Memoria

-- Práctica 1 --

Bases de Datos

2º Grado Ing. Informática, Curso 2021-2022

Álvaro Seral Gracia - 819425 - 819425@unizar.es
Cristian Andrei Selivanov Dobrisan - 816456 - 816456@unizar.es
Dorian Boleslaw Wozniak - 817570 - 817570@unizar.es

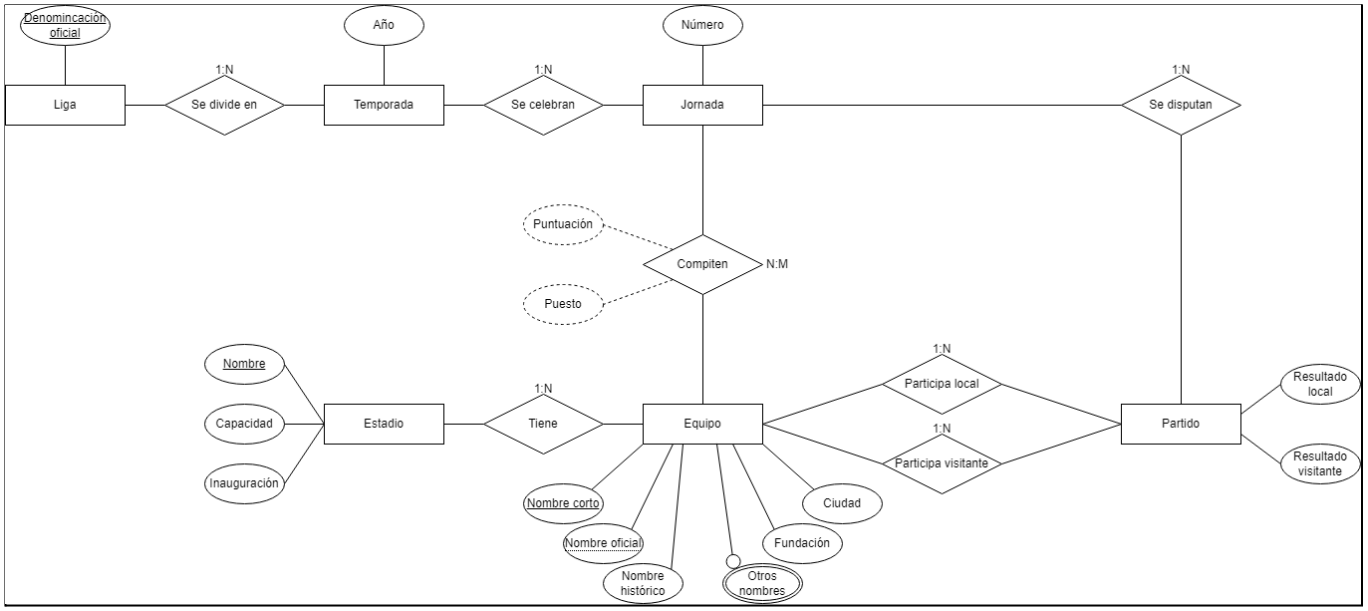
4 de abril de 2022

Parte 1:

Esquema E/R y restricciones.

Esquema E/R:

El modelo E/R que se ha diseñado para esta primera práctica de Bases de Datos es el siguiente.



Durante el desarrollo del esquema, surgió la necesidad de decidir si los tipos de entidades “Temporada”, “Jornada” y “Partido” debían ser débiles y depender respectivamente de “Liga”, “Temporada” y, en el caso de “Partido”, de “Jornada” y dos “Equipos”; o debían ser fuertes de modo que cada uno de los tipos de entidades tuviera una clave artificial. Después de evaluar ambos diseños se optó el modelo con tipos de entidades fuertes, bajo el razonamiento de que a pesar de sus claves son menos intuitivas al no estar relacionadas con los datos ofrecía una mayor facilidad y rapidez a la hora de realizar consultas por su brevedad y menor dependencia. Por esto último también es más eficiente en memoria el hecho de utilizar tipos de entidades fuertes.

Otra elección que se tuvo en cuenta fue al elegir el atributo identificador primario de “Equipo”. Lo lógico era pensar en utilizar “Nombre oficial”, pero como es más común que se utilice “Nombre corto” para denominar a “Equipo”, y ya que este también es único y no nulo, se optó por que el atributo identificador primario fuera el “Nombre corto” y el “Nombre oficial” pasara a ser un atributo identificador alternativo.

