Base De Datos Escudería Ferrari

Mario Espasa Planells

CURSO: 1 DAM

INDICE

- Situación De La Base de Datos
- Método Entidad-Relación
- Modelo Relacional

Situación

Estructura escudería Ferrari

Debido a las nuevas modificaciones realizadas por la FIA respecto al limite presupuestario para el año 2021, la escudería Ferrari ha decidido crear una nueva base de datos con todos los datos y presupuestos de la escudería.

Primero almacenaremos los mecánicos que nos interesará almacenar su nombre, apellido, dni, teléfono y sueldo, habrá tres tipos, los encargados de realizar ajustes y reglajes al coche, los encargados del cambio de ruedas durante la carrera, durante esta, y este deberá de tener la función especifica que va a realizar como ser el encargado de levantar el coche, quitar rueda, ponerla, apretarla, ajustar un alerón, etc. Y los ingenieros de carrera que se encargarán de comunicarse con el piloto y darle instrucciones y por último tener en cuenta que habrá un jefe de mecánicos encargado de liderar a los demás mecánicos.

También hay que tener en cuenta que habrá un director técnico, del cual almacenaremos su nombre, apellidos, dni, teléfono, sueldo, el director técnico será el encargado de dirigir todo el personal de la escudería, tanto mecánicos como pilotos.

También nos interesa saber los Pilotos de carreras de la escudería, de los que almacenaremos, su nombre, teléfono, apellidos, apodos, numero, sueldo, habrá que contar las sanciones que reciban los pilotos escribir la descripción de la misma, la sanción dada y un id y habrá que almacenar el numero de sanciones de cada piloto.

Sobre el Coche de la temporada nos interesa saber el nombre del modelo y el coste total de fabricación y además interesará saber las piezas del coche, junto al coste de fabricación individual de las mismas, una descripción de cada pieza y habrá que poner el numero de piezas fabricadas, ya que si excede el numero de piezas predispuestas por la FIA puede conllevar a una sanción.

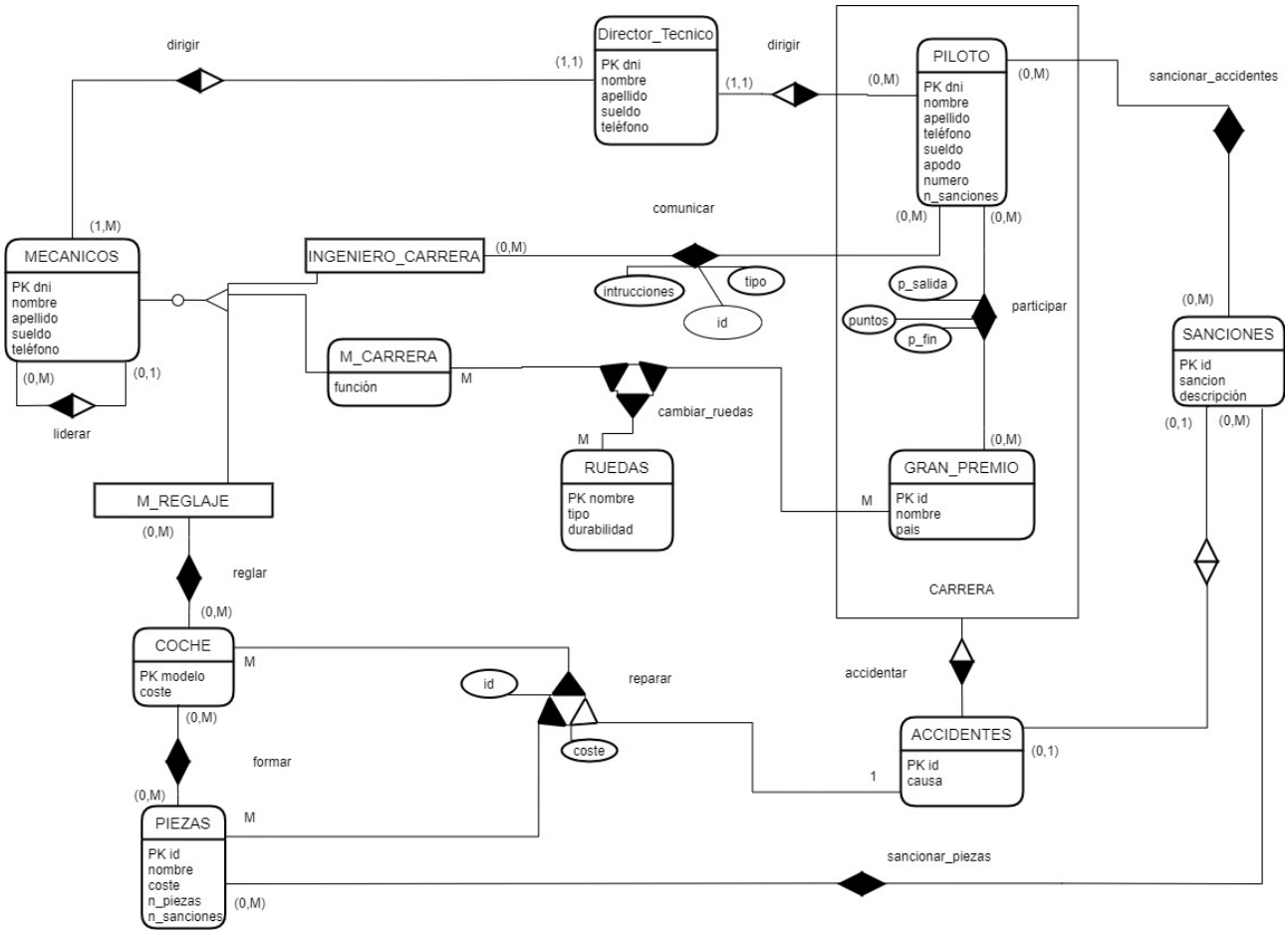
Las ruedas del monoplaza habrá que tenerlas en cuenta durante un gran premio ya que es una parte muy importante de este, entonces se deberá almacenar el nombre, tipo y durabilidad de la misma, y habrá que tener en cuenta que los mecánicos de carrera serán los encargados de realizar el cambio de ruedas al monoplaza durante todo el gran premio.

