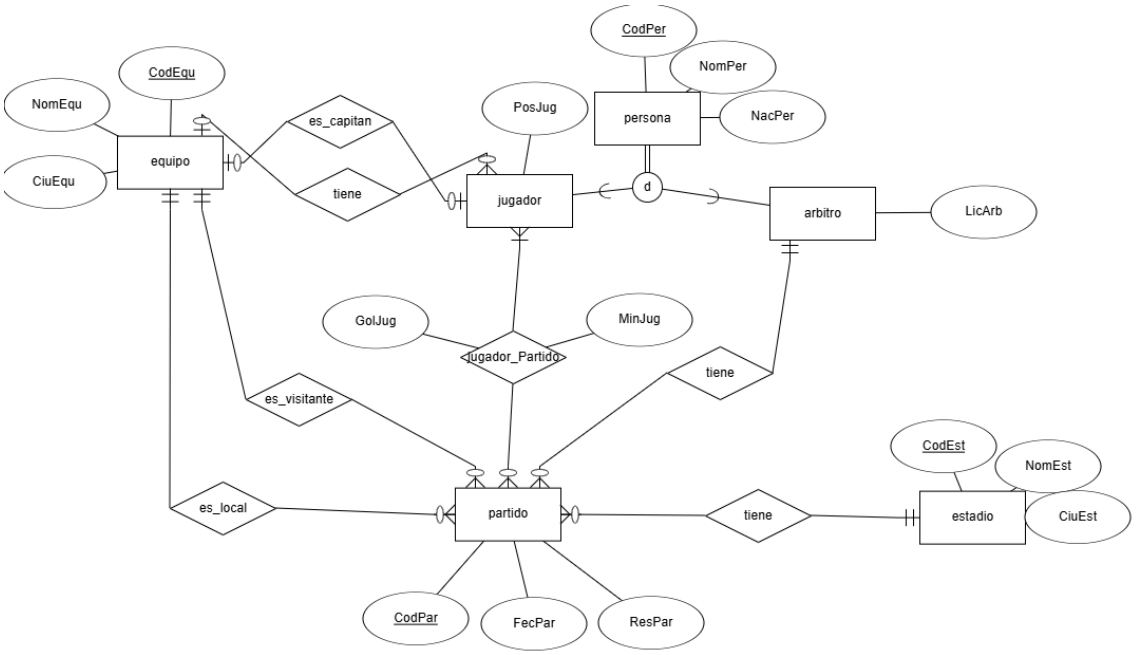
- La aplicación debe registrar la participación de los jugadores en cada partido.
- **Requisitos:**
  - Se implementará una relación muchos a muchos entre la tabla jugador y la tabla partido.
  - Cada registro contendrá el número de goles marcados (GolJug) y los minutos jugados (MinJug) por un jugador en un partido.
  - La clave primaria será compuesta por el código del jugador (CodPer) y el código del partido (CodPar), y se establecerán restricciones de integridad referencial (FK) con acciones en cascada para las actualizaciones y eliminaciones.

8. Optimización de Consultas

- La aplicación debe incluir mecanismos para optimizar el rendimiento de las consultas.
- **Requisito:**
  - Se creará un índice en la tabla equipo sobre el campo NomEqu para facilitar las búsquedas por nombre.

3. Realizar el Diseño Conceptual de Datos realizando el Modelo Entidad – Relación lo más expresivo posible. Utiliza ERD Plus. En este apartado debes poner captura del modelo

En este apartado se presenta el Modelo Entidad-Relación elaborado mediante ERD Plus. El diagrama ilustra las entidades principales (Estadio, Persona, Árbitro, Equipo, Jugador, Partido y Jugador_Partido) y las relaciones que existen entre ellas. Se ha diseñado un modelo que garantiza la integridad de los datos a través de restricciones y claves primarias/foráneas, asegurando que cada entidad se relacione de forma coherente con las demás.



4. Realizar el Diseño Lógico de Datos. Para ello debes hacer el Diagrama Referencial en ERD Plus y el Diagrama de Estructura de Datos (DED) en MySql WorkBench. En este apartado debes poner captura de ambos diagramas

El diseño lógico traduce el modelo conceptual en un esquema relacional detallado. En este apartado se muestran los diagramas referencial y de estructura de datos, donde se especifican las claves primarias, claves foráneas y las restricciones de integridad necesarias para mantener la consistencia de la información. Estos diagramas han sido elaborados utilizando ERD Plus y MySQL Workbench, y se presentan a continuación para facilitar su análisis.