En relación a este tipo de entidad, como muchos “Equipos” no disponían de apodos y otros disponían de varios, “Otros nombres” pasó a ser un atributo opcional y multivaluado.

Con el fin de poder acceder a los resultados acumulados de cada equipo hasta cada una de las jornadas de cada temporada se añadió un tipo de interrelación entre “Equipo” y “Jornada” que contuviese dos atributos derivados: “Puntuación”, que almacena el total de puntos que ha obtenido el equipo en esa temporada hasta esa jornada, y “Puesto”, que contiene la posición con respecto al resto de equipos en esa jornada concreta.

Como a través de Puesto y conociendo la Liga en la que jugaba un determinado Equipo se podía hallar rápidamente si dicho Equipo iba a ascender, descender o ir a Europa, se decidió no incorporarlo en el Esquema E/R para obtenerlos a través de futuras consultas.

Restricciones:

- Para cada partido, los equipos que participan deben ser diferentes.
- Para cada jornada, un equipo no puede participar más de una vez.
- Para cada temporada, dos equipos no pueden enfrentarse entre sí más de dos veces, siendo una local-visitante y la otra visitante-local.
- No puede haber un equipo en dos ligas diferentes en la misma temporada.
- Si un equipo va a ascender, no puede descender. Si un equipo va a descender, no puede ascender.
- Un equipo en primera división no puede ascender.
- A Europa únicamente van los primeros 6 equipos de la Liga de primera división.

Esquema relacional y normalización.

Dominios:

tpNombre = cadena(50);
tpDivisión = (1a, 2a, Promoción 1a/2a, Descenso a 2a);
tpTemporada = cadena(9);
tpAño = number(4);
tpDecena = number(2);

Esquema relacional (antes de normalizar):

Como en el esquema E/R los tipos de entidades “Temporada”, “Jornada” y “Partido” no tenían ningún atributo identificador primario, se han añadido claves artificiales en el esquema relacional.

Liga (
    Denominación oficial : tpDivisión);
Temporada (
    idTemporada : tpNatural;
    Año : tpTemporada, NO NULO;
    clvLiga : tpDivisión, clave ajena de **Liga**);
Jornada (
    idJornada : tpNatural;
    Número : tpDecena, NO NULO;
    clvTemporada : tpNatural, clave ajena de **Temporada**);
Partido (
    idPartido : tpNatural;
    Resultado local, Resultado visitante : tpDecena, NO NULO;
    clvJornada : tpNatural, clave ajena de **Jornada**;
    clvEquipoLocal, clvEquipoVisitante : tpNombre, clave ajena de **Equipo**);
Equipo (
    Nombre corto : tpNombre;
    Nombre oficial : tpNombre, ÚNICO, NO NULO;

Nombre histórico, Ciudad : tpNombre, NO NULO;
Otros nombres : tpNombre;
Fundación : tpAño, NO NULO;
clvEstadio : tpNombre, clave ajena de **Estadio**);

Estadio (
Nombre : tpNombre;
Capacidad : tpNatural, NO NULO;
Inauguración : tpAño, NO NULO);

Compiten (
Puntuación : tpNatural, NO NULO;
Puesto : tpDecena, NO NULO;
clvJornada : tpDivisión, clave ajena de **Jornada**;
clvEquipo : tpNombre, clave ajena de **Equipo**);

Normalización:

El esquema relacional anterior se ha normalizado hasta dejar todas sus relaciones en Primera Forma Normal Boyce-Codd. Para ello se verificó que cada una de las formas normales anteriores se cumpliese.

Como el tipo de entidad Equipo contiene un atributo multivaluado, Otros nombres, dicha relación no cumple la Primera Forma Normal. Para corregirlo se ha creado una nueva relación débil, “Apodos”, cuya clave primaria está compuesta por “Nombre corto” y “Otro nombre”. De esta manera se ha podido eliminar el atributo multivaluado de “Equipo”. Tras estos cambios, todas las relaciones están en la Primera Forma Normal.

Como ninguna clave primaria de las relaciones es compuesta, se verifica que está en la Segunda Forma Normal. Si se hubiera utilizado un modelo E/R con entidades débiles como el propuesto anteriormente habría que haber corregido el esquema relacional para que estuviera en Segunda Forma Normal, puesto que, por ejemplo, la clave primaria de “Temporada” estaría compuesta por la clave primaria de “Liga” y por una clave artificial.

