

# BASES DE DATOS SOBRE CARRERAS



# CONTENIDO

1

Especificación

2

Diseño Conceptual  
de Datos

3

Diseño lógico de  
datos

4

Diseño físico de  
datos

5

Inserción de  
datos

6

Modificaciones y  
borrados

7

Consultas

8

Creación de  
índice

9

Creación de  
Vista

# ESPECIFICACIÓN

El cliente solicita un sistema que permita almacenar la información de los pilotos, los grandes premios, los circuitos y los equipos, contemplando los siguientes requisitos:

- **De los pilotos se desea almacenar:** nombre, apellidos, edad, su número (que no se utiliza como identificador principal), su número de victorias (si las tiene), el equipo en el que compite, el número de campeonatos del mundo (si los tiene), su nacionalidad, los grandes premios en los que corre, los puntos obtenidos en los correspondientes grandes premios (si los ha conseguido), la posición en dichos grandes premios y los patrocinadores; un piloto puede tener varios patrocinadores y viceversa.
- **De las carreras se desea almacenar:** su nombre, el número en el calendario (orden de la temporada), la fecha de la clasificación, la temporada a la que pertenece, la fecha de los entrenamientos libres, la fecha del GP, el circuito donde se celebra, el tipo de carrera y un piloto puede disputar muchos grandes premios o no haber participado aún, y cada gran premio puede tener múltiples pilotos. La meteorología y las incidencias, las incidencias no tienen sentido si no hay carreras.





- **De los Grandes Premios:** de los grandes premios se desea saber el número de GP en la temporada, fecha de FP1, código único, fecha de Qualy, nombre de Gran Premio, fecha y temporada.
- **De los circuitos se desea almacenar:** el nombre del circuito, su longitud en kilómetros y el país donde se encuentra. Hay 3 tipos de circuitos: Callejero que almacena calles y tiempo de montaje, permanentes de los que se almacena las especificaciones, híbrido del que se guardan las zonas y el porcentaje de callejero.
- **De los equipos se desea conocer:** el nombre, si son satélites de algún otro equipo (indicando de cuál) y el motor que utilizan (por ejemplo, Mercedes o Ferrari), teniendo en cuenta que un mismo motor puede ser usado por varios equipos y que un equipo solo puede montar un motor que tiene un fabricante, las ruedas que usa y el fabricante que las hace.

- Un fabricante de motores fabrica varios motores, pero cada motor tiene un solo fabricante.
- Un fabricante de neumáticos fabrica varios neumáticos, pero cada neumático tiene un solo fabricante.
- Un equipo puede ser satélite de otro y tener varios equipos satélites.
- Un director está asignado a un solo equipo o a ninguno, pero un equipo tiene un solo director.
- Un coche tiene un solo motor, pero un motor puede estar en varios coches.
- Un coche usa un solo tipo de neumático, pero un tipo de neumático puede estar en varios coches.
- Un coche pertenece a un equipo, y un equipo tiene varios coches.
- Un piloto está en un solo equipo, pero un equipo tiene varios pilotos.
- Un piloto tiene una nacionalidad, pero una nacionalidad puede tener varios pilotos.
- Un patrocinador puede patrocinar a varios pilotos y un piloto puede tener varios patrocinadores.
- Una ciudad pertenece a un solo país, pero un país tiene varias ciudades.
- Un circuito está en una ciudad, pero una ciudad puede tener varios circuitos.
- Un circuito puede ser permanente, híbrido o callejero, pero cada tipo se refiere solo a un circuito.