Diagrama referencial con ERD Plus:

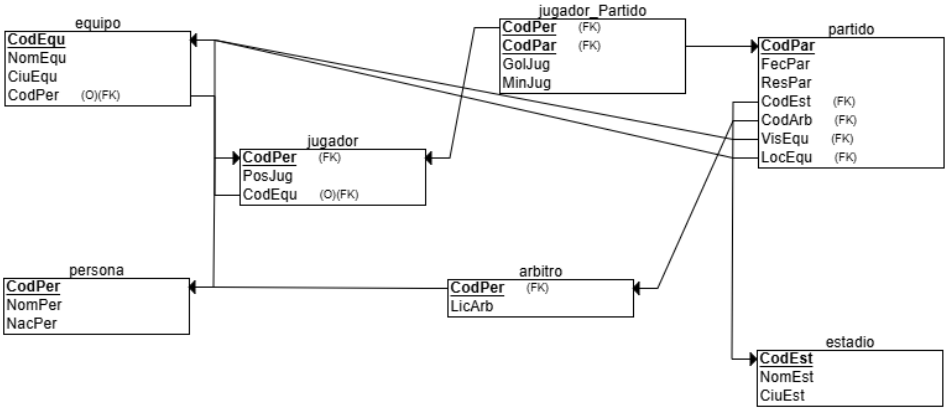
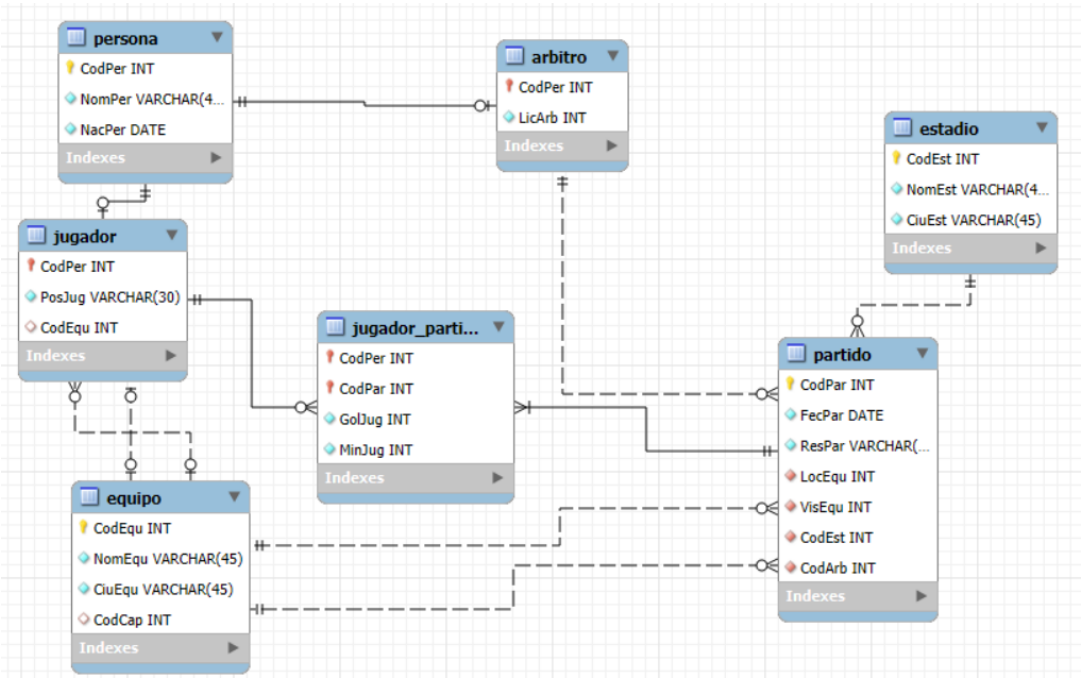