Como no hay dependencias transitivas entre los atributos que no son clave, todas las relaciones están en Tercera Forma Normal. Además, como todas las dependencias funcionales dependen de claves, las relaciones también se encuentran en Forma Normal Boyce-Codd.

Esquema relacional (después de normalizar):

Como en el esquema E/R los tipos de entidades “Temporada”, “Jornada” y “Partido” no tenían ningún atributo identificador primario, se han añadido claves artificiales en el esquema relacional.

Liga (
Denominación oficial : tpDivisión);

Temporada (
idTemporada : tpNatural;
Año : tpTemporada, NO NULO;
clvLiga : tpDivisión, clave ajena de **Liga**);

Jornada (
idJornada : tpNatural;
Número : tpDecena, NO NULO;
clvTemporada : tpNatural, clave ajena de **Temporada**);

Partido (
idPartido : tpNatural;
Resultado local, Resultado visitante : tpDecena, NO NULO;
clvJornada : tpNatural, clave ajena de **Jornada**;
clvEquipoLocal, clvEquipoVisitante : tpNombre, clave ajena de **Equipo**);

Equipo (
Nombre corto : tpNombre;
Nombre oficial : tpNombre, ÚNICO, NO NULO;
Nombre histórico, Ciudad : tpNombre, NO NULO;
Fundación : tpAño, NO NULO;
clvEstadio : tpNombre, clave ajena de **Estadio**);

Estadio (
Nombre : tpNombre;
Capacidad : tpNatural, NO NULO;
Inauguración : tpAño, NO NULO);

Compiten (
Puntuación : tpNatural, NO NULO;
Puesto : tpDecena, NO NULO;
clvJornada : tpDivisión, clave ajena de **Jornada**;
clvEquipo : tpNombre, clave ajena de **Equipo**);

Apodo (
Otro nombre : tpNombre;
Nombre corto : tpNombre, clave ajena de **Equipo**);

Sentencias SQL de creación de tablas.

Siguiendo el modelo relacional anteriormente descrito, se han creado las siguientes sentencias de creación de tablas de SQL:

```
CREATE TABLE Liga (
    clvLiga         VARCHAR(20)     PRIMARY KEY
);

CREATE TABLE Temporada (
    clvTemporada    NUMBER(4)       PRIMARY KEY,
    anyo            CHAR(9)         NOT NULL,
    clvLiga         VARCHAR(20)     REFERENCES Liga(clvLiga)
);

CREATE TABLE Jornada (
    clvJornada      NUMBER(6)       PRIMARY KEY,
    numero          NUMBER(2)       NOT NULL,
    clvTemporada    NUMBER(4)       REFERENCES Temporada(clvTemporada)
);

CREATE TABLE Estadio (
    clvEstadio      VARCHAR(50)     PRIMARY KEY,
    capacidad       NUMBER(6)       NOT NULL,
    inauguracion    NUMBER(4)       NOT NULL
);
```

```
CREATE TABLE Equipo (
    clvEquipo       VARCHAR(20)     PRIMARY KEY,
    nomOficial      VARCHAR(60)     NOT NULL UNIQUE,
    nomHistorico    VARCHAR(60)     NOT NULL,
    ciudad          VARCHAR(60)     NOT NULL,
    fundacion       NUMBER(4)       NOT NULL,
    clvEstadio      VARCHAR(50)     REFERENCES Estadio(clvEstadio)
);

CREATE TABLE Apodo (
    clvEquipo       VARCHAR(20)     REFERENCES Equipo(clvEquipo),
    otroNombre      VARCHAR(50),
    PRIMARY KEY (clvEquipo, otroNombre)
);

CREATE TABLE Partido (
    clvPartido      NUMBER          PRIMARY KEY,
    resLocal        NUMBER(2)       NOT NULL CHECK (resLocal >= 0),
    resVisit        NUMBER(2)       NOT NULL CHECK (resVisit >= 0),
    clvJornada      NUMBER(6)       REFERENCES Jornada(clvJornada),
    clvEqLocal      VARCHAR(20)     REFERENCES Equipo(clvEquipo),
    clvEqVisit      VARCHAR(20)     REFERENCES Equipo(clvEquipo),
    CHECK (clvEqLocal <> clvEqVisit),
    UNIQUE (clvJornada, clvEqLocal),
    UNIQUE (clvJornada, clvEqVisit)
);

CREATE TABLE Compiten (
    clvJornada      NUMBER(6)       REFERENCES Jornada(clvJornada),
    clvEquipo       VARCHAR(20)     REFERENCES Equipo(clvEquipo),
    puntuacion      NUMBER(3)       NOT NULL,
    puesto          NUMBER(2)       NOT NULL,
    PRIMARY KEY (clvJornada, clvEquipo)
);
```

Parte 2:

Pasos seguidos para poblar la Base de Datos.