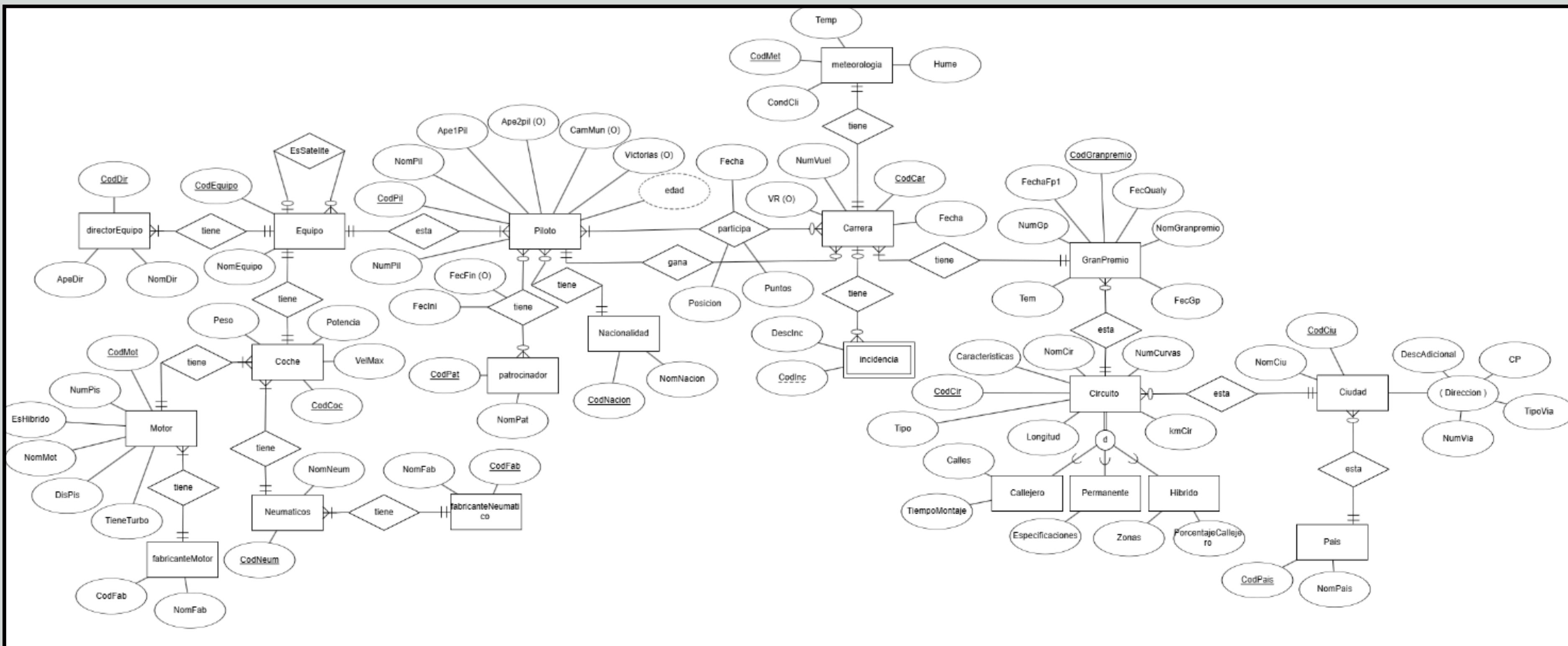


- Un gran premio se corre en un circuito, pero un circuito puede haber albergado varios grandes premios.
- Una carrera puede tener un ganador o no, y un piloto puede haber ganado varias carreras.
- Una carrera tiene una sola meteorología, pero una condición meteorológica puede repetirse en varias carreras.
- Una carrera pertenece a un gran premio, pero un gran premio puede tener varias carreras.
- Una incidencia ocurre en una sola carrera, pero una carrera puede tener varias incidencias.
- Un piloto participa en una carrera, pero puede haber participado en varias.
- Una participación está ligada a una sola carrera, pero una carrera tiene varios participantes.