Diagrama de Estructura de Datos con MySql Wordbench



5. Realizar el Diseño Físico de Datos en SQL Server. Para la creación de las tablas utilizar todas las restricciones estudiadas en clase y especificar las reglas de borrado y modificación en todas la foreign key. Escribe en este apartado todas las sentencias correspondientes

El diseño físico se implementa en SQL Server, donde se crea la estructura real de la base de datos. A continuación, se describen las sentencias SQL utilizadas para la creación de tablas, la definición de restricciones (como UNIQUE, CHECK y las acciones de actualización/borrado en cascada) y la implementación de índices. Cada bloque de código incluye comentarios que explican su propósito y la lógica detrás de cada decisión .

```
-- Crear tabla ESTADIO
CREATE TABLE estadio
(
  CodEst INT IDENTITY(1,1) NOT NULL,  -- Código del estadio, con autoincremento
  NomEst VARCHAR(45) NOT NULL,        -- Nombre del estadio
  CiuEst VARCHAR(54) NOT NULL,        -- Ciudad del estadio
  PRIMARY KEY (CodEst),
  CONSTRAINT UQ_Estadio UNIQUE (NomEst, CiuEst) -- Un estadio debe ser único por
nombre y ciudad
);

-- Crear tabla PERSONA (Supertipo)
CREATE TABLE persona
(
  CodPer INT IDENTITY(1,1) NOT NULL,  -- Código de la persona, con autoincremento
  NomPer VARCHAR(45) NOT NULL,        -- Nombre de la persona
  NacPer VARCHAR(45) NOT NULL,        -- Nacionalidad
  PRIMARY KEY (CodPer)
);

-- Crear tabla ARBITRO (Subtipo de persona)
CREATE TABLE arbitro
(
  LicArb INT NOT NULL,        -- Licencia del árbitro
  CodPer INT NOT NULL,                -- Código de la persona (PK en persona)
  PRIMARY KEY (CodPer),
  FOREIGN KEY (CodPer) REFERENCES persona(CodPer)
    ON DELETE CASCADE                 -- Si se elimina la persona, se elimina el
árbitro
    ON UPDATE CASCADE                 -- Si se actualiza el CodPer en persona, se
actualiza en arbitro
);

-- Crear tabla EQUIPO (sin FK hacia jugador por el momento)
CREATE TABLE equipo
(
  CodEqu INT IDENTITY(1,1) NOT NULL,  -- Código del equipo, con autoincremento
  NomEqu VARCHAR(45) NOT NULL,        -- Nombre del equipo
  CiuEqu VARCHAR(45) NOT NULL,        -- Ciudad del equipo
  CodCap INT,                       -- Código del jugador que es capitán (relación
futura)
  PRIMARY KEY (CodEqu)
);

-- Crear tabla JUGADOR (con FK hacia equipo)
CREATE TABLE jugador
```

```
(
  PosJug VARCHAR(30) NOT NULL,         -- Posición del jugador
  CodPer INT IDENTITY(1,1) NOT NULL,   -- Código de la persona (PK en persona)
  CodEqu INT,                          -- Código del equipo al que pertenece
  PRIMARY KEY (CodPer),
  FOREIGN KEY (CodEqu) REFERENCES equipo(CodEqu) -- Relación con equipo
    ON DELETE NO ACTION                  -- Si se elimina el equipo, se pone NULL
en CodEqu
    ON UPDATE NO ACTION                 -- Si se actualiza el CodEqu en equipo, se
actualiza en jugador
);

-- Modificar tabla EQUIPO para añadir la FK hacia JUGADOR (capitán)
ALTER TABLE equipo
ADD CONSTRAINT fk_equipo_capitan FOREIGN KEY (CodCap) REFERENCES jugador(CodPer)
    ON DELETE SET NULL                      -- Si se intenta eliminar el jugador
capitán, no lo dejara
    ON UPDATE CASCADE;                   -- Si se intenta actualizar el CodPer en
jugador, no dejara mientras sea el CodCap

-- Crear tabla PARTIDO
CREATE TABLE partido
(
  CodPar INT IDENTITY(1,1) NOT NULL,   -- Código del partido, con autoincremento
  FecPar DATE NOT NULL,                -- Fecha del partido
  ResPar VARCHAR(20) DEFAULT 'Por definir',-- Resultado del partido (ej. "2-1")
  CodEst INT NOT NULL,                 -- Estadio donde se juega
  CodArb INT NOT NULL,                 -- Código del árbitro (CodPer en arbitro)
  VisEqu INT NOT NULL,                 -- Equipo visitante
  LocEqu INT NOT NULL,                          -- Equipo local
  PRIMARY KEY (CodPar),
  FOREIGN KEY (CodEst) REFERENCES estadio(CodEst)
    ON DELETE NO ACTION                 -- No se puede eliminar un estadio si hay
partidos asociados
    ON UPDATE CASCADE,                  -- Si se actualiza el CodEst en estadio, se
actualiza en partido
  FOREIGN KEY (CodArb) REFERENCES arbitro(CodPer)
    ON DELETE NO ACTION                 -- No se puede eliminar un árbitro si está
asignado a un partido
    ON UPDATE CASCADE,                  -- Si se actualiza el CodPer en arbitro, se
actualiza en partido
  FOREIGN KEY (VisEqu) REFERENCES equipo(CodEqu)
    ON DELETE NO ACTION                 -- No se puede eliminar un equipo si está
en un partido
    ON UPDATE NO ACTION,                   -- No se puede actualizar CodEqu si esta
registrado en un partido. SQL Server solo permite una ruta de CASADE POR TABLE.
  FOREIGN KEY (LocEqu) REFERENCES equipo(CodEqu)
    ON DELETE NO ACTION                 -- No se puede eliminar un equipo si está
en un partido
    ON UPDATE NO ACTION,                -- Si se actualiza el CodEqu en equipo, se
actualiza en partido
  CHECK (VisEqu <> LocEqu), --Asegura que el equipo local y visitante sean distintos
);

-- Crear tabla JUGADOR_PARTIDO (Relación muchos a muchos entre jugador y partido)
CREATE TABLE jugador_Partido
(
  GolJug INT DEFAULT   NOT NULL,       -- Goles marcados en el partido, por defecto
0
  MinJug INT NOT NULL,                 -- Minutos jugados
  CodPer INT NOT NULL,                 -- Referencia al jugador (su PK en jugador)
  CodPar INT NOT NULL,                 -- Referencia al partido
```

```
  PRIMARY KEY (CodPer, CodPar),
  FOREIGN KEY (CodPer) REFERENCES jugador(CodPer)
    ON DELETE CASCADE                  -- Si se elimina el jugador, se elimina su
participación en partidos
    ON UPDATE CASCADE,                 -- Si se actualiza el CodPer en jugador, se
actualiza en jugador_Partido
  FOREIGN KEY (CodPar) REFERENCES partido(CodPar)
    ON DELETE CASCADE                  -- Si se elimina el partido, se elimina su
relación con los jugadores
    ON UPDATE CASCADE                  -- Si se actualiza el CodPar en partido, se
actualiza en jugador_Partido
);

-- Crear índice en la tabla EQUIPO para mejorar el rendimiento de consultas por
nombre

CREATE INDEX idx_equipo_nombre ON equipo (NomEqu);
```