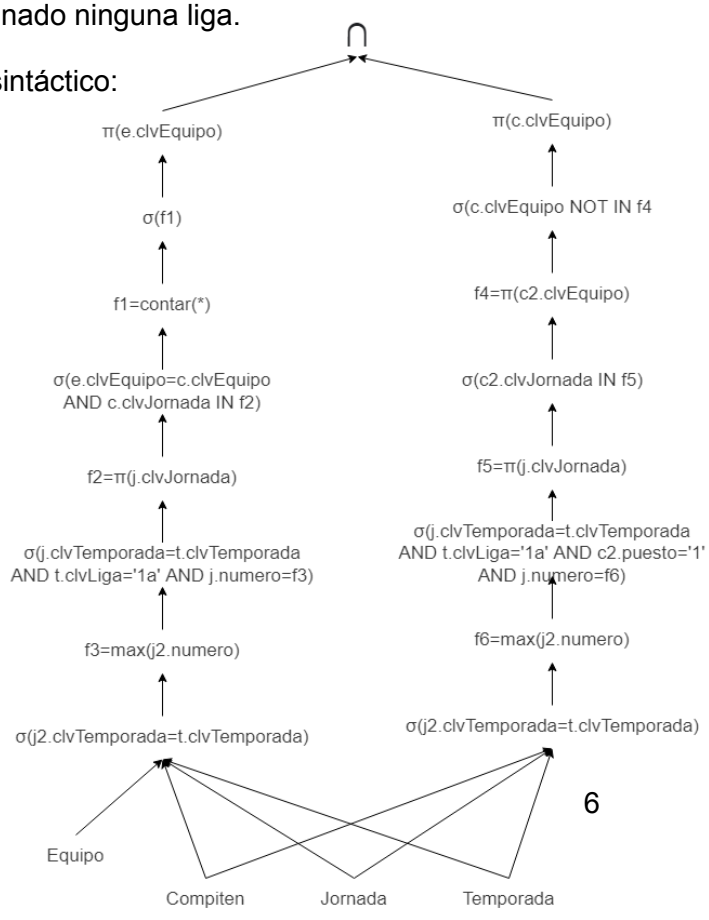
Inicialmente, se accedió a las páginas ofrecidas para extraer los datos relativos a la práctica. Se utilizó una herramienta de Excel para extraer dichos datos de las tablas de Wikipedia y poder trabajar con ellos desde un programa adecuado. Para poder tratarlas entre todos los miembros del grupo, estas fueron subidas a Drive y se trabajó con ellas a través de hojas de cálculo compartidas. Al completar todas las tablas de Wikipedia con un formato adecuado, resultó que había datos que faltaban. Por ejemplo, había partidos jugados entre equipos que no se encontraban registrados en la tabla extraída de Wikipedia. Por ello, se realizó una ardua búsqueda por todas las tablas, incorporando a cada una los datos faltantes. Para rellenar los “Nombres cortos” de “Equipo”, se creó un programa en C++ para extraer dichas claves de “LigaHost” en fichero de texto y se incorporaron a mano durante la corrección de falta de datos. Una vez terminadas las tablas de “Equipos”, “Estadios” y “Apodos”, se creó el resto de tablas a través de funciones proporcionadas por Google Sheets, el uso de búsquedas mediante expresiones regulares que ofrecen editores de texto como Visual Studio Code, y mediante programas de C++ para tareas más complejas. Posteriormente, se añadieron las respectivas claves identificadoras artificiales en aquellas tablas que lo necesitasen.

Las tablas de C++ fueron creadas directamente en fichero .csv mientras que las que se encontraban en hojas de cálculo tuvieron que exportarse como fichero .csv. A la hora de subir todos los ficheros a la base de datos para poblarla surgieron problemas con algunos caracteres especiales como pueden ser tildes, la ñ o la ç. Una vez corregidos estos caracteres se procedió a cargar los datos de manera satisfactoria en la base de datos. La población se llevó a cabo a través del programa de ORACLE SQL *Loader.