# DISEÑO CONCEPTUAL DE DATOS

## MODELO ENTIDAD-RELACIÓN

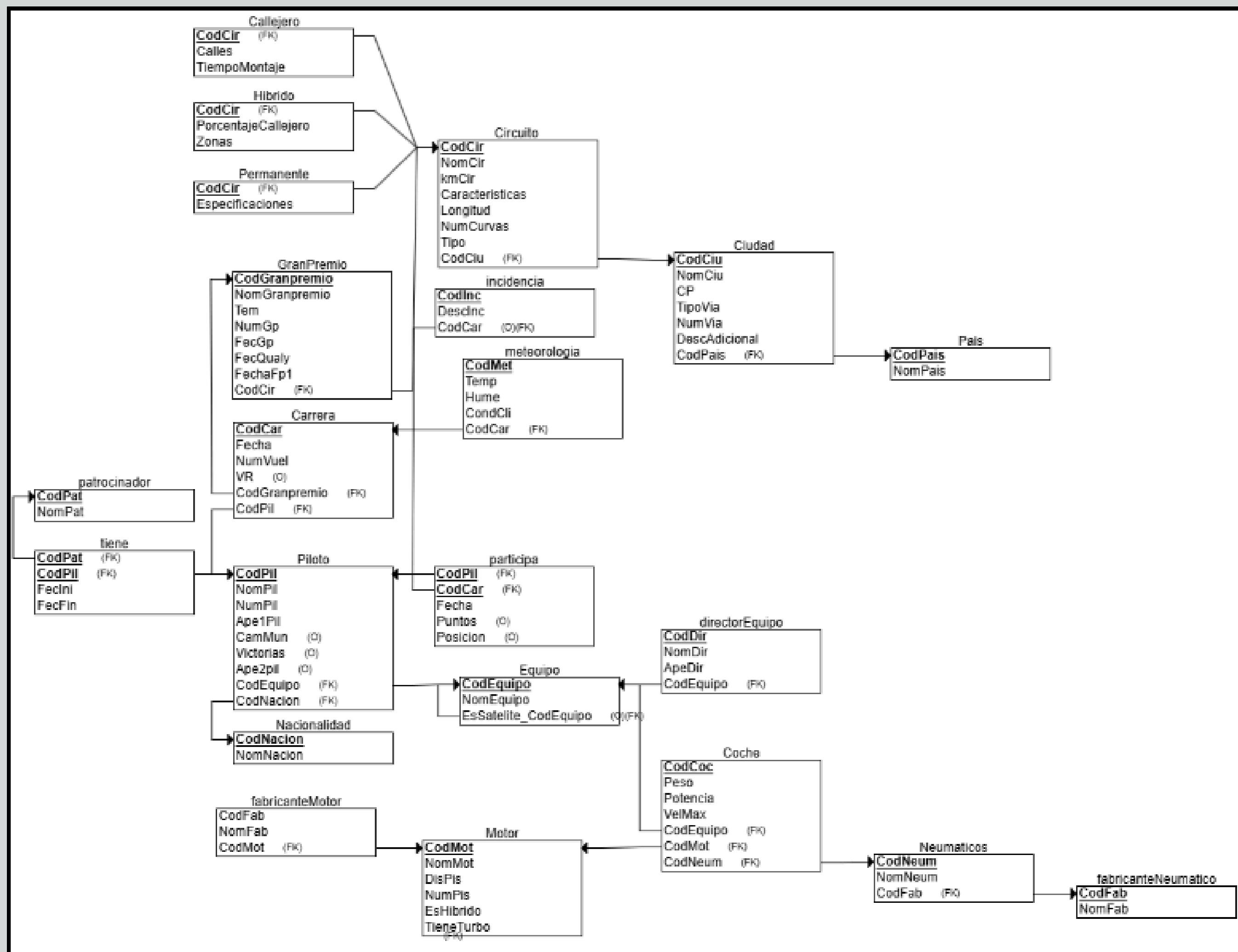




# DISEÑO LOGICO DE DATOS

## DIAGRAMA REFERENCIAL

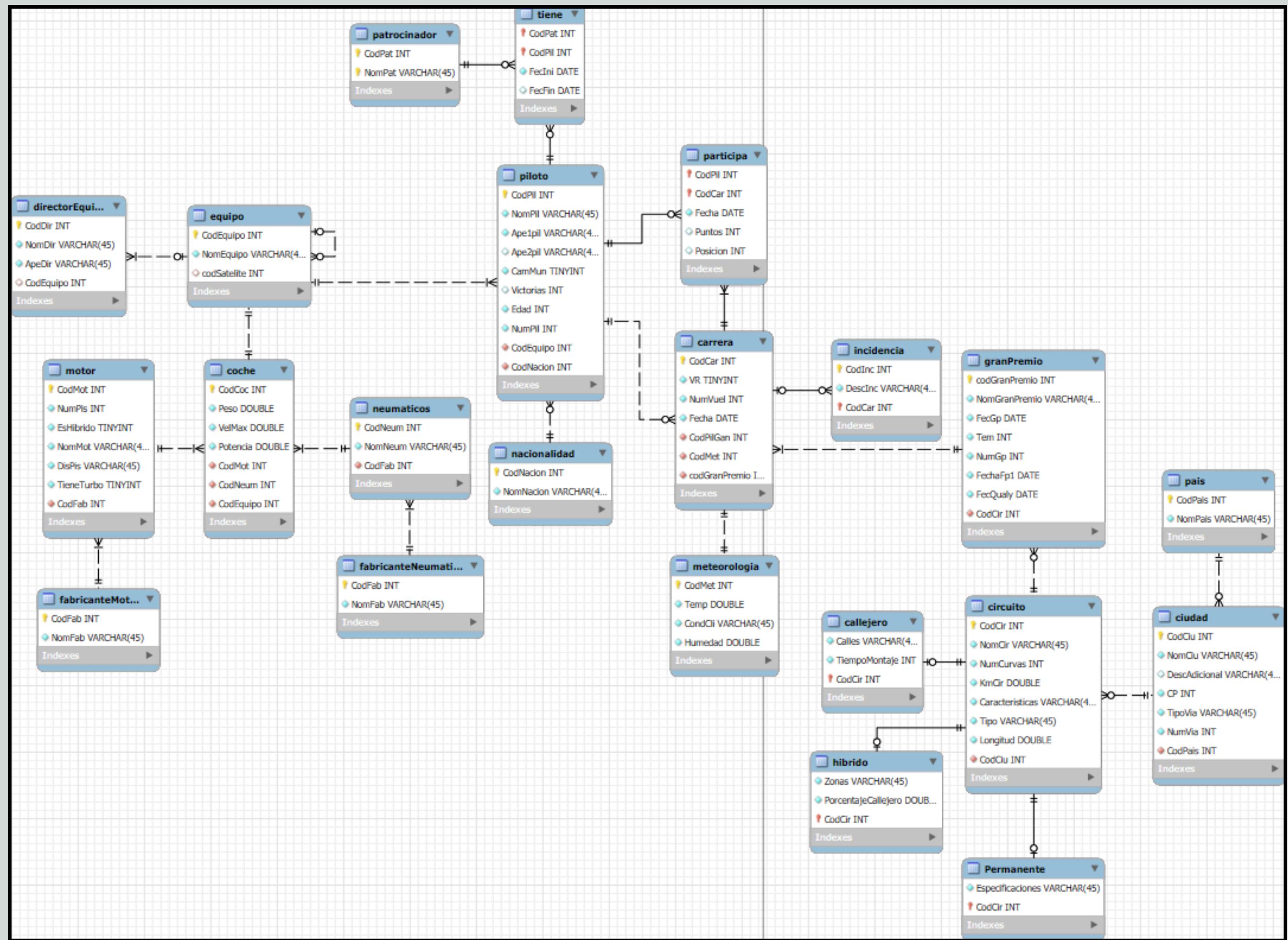






DISEÑO LOGICO DE DATOS

DIAGRAMA DE ESTRUCTURA DE  
DATOS



# DISEÑO FÍSICO DE DATOS



```
CREATE DATABASE TRABAJO
GO
USE TRABAJO
GO
CREATE TABLE fabricanteMotor
(
    CodFab INT PRIMARY KEY NOT NULL,
    NomFab VARCHAR(45) UNIQUE NOT NULL
);
CREATE TABLE motor
(
    CodMot INT PRIMARY KEY NOT NULL,
    NumPis INT NOT NULL CHECK (NumPis>0),
    EsHibrido TINYINT NOT NULL CHECK (EsHibrido IN(0,1)),
    NomMot VARCHAR(45) NOT NULL,
    DisPis VARCHAR(45) NOT NULL,
    TieneTurbo TINYINT NOT NULL CHECK (TieneTurbo IN(0,1)),
    CodFab INT NOT NULL,
    FOREIGN KEY (CodFab) REFERENCES fabricanteMotor(CodFab)
        ON DELETE NO ACTION ON UPDATE CASCADE
);
```

```
CREATE TABLE fabricanteNeumatico
(
    CodFab INT PRIMARY KEY NOT NULL,
    NomFab VARCHAR(45) UNIQUE NOT NULL
);
CREATE TABLE neumatico
(
    CodNeum INT PRIMARY KEY NOT NULL,
    NomNeum VARCHAR(45) NOT NULL,
    CodFab INT NOT NULL,
    FOREIGN KEY (CodFab) REFERENCES fabricanteNeumatico(CodFab)
        ON DELETE NO ACTION ON UPDATE CASCADE
);
CREATE TABLE equipo
(
    CodEquipo INT PRIMARY KEY NOT NULL,
    NomEquipo VARCHAR(45) NOT NULL,
    CodSatelite INT NULL,
    FOREIGN KEY (CodSatelite) REFERENCES equipo(CodEquipo)
        ON DELETE NO ACTION ON UPDATE NO ACTION
);
```

```
CREATE TABLE directorEquipo
(
    CodDir INT PRIMARY KEY NOT NULL,
    NomDir VARCHAR(45) NOT NULL,
    ApeDir VARCHAR(45) NOT NULL,
    CodEquipo INT NULL,
    FOREIGN KEY (CodEquipo) REFERENCES equipo(CodEquipo)
        ON DELETE SET NULL ON UPDATE CASCADE
);
CREATE TABLE coche
(
    CodCoc INT PRIMARY KEY NOT NULL,
    Peso FLOAT NOT NULL CHECK (Peso>0),
    VelMax FLOAT NOT NULL CHECK (VelMax>0),
    Potencia FLOAT NOT NULL CHECK (Potencia>0),
    CodMot INT NOT NULL,
    CodNeum INT NOT NULL,
    CodEquipo INT NOT NULL,
    FOREIGN KEY (CodMot) REFERENCES motor(CodMot)
        ON DELETE NO ACTION ON UPDATE CASCADE,
    FOREIGN KEY (CodNeum) REFERENCES neumatico(CodNeum)
        ON DELETE NO ACTION ON UPDATE CASCADE,
    FOREIGN KEY (CodEquipo) REFERENCES equipo(CodEquipo)
        ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
CREATE TABLE nacionalidad
(
    CodNacion INT PRIMARY KEY NOT NULL,
    NomNacion VARCHAR(45) UNIQUE NOT NULL
);
CREATE TABLE piloto
(
    CodPil INT PRIMARY KEY IDENTITY(1,1) NOT NULL,
    NomPil VARCHAR(45) NOT NULL,
    Ape1pil VARCHAR(45) NOT NULL,
    Ape2pil VARCHAR(45) NULL,
    CamMun TINYINT NOT NULL CHECK (CamMun IN(0,1)),
    Victorias INT NULL CHECK (Victorias>=0),
    Edad INT NOT NULL CHECK (Edad>=18),
    NumPil INT NOT NULL,
    CodEquipo INT NULL,
    CodNacion INT NOT NULL,
    FOREIGN KEY (CodEquipo) REFERENCES equipo(CodEquipo)
        ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (CodNacion) REFERENCES nacionalidad(CodNacion)
        ON DELETE NO ACTION ON UPDATE CASCADE
);
```

```
CREATE TABLE patrocinador
(
    CodPat INT PRIMARY KEY NOT NULL,
    NomPat VARCHAR(45) UNIQUE NOT NULL
);
CREATE TABLE tiene
(
    CodPat INT NOT NULL,
    CodPil INT NOT NULL,
    FecIni DATE NOT NULL DEFAULT CONVERT(DATE, GETDATE()),
    FecFin DATE NULL,
    PRIMARY KEY(CodPat,CodPil),
    FOREIGN KEY (CodPat) REFERENCES patrocinador(CodPat)
        ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (CodPil) REFERENCES piloto(CodPil)
        ON DELETE CASCADE ON UPDATE CASCADE
);
CREATE TABLE meteorologia
(
    CodMet INT PRIMARY KEY NOT NULL,
    Temp FLOAT NOT NULL CHECK (Temp BETWEEN -50 AND 60),
    CondCli VARCHAR(45) NOT NULL CHECK (CondCli IN ('soleado','nublado','lluvia','tormenta','nevado')),
    Humedad FLOAT NOT NULL CHECK (Humedad BETWEEN 0 AND 100)
);
```

```
CREATE DATABASE TRABAJO
GO
USE TRABAJO
GO
CREATE TABLE fabricanteMotor
(
    CodFab INT PRIMARY KEY NOT NULL,