6. Insertar datos coherentes en las tablas (INSERT). Hacer modificación de datos en las tablas (UPDATE) y borrado de algunas filas (DELETE). Escribe en este apartado todas las sentencias correspondientes. Para las modificaciones y borrados, escribe también los enunciados

Esta sección detalla las operaciones de inserción, actualización y eliminación de datos en la base de datos. Se incluyen ejemplos prácticos que demuestran el correcto funcionamiento de las restricciones definidas en el diseño físico. Además, se ilustra el uso de transacciones para garantizar la consistencia de la información durante operaciones complejas, mostrando la importancia del ON DELETE CASCADE y otros mecanismos de integridad referencial.

Inserción de datos:

```sql
-- Insertar en la tabla ESTADIO
INSERT INTO estadio (NomEst, CiuEst)
VALUES
('Estadio Nacional', 'Ciudad de México'),
('La Bombonera', 'Buenos Aires'),
('Camp Nou', 'Barcelona'),
('Santiago Bernabéu', 'Madrid'),
('Old Trafford', 'Manchester');

-- Insertar en la tabla PERSONA
INSERT INTO persona (NomPer, NacPer)
VALUES
('Juan Pérez', 'México'),
('Carlos González', 'Argentina'),
('Luis Suárez', 'Uruguay'),
('Cristiano Ronaldo', 'Portugal'),
('Lionel Messi', 'Argentina');

-- Insertar en la tabla ARBITRO
INSERT INTO arbitro (LicArb, CodPer)
VALUES
(1001, 1),
(1002, 2),
(1003, 3),
(1004, 4),
(1005, 5);

-- Insertar en la tabla EQUIPO
INSERT INTO equipo (NomEqu, CiuEqu, CodCap)
VALUES
('Chivas', 'Guadalajara', NULL),
('Boca Juniors', 'Buenos Aires', NULL),
('Barcelona', 'Barcelona', NULL),
('Real Madrid', 'Madrid', NULL),
('Manchester United', 'Manchester', NULL);

-- Insertar en la tabla JUGADOR
INSERT INTO jugador (PosJug, CodEqu)
VALUES
('Delantero', 1),
('Defensa', 1),
('Mediocampista', 2),
('Delantero', 2),
```

```
('Portero', 3),
('Defensa', 3),
('Delantero', 4),
('Mediocampista', 4),
('Delantero', 5),
('Mediocampista', 5);

-- Modificar la tabla EQUIPO para asignar un capitán (cuando ya tengas jugadores
asignados)
UPDATE equipo
SET CodCap = 1
WHERE CodEqu = 1;  -- Asumiendo que el CodPer 1 es el capitán de Chivas

UPDATE equipo
SET CodCap = 3
WHERE CodEqu = 2;  -- Asumiendo que el CodPer 3 es el capitán de Boca Juniors

UPDATE
SET CodCap = 5
WHERE CodEqu = 3;  -- Asumiendo que el CodPer 5 es el capitán de Barcelona

UPDATE
SET CodCap = 7
WHERE CodEqu = 4;  -- Asumiendo que el CodPer 7 es el capitán de Real Madrid

UPDATE
SET CodCap = 9
WHERE CodEqu = 5;  -- Asumiendo que el CodPer 9 es el capitán de Manchester United

-- Insertar en la tabla PARTIDO
INSERT INTO partido (FecPar, ResPar, CodEst, CodArb, VisEqu, LocEqu)