Consultas SQL.

Consulta 1: Equipo(s) que han estado en primera división un mínimo de cinco temporadas y que no han ganado ninguna liga.

Árbol sintáctico:



Sentencia:

```
-- Selecciona equipos que aparecen en primera división 5 veces o más
SELECT DISTINCT e.clvEquipo FROM Equipo e
WHERE '5'<=(
  -- Obtiene equipos que aparezcan en la tabla final de la temporada
  SELECT count(*) FROM Compiten c
  WHERE e.clvEquipo=c.clvEquipo AND c.clvJornada IN (
    SELECT DISTINCT j.clvJornada FROM Temporada t, Jornada j
    -- Temporadas identificadas como de 1a división
    WHERE j.clvTemporada=t.clvTemporada AND t.clvLiga='1a' AND j.numero=(
      -- Selecciona la última jornada de cada temporada
      SELECT max(j2.numero) FROM Jornada j2
      WHERE j2.clvTemporada=t.clvTemporada)))
INTERSECT
-- Selecciona equipos han participado en 1a división y no ganaron
SELECT DISTINCT c.clvEquipo FROM Compiten c
-- Invierte selección
WHERE c.clvEquipo NOT IN (
  -- Equipos que han ganado 1a división
  SELECT DISTINCT c2.clvEquipo FROM Compiten c2
  WHERE c2.clvJornada IN (
    SELECT DISTINCT j.clvJornada FROM Temporada t, Jornada j
    -- Temporadas identificadas como de 1a división
    WHERE j.clvTemporada=t.clvTemporada AND t.clvLiga='1a'
    -- Donde el equipo seleccionado queda primero
    AND c2.puesto='1' AND j.numero=(
      -- En la ultima jornada de la temporada
      SELECT max(j2.numero) FROM Jornada j2
      WHERE j2.clvTemporada=t.clvTemporada)));
```

Resultado:

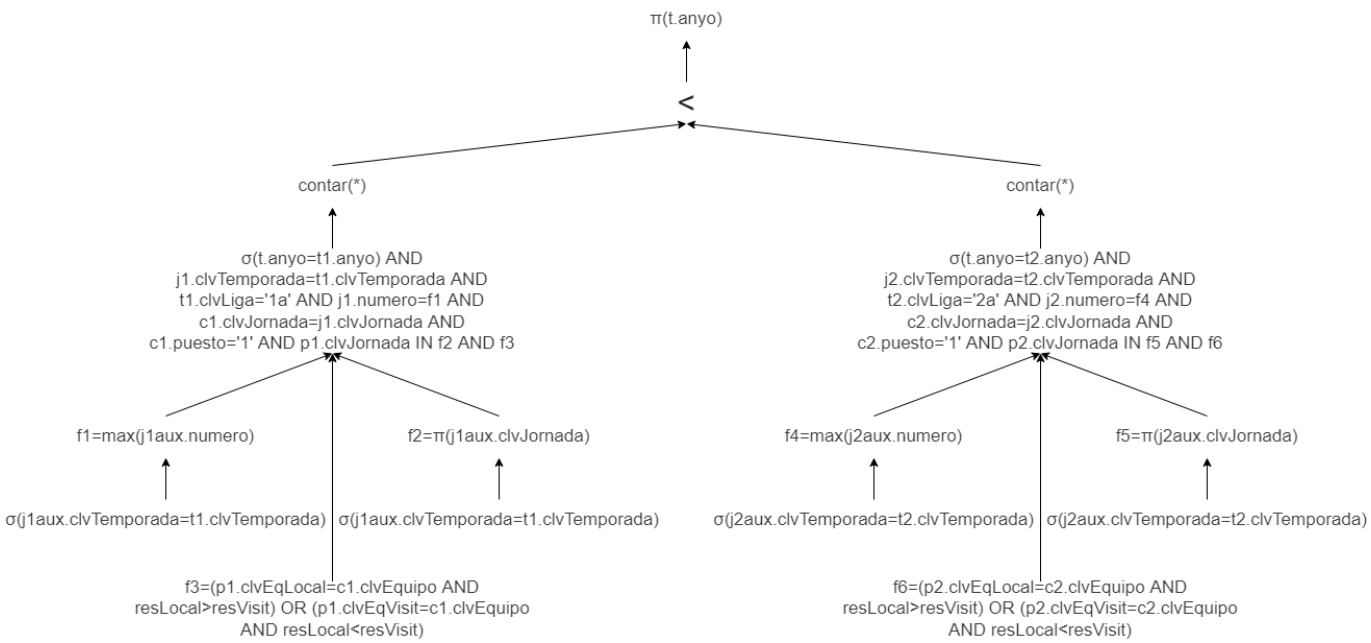
```
CLVEQUIPO
--------------------
Alaves
Albacete
Almeria
Betis
Burgos
Cadiz
Castellon
Celta
Elche
Espanyol
Getafe
Granada
Hercules
Las Palmas
```