    NomFab VARCHAR(45) UNIQUE NOT NULL
);
CREATE TABLE motor
(
    CodMot INT PRIMARY KEY NOT NULL,
    NumPis INT NOT NULL CHECK (NumPis>0),
    EsHibrido TINYINT NOT NULL CHECK (EsHibrido IN(0,1)),
    NomMot VARCHAR(45) NOT NULL,
    DisPis VARCHAR(45) NOT NULL,
    TieneTurbo TINYINT NOT NULL CHECK (TieneTurbo IN(0,1)),
    CodFab INT NOT NULL,
    FOREIGN KEY (CodFab) REFERENCES fabricanteMotor(CodFab)
        ON DELETE NO ACTION ON UPDATE CASCADE
);
```

```
CREATE TABLE pais
(
    CodPais INT PRIMARY KEY NOT NULL,
    NomPais VARCHAR(45) UNIQUE NOT NULL
);
CREATE TABLE ciudad
(
    CodCiu INT PRIMARY KEY NOT NULL,
    NomCiu VARCHAR(45) NOT NULL,
    DescAdicional VARCHAR(45) NULL,
    CP INT NOT NULL CHECK (CP>0),
    TipoVia VARCHAR(45) NOT NULL CHECK (TipoVia IN ('calle','avenida','paseo','carretera','camino','plaza')),
    NumVia INT NOT NULL CHECK (NumVia>0),
    CodPais INT NOT NULL,
    FOREIGN KEY (CodPais) REFERENCES pais(CodPais)
        ON DELETE NO ACTION ON UPDATE CASCADE
);
CREATE TABLE circuito
(
    CodCir INT PRIMARY KEY NOT NULL,
    NomCir VARCHAR(45) UNIQUE NOT NULL,
    KmCir FLOAT NOT NULL CHECK (KmCir>0),
    Caracteristicas VARCHAR(45) NOT NULL,
    Tipo VARCHAR(45) NOT NULL CHECK (Tipo IN ('permanente','hibrido','callejero')),
    Longitud FLOAT NOT NULL CHECK (Longitud>0),
    CodCiu INT NOT NULL,
    FOREIGN KEY (CodCiu) REFERENCES ciudad(CodCiu)
        ON DELETE NO ACTION ON UPDATE CASCADE
);
```

```
CREATE TABLE permanente
(
    Esoecificaciones VARCHAR(45) NOT NULL,
    CodCir INT NOT NULL,
    PRIMARY KEY (CodCir),
    FOREIGN KEY (CodCir) REFERENCES circuito(CodCir)
        ON DELETE CASCADE ON UPDATE CASCADE
);
CREATE TABLE hibrido
(
    Zonas VARCHAR(45) NOT NULL,
    PorcentajeCallejero FLOAT NOT NULL CHECK (PorcentajeCallejero BETWEEN 0 AND 100),
    CodCir INT NOT NULL,
    PRIMARY KEY (CodCir),
    FOREIGN KEY (CodCir) REFERENCES circuito(CodCir)
        ON DELETE CASCADE ON UPDATE CASCADE
);
CREATE TABLE callejero
(
    Calles VARCHAR(45) NOT NULL,
    TiempoMontaje INT NOT NULL CHECK (TiempoMontaje>0),
    CodCir INT NOT NULL,
    PRIMARY KEY(CodCir),
    FOREIGN KEY (CodCir) REFERENCES circuito(CodCir)
        ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
CREATE TABLE granPremio
(
    CodGranPremio INT PRIMARY KEY NOT NULL,
    NomGranPremio VARCHAR(45) UNIQUE NOT NULL,
    FecGp DATE NOT NULL,
    Tem INT NOT NULL DEFAULT YEAR(GETDATE()),
    NumGp INT NOT NULL CHECK (NumGp>0),
    FechaFp1 DATE NOT NULL,
    FechaQualy DATE NULL,
    CodCir INT NOT NULL,
    FOREIGN KEY (CodCir) REFERENCES circuito(CodCir)
        ON DELETE NO ACTION ON UPDATE CASCADE
);
CREATE TABLE carrera
(
    CodCar INT PRIMARY KEY NOT NULL,
    VR TINYINT NOT NULL CHECK (VR IN(0,1)),
    NumVuel INT NOT NULL CHECK (NumVuel>0),
    Fecha DATE NOT NULL,
    CodPilGan INT NULL,
    CodMet INT NOT NULL,
    CodGranPremio INT NOT NULL,
    FOREIGN KEY (CodPilGan) REFERENCES piloto(CodPil)
        ON DELETE SET NULL ON UPDATE CASCADE,
    FOREIGN KEY (CodMet) REFERENCES meteorologia(CodMet)
        ON DELETE NO ACTION ON UPDATE CASCADE,
    FOREIGN KEY (CodGranPremio) REFERENCES granPremio(CodGranPremio)
        ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
CREATE TABLE incidencia
(
    CodInc INT NOT NULL,
    DescInc VARCHAR(45) NOT NULL,
    CodCar INT NOT NULL,
    PRIMARY KEY(CodInc, CodCar),
    FOREIGN KEY (CodCar) REFERENCES carrera(CodCar)
        ON DELETE CASCADE ON UPDATE CASCADE
);
CREATE TABLE participa
(
    CodPil INT NOT NULL,
    CodCar INT NOT NULL,
    Fecha DATE NOT NULL DEFAULT CONVERT(DATE, GETDATE()),
    Puntos INT NULL CHECK (Puntos>=0),
    Posicion INT NULL CHECK (Posicion>0),
    PRIMARY KEY(CodPil,CodCar),
    FOREIGN KEY (CodPil) REFERENCES piloto(CodPil)
        ON DELETE NO ACTION ON UPDATE NO ACTION,
    FOREIGN KEY (CodCar) REFERENCES carrera(CodCar)
        ON DELETE NO ACTION ON UPDATE NO ACTION
);
```