Interesará saber la cantidad de gran premios que hay en la temporada de los cuales queremos el nombre del gran premio, el pais donde se disputa y un id del mismo, en un gran premio podrán correr uno o varios pilotos a la vez, y los pilotos podrán correr o no en los gran premios y nos interesará saber de los pilotos la cantidad de puntos que consiguen después de un gran premio y la posición de salida y de finalización de los pilotos.

También hay que tener en cuenta que durante un gran premio pueden ocurrir accidentes a los pilotos, de los cuales nos interesará saber, si es por causa humana o fallo mecánico, y que estos accidentes pueden traer consigo sanciones a los pilotos.

Habrá que tener en cuenta las reparaciones hechas después del accidente de las cuales habrá que guardar un id de la misma junto una descripción y el coste, estas reparaciones estarán hechas por los mecánicos encargados de los ajustes del coche.

Modelo Entidad-Relación



Modelo Relacional

MECANICOS(dni,nombre,apellido,sueldo,teléfono,dni_dt)
PK(dni)
FK(dni_dt) → DIRECTOR_TECNICO;

M_REGLAJE(dni)
PK(dni)
FK(dni) → MECANICOS;

M_CARRERA(dni, función)
PK(dni)
FK(dni) → MECANICOS;

INGENIERO_CARRERA(dni)
PK(dni)
FK(dni) → MECANICOS;

DIRECTOR_TECNICO(dni,nombre,apellido,sueldo,teléfono)
PK(dni);

PILOTO(dni,nombre,apellido,teléfono,sueldo,apodo,número,dni_dt,n_sanciones)
PK(dni)
FK(dni_dt) → DIRECTOR_TECNICO;

GRAN_PREMIO(id,nombre,pais)
PK(id);

RUEDAS(nombre,tipo,durabilidad)
PK(nombre);

SANCIONES(id,sancion,descripción)
PK(id);

ACCIDENTES(id,causa,id_p,dni)
PK(id)
FK(dni,id_p) → PARTICIPAR;

COCHE(modelo, coste)
PK(modelo);

PIEZAS(id, nombre, coste, n_piezas,n_sanciones)
PK(id);

SANCIONAR_PIEZAS(id_pieza,id_sancion)
PK(id_pieza,id_sancion)
FK(id_sancion) → SANCIONES
FK(id_pieza) → PIEZAS;

SANCIONAR_PILOTOS(dni,id,)
PK(id,dni)
FK(id) → SANCIONES

FK(dni) → PILOTO;