Levante
Logrones
Malaga
Malaga (C.D.)
Mallorca
Murcia
Osasuna
Oviedo
Rac. Santander
Rayo Vallecano
Rec. Huelva
Salamanca
Sevilla
Sporting Gijón
Tenerife
Valladolid
Villarreal
Zaragoza

32 filas seleccionadas.

Consulta 2: Temporada(s) en las que el ganador de segunda división ha ganado más partidos que el ganador de primera división.

Árbol sintáctico:



Sentencia:

```
SELECT DISTINCT t.anyo FROM Temporada t
WHERE
-- Cuenta victorias de ganador de temporada de 1a división
(SELECT count(*) FROM Partido p1, Temporada t1, Jornada j1, Compiten c1
-- Comprueba que son de la misma temporada
WHERE t.anyo=t1.anyo
-- Última jornada de 1a división
AND j1.clvTemporada=t1.clvTemporada AND t1.clvLiga='1a' AND j1.numero=
  (SELECT max(j1aux.numero) FROM Jornada j1aux
  WHERE j1aux.clvTemporada=t1.clvTemporada)
-- Primer puesto
AND c1.clvJornada=j1.clvJornada AND c1.puesto='1' AND p1.clvJornada IN
  (SELECT j1aux.clvJornada FROM Jornada j1aux
  WHERE j1aux.clvTemporada=t1.clvTemporada)
-- Partidos ganados
AND ((p1.clvEqLocal=c1.clvEquipo AND resLocal>resVisit) OR
(p1.clvEqVisit=c1.clvEquipo AND resLocal<resVisit)))
< -- MENOR QUE
-- Cuenta victorias de ganador de temporada de 2a división
(SELECT count(*) FROM Partido p2, Temporada t2, Jornada j2, Compiten c2
-- Misma temporada
WHERE t.anyo=t2.anyo
-- Última jornada
AND j2.clvTemporada=t2.clvTemporada AND t2.clvLiga='2a' AND j2.numero=
  (SELECT max(j2aux.numero) FROM Jornada j2aux
  WHERE j2aux.clvTemporada=t2.clvTemporada)
-- Primer puesto
AND c2.clvJornada=j2.clvJornada AND c2.puesto='1' AND p2.clvJornada IN
  (SELECT j2aux.clvJornada FROM Jornada j2aux
  WHERE j2aux.clvTemporada=t2.clvTemporada)
-- Partidos gandaos
AND ((p2.clvEqLocal=c2.clvEquipo AND resLocal>resVisit) OR
(p2.clvEqVisit=c2.clvEquipo AND resLocal<resVisit)));
```