# INSERCIÓN DE DATOS

```
INSERT INTO fabricanteMotor
VALUES (1, 'MotoX'),
       (2, 'TurboMotors'),
       (3, 'SpeedEngines'),
       (4, 'PowerCrafters');

INSERT INTO motor
VALUES (1, 4, 0, 'MotoX Standard', 'I4', 0, 1),
       (2, 6, 1, 'TurboMotor V8', 'V8', 1, 2),
       (3, 8, 0, 'SpeedEngine Ultra', 'V8', 1, 3),
       (4, 4, 0, 'PowerCraft Compact', 'I4', 0, 4);

INSERT INTO fabricanteNeumatico
VALUES (1, 'GripTires'),
       (2, 'UltraRubber');

INSERT INTO neumatico
VALUES (1, 'GripTire Sport', 1),
       (2, 'GripTire Eco', 1),
       (3, 'UltraRubber Pro', 2),
       (4, 'UltraRubber Max', 2);
```

```
INSERT INTO equipo
VALUES (1, 'Team Alpha', NULL),
       (2, 'Team Beta', 1),
       (3, 'Team Gamma', NULL),
       (4, 'Team Delta', 3);
```

```
INSERT INTO directorEquipo
VALUES (1, 'John', 'Doe', 1),
       (2, 'Alice', 'Smith', 2),
       (3, 'Bob', 'Johnson', NULL),
       (4, 'Carmen', 'Lopez', 4);
```

```
INSERT INTO coche
VALUES (1, 1200, 220, 300, 1, 1, 1),
       (2, 1300, 240, 350, 2, 2, 2),
       (3, 1250, 230, 320, 3, 3, 3),
       (4, 1400, 210, 310, 4, 4, 4);
```

```
INSERT INTO nacionalidad
VALUES (1, 'Freedonia'),
       (2, 'Utopia'),
       (3, 'Elbonia'),
       (4, 'Genovia'),
       (5, 'Ruritania');
```

```
INSERT INTO piloto
VALUES ('Maxim', 'Power', NULL, 1, 3, 25, 10, 1, 1),
       ('Sarah', 'Speed', 'Johnson', 0, 5, 28, 11, 2, 2),
       ('Ivan', 'Turbo', NULL, 1, NULL, 30, 12, 3, 3),
       ('Lucy', 'Rapid', 'Lee', 0, 2, 22, 13, 4, 4),
       ('Oscar', 'Quick', NULL, 1, 1, 35, 14, 1, 5),
       ('Emily', 'Swift', 'Brown', 0, 4, 27, 15, 2, 1);

INSERT INTO patrocinador
VALUES (1, 'SponAlpha'),
       (2, 'SponBeta'),
       (3, 'SponGamma');

INSERT INTO tiene
VALUES (1, 1, DEFAULT, NULL),
       (2, 2, '2023-11-15', '2023-12-31'),
       (3, 1, DEFAULT, NULL),
       (1, 4, '2023-10-01', '2026-01-15'),
       (2, 5, DEFAULT, '2024-02-20'),
       (3, 6, '2023-09-20', NULL),
       (2, 1, DEFAULT, NULL);

INSERT INTO meteorologia
VALUES (1, 25.0, 'soleado', 50),
       (2, -10.0, 'nublado', 80),
       (3, 5.0, 'lluvia', 90);
```

```
INSERT INTO pais
VALUES (1, 'Fictionland'),
(2, 'Imaginaria'),
(3, 'Nowhereland'),
(4, 'Dreamworld');
```

```
INSERT INTO ciudad
VALUES (1, 'Alpha City', 'Central hub', 10001, 'calle', 1, 1),
(2, 'Beta Town', NULL, 20002, 'avenida', 2, 2),
(3, 'Gamma Village', 'Coastal', 30003, 'paseo', 3, 3),
(4, 'Delta Metropolis', 'Large urban', 40004, 'carretera', 4, 4),
(5, 'Epsilon', NULL, 50005, 'camino', 5, 1);
```

```
INSERT INTO circuito
VALUES (1, 'Circuito Uno', 3.5, 'Fast and technical', 'permanente', 3.5, 1),
(2, 'Circuito Dos', 4.2, 'Challenging corners', 'hibrido', 4.2, 2),
(3, 'Circuito Tres', 2.8, 'Short street track', 'callejero', 2.8, 3),
(4, 'Circuito Cuatro', 5.0, 'Long endurance', 'permanente', 5.0, 4),
(5, 'Circuito Cinco', 3.0, 'Technical and twisty', 'hibrido', 3.0, 5);
```

```
INSERT INTO permanente
VALUES ('Specs Uno', 1),
       ('Specs Cuatro', 4);

INSERT INTO hibrido
VALUES ('Hybrid Zone Two', 25, 2),
       ('Hybrid Zone Five', 40, 5);

INSERT INTO callejero
VALUES ('Urban Streets', 60, 3);

INSERT INTO granPremio
VALUES (1, 'Gran Premio Uno', '2024-06-15', DEFAULT, 1, '2024-06-14', '2024-06-14', 1),
       (2, 'Gran Premio Dos', '2024-07-20', 2024, 2, '2024-07-19', '2024-07-19', 2),
       (3, 'Gran Premio Tres', '2024-08-25', DEFAULT, 3, '2024-08-24', NULL, 3),
       (4, 'Gran Premio Cuatro', '2024-09-10', 2025, 4, '2024-09-09', '2024-09-09', 4);
```

```
INSERT INTO carrera
VALUES (1, 1, 55, '2024-06-15', 1, 1, 1),
       (2, 0, 60, '2024-07-20', 2, 2, 2),
       (3, 1, 50, '2024-08-25', NULL, 3, 3);

INSERT INTO incidencia
VALUES (1, 'Minor collision', 1),
       (2, 'Engine failure', 2);

INSERT INTO participa
VALUES (1, 1, DEFAULT, NULL, 1),
       (2, 1, '2024-06-15', NULL, 2),
       (3, 2, DEFAULT, NULL, 1),
       (4, 2, '2024-07-20', NULL, 3),
       (1, 3, DEFAULT, NULL, 1),
       (5, 3, '2024-08-25', NULL, 2),
       (6, 3, DEFAULT, NULL, 3),
       (4, 3, '2024-08-25', NULL, 4);
```