VALUES
--Partidos en 2024
('2024-05-10', '2-1', 1, 1, 1, 2),  -- Chivas vs Boca Juniors
('2024-05-11', '0-3', 2, 2, 2, 3),  -- Boca Juniors vs Barcelona
('2024-05-12', '1-1', 3, 3, 3, 4),  -- Barcelona vs Real Madrid
('2024-05-13', '3-2', 4, 4, 4, 5),  -- Real Madrid vs Manchester United
('2024-05-14', '4-0', 5, 5, 5, 1),  -- Manchester United vs Chivas

--Partidos en 2025
('2025-01-06', '2-1', 1, 1, 1, 3),  -- Chivas vs Barcelona
('2025-01-07', '0-0', 2, 2, 2, 4),  -- Boca Juniors vs Real Madrid
('2025-01-08', '3-1', 3, 3, 3, 5),  -- Barcelona vs Manchester United
('2025-01-09', '1-2', 4, 4, 4, 1),  -- Real Madrid vs Chivas
('2025-01-10', '2-0', 5, 5, 5, 2),  -- Manchester United vs Boca Juniors

('2025-02-01', '2-1', 1, 1, 1, 2),  -- Chivas vs Boca Juniors
('2025-02-02', '3-0', 2, 2, 2, 3),  -- Boca Juniors vs Barcelona
('2025-02-03', '1-1', 3, 3, 3, 4),  -- Barcelona vs Real Madrid
('2025-02-04', '0-2', 4, 4, 4, 5),  -- Real Madrid vs Manchester United
('2025-02-05', '4-1', 5, 5, 5, 1);  -- Manchester United vs Chivas

INSERT INTO partido (FecPar, CodEst, CodArb, VisEqu, LocEqu)
VALUES
--Partidos en 2025 no jugados todavía
('2025-04-01', 1, 1, 2, 3),  -- Chivas vs Barcelona
('2025-04-02', 2, 2, 3, 4),  -- Boca Juniors vs Real Madrid
('2025-04-03', 3, 3, 4, 5),  -- Barcelona vs Manchester United
('2025-04-04', 4, 4, 5, 1),  -- Real Madrid vs Chivas
('2025-04-05', 5, 5, 1, 2),  -- Manchester United vs Boca Juniors

('2025-07-01', 1, 1, 3, 1),  -- Chivas vs Real Madrid
```

```
('2025-07-02', 2, 2, 4, 2),  -- Boca Juniors vs Manchester United
('2025-07-03', 3, 3, 5, 3),  -- Barcelona vs Chivas
('2025-07-04', 4, 4, 1, 4),  -- Real Madrid vs Boca Juniors
('2025-07-05', 5, 5, 2, 5);  -- Manchester United vs Barcelona

-- Insertar en la tabla JUGADOR_PARTIDO
INSERT INTO jugador_Partido (GolJug, MinJug, CodPer, CodPar)
VALUES
(1, 45, 1, 1),  -- Juan Pérez (jugador 1) en el partido 1
(0, 90, 2, 1),  -- Carlos González (jugador 2) en el partido 1
(2, 75, 3, 2),  -- Luis Suárez (jugador 3) en el partido 2
(1, 60, 4, 2),  -- Cristiano Ronaldo (jugador 4) en el partido 2
(0, 90, 5, 3),  -- Lionel Messi (jugador 5) en el partido 3
(0, 85, 6, 3),  -- Jugador 6 en el partido 3
(1, 70, 7, 4),  -- Jugador 7 en el partido 4
(0, 90, 8, 4),  -- Jugador 8 en el partido 4
(3, 60, 9, 5),  -- Jugador 9 en el partido 5
(0, 90, 10, 5); -- Jugador 10 en el partido 5


SELECT * FROM jugador;
SELECT * FROM partido;
SELECT * FROM jugador_Partido;
SELECT * FROM persona;
SELECT * FROM equipo;
SELECT * FROM arbitro;
SELECT * FROM partido;
```