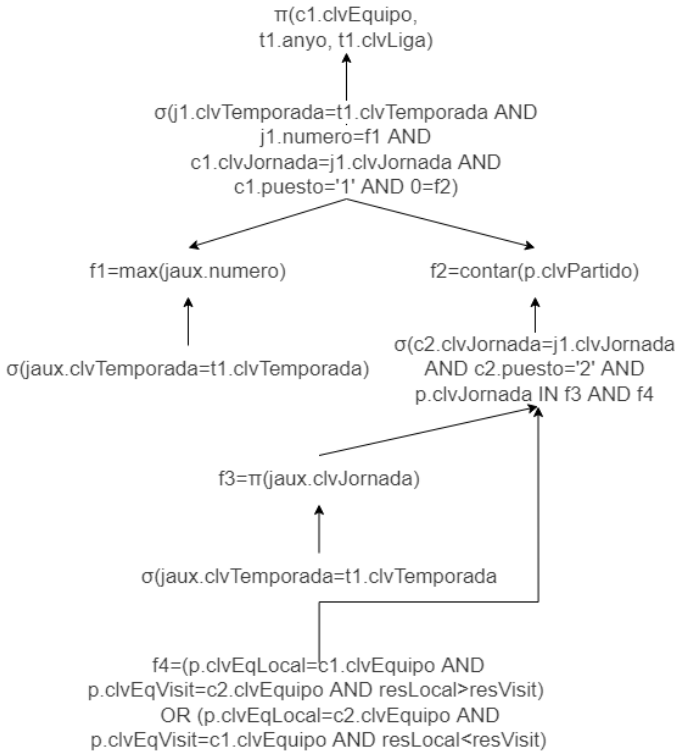
Resultado:

```
ANYO
---------
1997-1998
1974-1975
1978-1979
1981-1982
1984-1985
2001-2002
2006-2007
1972-1973
2002-2003

9 filas seleccionadas.
```

Consulta 3: Equipo(s) y temporada(s) donde dicho equipo ha ganado dicha temporada, habiendo perdido todos los partidos contra el equipo que quedó en segunda posición.

Árbol sintáctico:



Sentencia:

```
SELECT DISTINCT c1.clvEquipo, t1.anyo, t1.clvLiga FROM Temporada t1, Jornada j1,
Compiten c1
-- Última jornada de temporada
WHERE j1.clvTemporada=t1.clvTemporada AND j1.numero=
  (SELECT DISTINCT max(jaux.numero) FROM Jornada jaux
  WHERE jaux.clvTemporada=t1.clvTemporada)
-- Primero de la temporada
AND c1.clvJornada=j1.clvJornada AND c1.puesto='1' AND 0=
-- Selecciona sólo si no ha ganado ningún partido contra el segundo
  (SELECT count(p.clvPartido) FROM Jornada j2, Compiten c2, Partido p
-- Segundo de la temporada
  WHERE c2.clvJornada=j1.clvJornada AND c2.puesto='2' AND p.clvJornada IN
-- Dentro de la temporada...
    (SELECT DISTINCT jaux.clvJornada FROM Jornada jaux
    WHERE jaux.clvTemporada=t1.clvTemporada)
-- ...partidos que haya ganado el primero de liga al final contra el segundo
  AND ((p.clvEqLocal=c1.clvEquipo AND p.clvEqVisit=c2.clvEquipo AND
resLocal>resVisit)
    OR (p.clvEqLocal=c2.clvEquipo AND p.clvEqVisit=c1.clvEquipo AND
resLocal<resVisit)));
```

Resultado:

CLVEQUIPO	ANYO	CLVLIGA
Osasuna	1986-1987	Descenso a 2a
Compostela	1997-1998	Promocion 1a/2a
Malaga	1998-1999	2a
Levante	2003-2004	2a
Real Sociedad	1981-1982	1a
Tenerife	1989-1990	Promocion 1a/2a
Cadiz	1991-1992	Promocion 1a/2a
Alaves	1997-1998	2a
Real Madrid	2002-2003	1a
Murcia	1987-1988	Promocion 1a/2a
Zaragoza	1990-1991	Promocion 1a/2a
Las Palmas	2013-2014	Promocion 1a/2a
Ath. Bilbao B	1983-1984	2a
Elche	2010-2011	Promocion 1a/2a
Ath. Bilbao	1982-1983	1a
Murcia	1982-1983	2a
Extremadura	1995-1996	Promocion 1a/2a
Rayo Vallecano	1998-1999	Promocion 1a/2a
Barcelona	2006-2007	1a
Real Madrid	1979-1980	1a
Las Palmas	1999-2000	2a
Murcia	1979-1980	2a
Real Madrid	1986-1987	1a
Betis	1988-1989	Promocion 1a/2a
Sporting Gijon	1994-1995	Promocion 1a/2a
Real Madrid	1983-1984	1a
Cartagena F.C.	1986-1987	2a
Rac. Santander	1992-1993	Promocion 1a/2a
Real Madrid	1975-1976	1a
Barcelona	2012-2013	1a
At. Madrid	2013-2014	1a
Valladolid	2006-2007	2a

35 filas seleccionadas.

Parte 3:

Definición de triggers.