# MODIFICACIONES Y BORRADOS



--UPDATES Y DELETES MÁS RELEVANTES:

--Asignar puntuación en base a la posición

BEGIN TRANSACTION

SELECT \* FROM participa;

UPDATE participa

SET Puntos = CASE

WHEN Posicion = 1 THEN 25

WHEN Posicion = 2 THEN 18

WHEN Posicion = 3 THEN 15

WHEN Posicion = 4 THEN 12

WHEN Posicion = 5 THEN 10

WHEN Posicion = 6 THEN 8

WHEN Posicion = 7 THEN 6

WHEN Posicion = 8 THEN 4

WHEN Posicion = 9 THEN 2

WHEN Posicion = 10 THEN 1

ELSE 0

END;

SELECT \* FROM participa;

ROLLBACK TRANSACTION

```
--Actualizar los puntos asociados a un equipo en caso de incidencia en la carrera
BEGIN TRANSACTION
SELECT * FROM participa;

UPDATE participa
SET Puntos = Puntos+3
FROM participa P JOIN carrera C ON (P.CodCar=C.CodCar)
JOIN piloto P2 ON (P.CodPil=P2.CodPil)
JOIN equipo E ON (P2.CodEquipo=E.CodEquipo)
WHERE C.CodCar IN (SELECT CodCar
                    FROM incidencia I
                    WHERE (I.CodCar=C.CodCar));

SELECT * FROM participa;
ROLLBACK TRANSACTION
```

```
--Eliminar de la base de datos las relaciones entre patrocinadores y pilotos si el plazo de la relacion ya expiró
BEGIN TRANSACTION
SELECT * FROM tiene;

DELETE FROM tiene
WHERE FecFin IS NOT NULL AND FecFin<GETDATE();

SELECT * FROM tiene;
ROLLBACK TRANSACTION
```

```
--Eliminar pilotos de un equipo si no tienen victorias, son mayores a 28 años y no tienen patrocinadores
BEGIN TRANSACTION
SELECT * FROM piloto;
SELECT * FROM tiene;

UPDATE piloto
SET CodEquipo = NULL
WHERE Victorias IS NULL AND Edad>28 AND CodPil NOT IN (SELECT CodPil FROM tiene);

SELECT * FROM piloto;
ROLLBACK TRANSACTION
```

```
--Aumentar número de victorias en caso de victoria
BEGIN TRANSACTION
SELECT * FROM piloto;

UPDATE piloto
SET Victorias = Victorias+1
FROM piloto P JOIN participa P2 ON (P.CodPil=P2.CodPil)
WHERE Posicion=1;

SELECT * FROM piloto;
ROLLBACK TRANSACTION
```

```
--Aumentar el peso de un coche en caso de que este tenga turbo(se asume que el peso del turbo es estático)
BEGIN TRANSACTION
SELECT * FROM coche;

UPDATE coche
SET Peso = Peso*1.3
FROM coche C JOIN motor M ON (C.CodCoc=M.CodMot)
WHERE TieneTurbo=1;

SELECT * FROM coche;
ROLLBACK TRANSACTION
```

```
--Reducir el peso de un coche en caso de ser híbrido(se asume que el peso asociado es estático otra vez)
BEGIN TRANSACTION
SELECT * FROM coche;

UPDATE coche
SET Peso = Peso*0.8
FROM coche C JOIN motor M ON (C.CodCoc=M.CodMot)
WHERE EsHibrido=1;

SELECT * FROM coche;
ROLLBACK TRANSACTION
```