Borrado y modificación de datos:

```
/*Enunciado:
Se requiere eliminar de la base de datos al equipo "Boca Juniors" con CodEqu = 2,
asegurando la integridad.
Dado que existen dependencias en otras tablas, antes de eliminar el equipo, es
necesario eliminar primero:
1. Las participaciones de sus jugadores en partidos (tabla 'jugador_partido').
2. Los partidos en los que Boca Juniors haya participado como equipo local o
visitante (tabla 'partido').
3. Los jugadores del equipo (tabla 'jugador').
4. Finalmente, eliminar al equipo Boca Juniors de la tabla `equipo`.
Demostrando así que el ON DELETE CASCADE tiene mucha utilidad, sobre todo si no
esta capado como es en SQL Server, que solo permite una ruta de CASCADE POR TABLE.
Todo esto debe realizarse dentro de una transacción para garantizar la consistencia
de los datos y evitar errores en caso de fallos intermedios.*/

BEGIN TRANSACTION;

-- Eliminar registros de jugador_partido asociados a jugadores de Boca Juniors
DELETE FROM jugador_partido
WHERE CodPer IN (SELECT CodPer FROM jugador WHERE CodEqu = 2);

-- Eliminar partidos donde Boca Juniors participa como equipo local o visitante
DELETE FROM partido
WHERE LocEqu = 2 OR VisEqu = 2;

-- Quitar el capitán de equipo antes de eliminar los jugadores para prevenir Errores
UPDATE
SET CodCap = NULL
WHERE CodEqu = 2;
```

```sql
-- Eliminar jugadores del equipo Boca Juniors
DELETE FROM jugador
WHERE CodEqu = 2;

-- Eliminar el equipo Boca Juniors
DELETE FROM equipo
WHERE CodEqu = 2;

SELECT * FROM equipo;

ROLLBACK TRANSACTION;
```

7. Realizar consultas sobre dicha base de datos. Las consultas constaran de enunciado y sentencia. Las consultas deben abarcar la mayoría de lo que se ha explicado en clase sobre la sentencia SELECT