Para verificar la integridad de la base de datos al añadir nuevas tuplas hay que asegurar que sólo se proceda con las inserciones correctas. Para tal fin se han descrito dos triggers que tratan de solucionar algunos de estas comprobaciones.

Trigger 1: No se deben introducir dos partidos con los mismos equipos local y visitante en la misma temporada.

```
CREATE OR REPLACE TRIGGER trigger1
BEFORE INSERT ON Partido
FOR EACH ROW
DECLARE
  n NUMBER;
BEGIN
  -- Obtiene número de partidos jugados en la temporada...
  SELECT COUNT(p.clvPartido) INTO n FROM Partido p, Jornada j
    WHERE p.clvJornada IN
      (SELECT jaux.clvJornada FROM Jornada jaux
      WHERE jaux.clvTemporada=j.clvTemporada AND j.clvJornada=:NEW.clvJornada)
      -- ...tal que los equipos local y visitante sean iguales
    AND p.clvEqLocal=:NEW.clvEqLocal AND p.clvEqVisit=:NEW.clvEqVisit;
  IF n>0 THEN
    RAISE_APPLICATION_ERROR(-20001,'Ya existe ese partido en esa temporada');
  END IF;
END trigger1;
/
```

Trigger 2: Un equipo no puede participar en dos divisiones diferentes en la misma temporada

```
CREATE OR REPLACE TRIGGER trigger2
BEFORE INSERT ON Partido
FOR EACH ROW
DECLARE
  n NUMBER;
  m NUMBER;
BEGIN
  -- Cuenta divisiones en las que participa el equipo local aparte de la que se va
a insertar
  SELECT COUNT(DISTINCT t.clvLiga) INTO n FROM Partido p, Jornada j, Temporada t
    WHERE j.clvJornada=:NEW.clvJornada AND t.clvTemporada=j.clvTemporada AND
p.clvJornada IN
      (SELECT jaux.clvJornada FROM Jornada jaux, Temporada taux
      -- Mismo año, liga diferente
      WHERE jaux.clvTemporada=taux.clvTemporada AND taux.anyo=t.anyo AND
taux.clvLiga<>t.clvLiga)
    -- el equipo nuevo local debe ser el mismo que cualquiera de los equipos local
o visitante de otras temporadas
    AND (p.clvEqLocal=:NEW.clvEqLocal OR p.clvEqVisit=:NEW.clvEqLocal);
```

```
  -- Cuenta divisiones en las que participa el equipo local aparte de la que se va
a insertar
  SELECT COUNT(DISTINCT t.clvLiga) INTO m FROM Partido p, Jornada j, Temporada t
    WHERE j.clvJornada=:NEW.clvJornada AND t.clvTemporada=j.clvTemporada AND
p.clvJornada IN
      (SELECT jaux.clvJornada FROM Jornada jaux, Temporada taux
      -- Mismo año, liga diferente
      WHERE jaux.clvTemporada=taux.clvTemporada AND taux.anyo=t.anyo AND
taux.clvLiga<>t.clvLiga)
    -- el equipo nuevo visitante debe ser el mismo que cualquiera de los equipos
local o visitante de otras temporadas
    AND (p.clvEqLocal=:NEW.clvEqVisit OR p.clvEqVisit=:NEW.clvEqVisit);
  IF n>0 OR m>0 THEN
    RAISE_APPLICATION_ERROR(-20002,'Ese partido ya esta en otra division esta
temporada');
  END IF;
END trigger2;
/
```

ANEXO:

Distribución de horas.

	Cristian Selivanov	Álvaro Seral	Dorian Wozniak
Esquema E/R y restricciones.	4	4	4
Esquema relacional y normalización.	2	2	2
Recopilación de información.	10	10	11
Sentencias SQL de creación de tablas.	1	1	6
Consultas SQL.	9	11	4
Definición de triggers.	4	6	2
Documentación	3	2	1
Total	33 horas	36 horas	30 horas

Problemas presentados.

- Mala organización inicial, causa por la cual no ha dado tiempo a terminar de analizar la etapa de diseño físico.
- Mala gestión del tiempo, centrando el mayor bloque de trabajo en los días de antes de la entrega de la práctica.
- Dificultades al realizar las consultas, falta de entendimiento general del lenguaje sql a la hora de ponerlo en práctica en una base de datos real por primera vez.
- Falta de ideas para sacar triggers. A la hora de realizar el tercer trigger, no se han encontrado restricciones más allá de la propia inserción de los datos.