# CONSULTAS



```
--Seleccionar el nombre de los equipos que tengan pilotos de nacionalidad utópica, seguido del número. En caso de no haber debe mostrarse
--un 0. Se utilizarán alias para las columnas
SELECT NomEquipo AS 'Nombre equipo', ISNULL((SELECT COUNT(P.CodPil)
                                              FROM piloto P JOIN nacionalidad N ON (P.CodNacion=N.CodNacion)
                                              WHERE N.NomNacion LIKE 'Utopia' AND (P.CodEquipo=E.CodEquipo)), 0) AS 'Nº Pilotos utópicos'
FROM equipo E
ORDER BY 2 DESC;
```

```
--Seleccionar el nombre de los equipos en los que todos sus miembros tengan menos de 30 años, además de la media de edad de los mismos, la máxima y la mínima.  
--Usar alias para las columnas  
SELECT E.NomEquipo AS 'Nombre equipo', AVG(P.Edad) AS 'Media de edad', MAX(P.Edad) AS 'Edad máxima', MIN(P.Edad) AS 'Edad mínima'  
FROM equipo E JOIN piloto P ON (E.CodEquipo=P.CodEquipo)  
GROUP BY E.NomEquipo  
HAVING AVG(P.Edad)<30  
ORDER BY 'Media de edad';
```

```
--Seleccionar el nombre de aquellos equipos cuyos pilotos no tengan más de 3 victorias, así como el nombre del que más tenga
SELECT E.NomEquipo AS 'Nombre equipo', (SELECT TOP 1 P2.NomPil
                                         FROM piloto P2
                                         WHERE P2.CodEquipo = E.CodEquipo
                                         ORDER BY P2.Victorias DESC) AS 'Piloto con más victorias'
FROM equipo E
WHERE NOT EXISTS (SELECT 1
                   FROM piloto P
                   WHERE (P.CodEquipo=E.CodEquipo) AND P.Victorias>3)
ORDER BY E.NomEquipo;
```

```
--Seleccionar los nombres de los pilotos que tengan más de 2 victorias, así como sus apellidos (en caso de no tener segundo debe de aparecer un mensaje), edades y nombres de equipo
SELECT NomPil AS 'Nombre piloto', Ape1pil AS 'Primer Apellido', ISNULL(Ape2pil, 'No existe segundo apellido') AS 'Segundo apellido', Edad, NomEquipo AS 'Nombre equipo'
FROM piloto P JOIN equipo E ON (P.CodEquipo=E.CodEquipo)
WHERE EXISTS (SELECT 1
              FROM piloto P2
              WHERE P2.CodPil=P.CodPil AND P2.Victorias>2)
ORDER BY Victorias DESC;
```

```
--Seleccionar las incidencias ocurridas en carreras realizadas en 2024, mostrando su codigo junto con los pilotos involucrados. Usando alias
SELECT CodInc AS 'Codigo incidencia', NomPil AS 'Nombre piloto'
FROM incidencia I JOIN carrera C ON (I.CodCar=C.CodCar)
    JOIN participa P ON (P.CodCar=C.CodCar)
    JOIN piloto P2 ON (P.CodPil=P2.CodPil)
WHERE I.CodCar = ANY (SELECT CodCar
    FROM carrera
    WHERE YEAR(Fecha)=2024)
ORDER BY 1;
```

```
--ordena por numero de victorias los fabricantes de neumaticos
SELECT fn.NomFab, ISNULL(SUM(P.Victorias),0) AS VICTORIAS FROM fabricanteNeumatico fn
JOIN neumatico n ON fn.CodFab = n.CodFab
JOIN coche C ON C.CodNeum = n.CodNeum
JOIN equipo e ON e.CodEquipo = c.CodEquipo
JOIN piloto p ON p.CodEquipo = e.CodEquipo
GROUP BY fn.NomFab
ORDER BY SUM(p.Victorias) DESC
```

```
--ordena por numero de victorias los modelos de neumaticos
SELECT  N.NomNeum,ISNULL(SUM(P.Victorias),0) AS VICTORIAS FROM fabricanteNeumatico fn
JOIN neumatico n ON fn.CodFab = n.CodFab
JOIN coche C ON C.CodNeum = n.CodNeum
JOIN equipo e ON e.CodEquipo = c.CodEquipo
JOIN piloto p ON p.CodEquipo = e.CodEquipo
GROUP BY N.NomNeum
ORDER BY SUM(p.Victorias) DESC
```

```
-- ordena los pilotos por el numero de podios
SELECT NomPil,COUNT(CASE WHEN PA.Posicion<=3 THEN 1 ELSE NULL end) as Podios FROM piloto PIL
JOIN participa pa ON pil.CodPil =pa.CodPil
GROUP BY NomPil
ORDER BY COUNT(CASE WHEN PA.Posicion<=3 THEN 1 ELSE 0 end) DESC
```

```
--ordenar los pilotos por tener mas patrocinadores
SELECT pil.NomPil,COUNT(*) as 'NUMERO DE PATROCINADORES' FROM piloto pil
JOIN tiene ti ON ti.CodPil =pil.CodPil
JOIN patrocinador pa ON ti.CodPat = pa.CodPat
GROUP BY PIL.NomPil
ORDER BY COUNT(*) DESC
```

```
--listar los pilotos que tengan podios pero no victorias
SELECT pil.NomPil,COUNT(CASE WHEN PAR.Posicion<=3 THEN 1 ELSE NULL end) as Podios from piloto pil
JOIN participa par ON par.CodPil = pil.CodPil
GROUP BY PIL.NomPil
HAVING COUNT(CASE WHEN Posicion = 1 then 1 ELSE NULL END) = 0
ORDER BY COUNT(CASE WHEN PAR.Posicion<=3 THEN 1 ELSE NULL end);
```

```
--mostrar el nombre del motor si tiene turbo poner la velocidad maxima ,  
--si es hibrido el peso y si son las dos la potencia el campo se llamará y si no tiene nada de eso el nombre del neumatico 'PesoVelocidadPotencia'  
SELECT M.NomMot, CASE WHEN M.TieneTurbo=1  
AND M.EsHibrido=0 THEN CONVERT (VARCHAR,VelMax) when M.EsHibrido=1  
AND M.TieneTurbo=0 then CONVERT(VARCHAR,C.Peso) when M.TieneTurbo=1  
AND M.EsHibrido=1 THEN CONVERT(VARCHAR, c.VelMax) else N.NomNeum end as 'PesoVelocidadPotencia'  FROM motor M  
JOIN coche C on C.CodMot = M.CodMot  
JOIN neumatico N on N.CodNeum =C.CodNeum;
```

```
--ordenar pilotos por numero de paises visitados
SELECT PIL.NomPil,COUNT(DISTINCT paI.CodPais) as 'numero de paises visitados' FROM piloto pil
JOIN participa par ON pil.CodPil = par.CodPil
JOIN carrera CAR on car.CodCar = par.CodCar
JOIN granPremio GP on GP.CodGranPremio =car.CodGranPremio
JOIN circuito cir ON cir.CodCir=GP.CodCir
JOIN ciudad ciu ON ciu.CodCiu=cir.CodCiu
JOIN pais PAI on CIU.CodPais=PAI.CodPais
group by PIL.NomPil
ORDER BY COUNT(DISTINCT paI.CodPais) DESC;
```

```
--mostrar resultados de la temporada con el siguiente formato (Nombre Piloto|Apellido Piloto|Nacionalidad|Nombre Del equipo|
--Numero de podios|Numero de victorias|Numero de puntos ) ordenado por equipos usando alias
SELECT pil.NomPil,pil.Ape1pil,NA.NomNacion,E.NomEquipo,COUNT(CASE WHEN par.posicion <= 3 then 1 end ) AS podios,
COUNT(CASE when PAR.Posicion=1 THEN 1 END) AS 'numero de victorias',SUM(PAR.Puntos) AS 'NUM PUNTOS' FROM piloto pil
JOIN participa par ON pil.CodPil =par.CodPil
JOIN nacionalidad NA ON na.CodNacion =pil.CodNacion
JOIN equipo E ON E.CodEquipo = pil.CodEquipo
GROUP BY pil.NomPil, pil.Ape1pil,NA.NomNacion,E.NomEquipo
ORDER BY sum(PAR.Puntos) DESC
```

```
--mostrar resultados por equipos ordenado por puntos usando el siguiente orden(Nombre Equipo|Numero de Victorias|Numero de podios|Numero de puntos)
SELECT e.NomEquipo,COUNT(case when Posicion = 1 then 1 end) AS 'VICTORIAS',COUNT(case when Posicion <= 3 then 1 end) AS 'PODIOS' ,SUM(PAR.Puntos) AS 'PUNTOS' FROM equipo e
JOIN piloto PIL ON pil.CodEquipo = e.CodEquipo
JOIN participa PAR ON PAR.CodPil = PIL.CodPil
GROUP BY E.NomEquipo
ORDER BY SUM(PAR.Puntos) DESC
```