En este apartado se presentan diversas consultas SQL diseñadas para extraer información relevante de la base de datos. Las consultas abordan diferentes escenarios, desde la obtención de listas de jugadores y partidos hasta la realización de filtros complejos utilizando funciones como JOIN, LIKE, EXISTS, IN, ANY y BETWEEN. Cada consulta va acompañada de un enunciado que explica su propósito y los resultados esperados.

```
/*Consulta1: Queremos obtener una lista completa de los jugadores de un equipo
llamado Chivas, con todos sus datos relevantes.
Esto incluye el nombre del jugador, su nacionalidad, su posición, el nombre del
equipo al que pertenece y la ciudad del equipo.*/SELECT
    p.NomPer AS NombreJugador,
    p.NacPer AS Nacionalidad,
    j.PosJug AS Posicion,
    e.NomEqu AS NombreEquipo,
    e.CiuEqu AS CiudadEquipo
FROM jugador j JOIN persona p ON (j.CodPer = p.CodPer) JOIN equipo e ON (j.CodEqu
= e.CodEqu)
WHERE e.NomEqu = 'Chivas';  -- Aquí seleccionas el equipo que deseas consultar


/*Consulta2: Queremos obtener los próximos partidos de un equipo específico,
mostrando la fecha, el resultado (si ya se ha jugado),
el nombre del estadio donde se jugará el partido y el nombre del árbitro asignado.
Además, en base de lo que tengas haz otro en el cual filtra los partidos
que aún no se han jugado (es decir, aquellos cuya fecha es posterior a la fecha
actual).*/

/*Primera parte */
SELECT
    p.FecPar AS FechaPartido,
    p.ResPar AS Resultado,
    es.NomEst AS Estadio,
    es.CiuEst AS CiudadEstadio,
    per.NomPer AS Arbitro
FROM partido p JOIN estadio es ON (p.CodEst = es.CodEst) JOIN arbitro a ON (p.CodArb
= a.CodPer) JOIN persona per ON (a.CodPer = per.CodPer)
WHERE (p.VisEqu = 1 OR p.LocEqu = 1)   -- Filtrar por equipo, '1' es el equipo
seleccionado (por ejemplo, Chivas)
ORDER BY p.FecPar;  -- Ordenar los partidos por fecha

/*Segunda parte*/
SELECT
    p.FecPar AS FechaPartido,
    p.ResPar AS Resultado,
    es.NomEst AS Estadio,
    es.CiuEst AS CiudadEstadio,
    per.NomPer AS Arbitro
FROM partido p JOIN estadio es ON (p.CodEst = es.CodEst) JOIN arbitro a ON (p.CodArb
= a.CodPer) JOIN persona per ON (a.CodPer = per.CodPer)
WHERE (p.FecPar > GETDATE())  -- Solo los partidos posteriores a la fecha actual
    AND (p.VisEqu = 1 OR p.LocEqu = 1)
ORDER BY p.FecPar;
```

```sql
/*Consulta3: Realizar una consulta que devuelva los partidos jugados entre "Chivas"
y otros equipos,
que hayan tenido al menos un gol marcado por el jugador "Juan Pérez" (CodPer = 1).
Utilizando las funciones LIKE, IN, y EXISTS.*/
SELECT p.CodPar, p.FecPar, p.ResPar, e1.NomEqu AS Equipo_Local, e .NomEqu AS
Equipo_Visitante
FROM partido p JOIN equipo e1 ON (p.LocEqu = e1.CodEqu) JOIN equipo e2 ON (p.VisEqu
= e2.CodEqu)
WHERE (e1.NomEqu LIKE 'Chivas' OR e2.NomEqu LIKE 'Chivas')  -- Filtrar partidos
con Chivas
  AND EXISTS (  -- Verifica que Juan Pérez haya jugado en el partido
      SELECT
      FROM jugador_Partido jp
      WHERE jp.CodPar = p.CodPar
      AND jp.CodPer = 1  -- Verificar si Juan Pérez jugó en el partido
      AND jp.GolJug > 0  -- Verificar que haya marcado gol
);


/*Consulta4: Realizar una consulta que devuelva todos los equipos que hayan jugado
partidos en "El Estadio Nacional" (CodEst = 1),
con una fecha de partido entre 2024-05-10 y 2025-02-05, y que hayan tenido un
resultado de "2-1" o "1-2".
Utilizando las funciones BETWEEN, IN, y ANY.*/
SELECT e.NomEqu, p.FecPar, p.ResPar
FROM equipo e JOIN partido p ON (e.CodEqu = p.LocEqu) OR (e.CodEqu = p.VisEqu)
WHERE p.CodEst = 1  -- Estadio Nacional
  AND p.FecPar BETWEEN '2024-05-10' AND '2025-02-05'  -- Fecha entre 2024-05-10 y
2025-02-05
  AND p.ResPar IN ('2-1', '1-2');  -- Resultado 2-1 o 1-2


/*Consulta5: Realizar una consulta para encontrar todos los jugadores que jugaron
en un partido en el que su equipo haya jugado
como visitante en "La Bombonera" (CodEst = 2). Utilizando las funciones ANY y
EXISTS.*/
SELECT j.PosJug, p.NomPer
FROM jugador j JOIN persona p ON (j.CodPer = p.CodPer)
WHERE EXISTS (SELECT 1
              FROM partido pa
              WHERE p .CodEst = 2  -- Estadio La Bombonera
              AND (p .VisEqu = j.CodEqu)  -- Jugador pertenece a equipo visitante
              AND pa.CodPar = ANY (SELECT CodPar
                                   FROM jugador_Partido jp
                                   WHERE jp.CodPer = j.CodPer
      )  -- El jugador jugó en alguno de los partidos en La Bombonera
);
```

8. Modifica la estructura de al menos una tabla. Crea algún índice útil

Para mejorar el rendimiento de las consultas, se ha creado un índice en la tabla EQUIPO basado en el campo NomEqu, lo que facilita las búsquedas por nombre. Asimismo, se han realizado modificaciones en la estructura de la tabla EQUIPO, asignando la clave foránea para el capitán de equipo.

Modificación de tabla:

```
-- Modificar tabla EQUIPO para añadir la FK hacia JUGADOR (capitán)
ALTER TABLE
ADD CONSTRAINT fk_equipo_capitan FOREIGN KEY (CodCap) REFERENCES jugador(CodPer)
    ON DELETE SET NULL                      -- Si se intenta eliminar el jugador
capitán, no lo dejara
    ON UPDATE CASCADE;                  -- Si se intenta actualizar el CodPer en
jugador, no dejara mientras sea el CodCap
```

Creación de índice:

```
-- Crear índice en la tabla EQUIPO para mejorar el rendimiento de consultas por
nombre
CREATE INDEX idx_equipo_nombre ON equipo (NomEqu);
```

9. Crea una vista y consúltala

Se ha creado una vista denominada 'vista_partidos_resumen' que consolida información relevante de los partidos jugados hasta la fecha actual. Esta vista simplifica la consulta de datos y permite un acceso rápido a la información resumida, facilitando la generación de reportes y análisis del desempeño de los equipos.

```
CREATE VIEW vista_partidos_resumen AS
SELECT
    p.FecPar,                    -- Fecha del partido
    p.ResPar,                    -- Resultado del partido
    e1.NomEqu AS Equipo_Local,   -- Nombre del equipo local
     2.NomEqu AS Equipo_Visitante, -- Nombre del equipo visitante
    per.NomPer AS Arbitro        -- Nombre del árbitro
FROM partido p JOIN equipo e1 ON (p.LocEqu = e1.CodEqu) JOIN equipo e2 ON (p.VisEqu
= e2.CodEqu) JOIN arbitro a ON (p.CodArb = a.CodPer) JOIN persona per ON (a.CodPer
= per.CodPer)
WHERE p.FecPar <= GETDATE();  -- Solo partidos que hayan ocurrido hasta la fecha
actual

SELECT * FROM vista_partidos_resumen;
```

10. Conclusión

El desarrollo de esta base de datos ha permitido implementar una solución integral para la gestión de información futbolística. Se han aplicado buenas prácticas en el diseño conceptual, lógico y físico, y se han demostrado técnicas avanzadas para garantizar la integridad y el rendimiento del sistema. Además, la implementación de transacciones y vistas refuerza la robustez del proyecto. Este trabajo no solo cumple con los requisitos de la evaluación, sino que también sienta las bases para futuras mejoras y ampliaciones en la gestión de datos.