# CREACIÓN DE ÍNDICE



--Índice para mejorar la búsqueda de motores por fabricante

```
CREATE INDEX indiceMotorFabricante ON motor(CodFab);
```

--Índice para acelerar la búsqueda de neumáticos por fabricante

```
CREATE INDEX indiceNeumaticoFabricante ON neumatico(CodFab);
```

--Índice para mejorar las búsquedas y filtrados por equipo en la tabla piloto

```
CREATE INDEX indicePilotoEquipo ON piloto(CodEquipo);
```

--Índice para optimizar la búsqueda de carreras por gran premio

```
CREATE INDEX indiceCarreraGranPremio ON carrera(CodGranPremio);
```

--Índice para mejorar la búsqueda de incidencias en una carrera específica

```
CREATE INDEX indiceIncidenciaCarrera ON incidencia(CodCar);
```

A black and white photograph showing a Formula 1 race track through a metal fence. In the background, a racing car is visible on the track, and a DHL advertisement is on a wall. The sky is overcast.

# CREACIÓN DE VISTA

```
--Vista que muestra pilotos, su equipo y número de victorias:  
CREATE VIEW vistaPilotosEquipos  
AS  
SELECT NomPil AS 'Nombre piloto',  
       Ape1pil AS 'Primer apellido',  
       ISNULL(Ape2pil, 'Sin segundo apellido') AS 'Segundo apellido',  
       ISNULL(Victorias, 0) AS 'Victorias',  
       NomEquipo AS 'Nombre equipo'  
FROM piloto P JOIN equipo E ON (P.CodEquipo=E.CodEquipo)  
ORDER BY Victorias;
```

--Consulta a la vista:

```
SELECT * FROM vistaPilotosEquipos;
```

FITN