REGLAR(dni,modelo)
PK(dni,modelo)
FK(dni) → M_REGLAJE
FK(modelo) → COCHE;

FORMAR(id, modelo)
PK(id, modelo)
FK(id) → PIEZAS
FK(modelo) → COCHE;

PARTICIPAR(dni,id,p_salida,puntos,p_fin)
PK(dni,id)
FK(id) → GRAN_PREMIO
FK(dni) → PILOTOS;

COMUNICAR(id,dni_m,dni_p,tipo,instrucciones)
PK(id,dni_m,dni_p)
FK(dni_m) → INGENIERO_CARRERA
FK(dni_p) → PILOTO;

CAMBIAR_RUEDAS(nombre,dni,id)
PK(nombre,dni,id)
FK(nombre) → RUEDAS
FK(id) → GRAN_PREMIO
FK(dni) → M_CARRERA;

REPARAR(modelo,id_pieza,id_accidente)
PK(modelo,id_pieza)
VNN(id_accidente)
FK(modelo) → COCHE
FK(id_pieza) → PIEZAS
FK(id_accidente) → ACCIDENTES;

DDL

```
CREATE TABLE gran_premio (
   id VARCHAR(2),
   nombre VARCHAR(50),
        pais VARCHAR(30),
   PRIMARY KEY (id)
);

CREATE TABLE coche (
   modelo VARCHAR(4),
   coste NUMERIC(11,2),
   PRIMARY KEY (modelo)
);


CREATE TABLE director_tecnico (
   dni VARCHAR(9),
   nombre VARCHAR(30),
        apellido VARCHAR(30),
        sueldo NUMERIC(7,2),
        telefono VARCHAR(9),
   PRIMARY KEY (dni)
);

CREATE TABLE mecanicos (
   dni VARCHAR(9),
   nombre VARCHAR(30),
        apellido VARCHAR(30),
        telefono VARCHAR(9),
```

```
        dni_dt VARCHAR(9),
    sueldo NUMERIC(9,2),
    PRIMARY KEY (dni),
    FOREIGN KEY (dni_dt) REFERENCES director_tecnico(dni)
);

CREATE TABLE m_reglaje (
    dni VARCHAR(9),
    PRIMARY KEY (dni),
    FOREIGN KEY (dni) REFERENCES mecanicos(dni)
);

CREATE TABLE m_carrera (
    dni VARCHAR(9),
        funcion VARCHAR(30),
    PRIMARY KEY (dni),
    FOREIGN KEY (dni) REFERENCES mecanicos(dni)
);

CREATE TABLE ingeniero_carrera (
    dni VARCHAR(9),
    PRIMARY KEY (dni),
    FOREIGN KEY (dni) REFERENCES mecanicos(dni)
);



CREATE TABLE pilotos (
    dni VARCHAR(9),
    nombre VARCHAR(30),
        apellido VARCHAR(30),
        telefono VARCHAR(9),
```

```
        sueldo NUMERIC(10,2),
        apodo VARCHAR(20),
        numero VARCHAR(2),
        dni_dt VARCHAR(9),
        n_sanciones NUMERIC(2),
    PRIMARY KEY (dni),
    FOREIGN KEY (dni_dt) REFERENCES director_tecnico(dni)
);


CREATE TABLE piezas(
id VARCHAR(5),
nombre VARCHAR(20),
coste NUMERIC(9,2),
n_piezas NUMERIC(2),
n_sanciones NUMERIC(2),
PRIMARY KEY (id)
);

CREATE TABLE ruedas (
    nombre VARCHAR(2),
    tipo VARCHAR(10),
        durabilidad VARCHAR(10),
    PRIMARY KEY (nombre)
);

CREATE TABLE sancion (
    id VARCHAR(2),
    sancion VARCHAR(100),
        descripcion VARCHAR(100),
    PRIMARY KEY (id)
);
```

```
CREATE TABLE sancionar_piezas(
piez_id VARCHAR(5),
sancion_id VARCHAR(2),
PRIMARY KEY (piez_id,sancion_id),
CONSTRAINT FK_sancionar_piez1 FOREIGN KEY (piez_id) REFERENCES piezas(id),
CONSTRAINT FK_sancionar_piez2 FOREIGN KEY (sancion_id) REFERENCES sancion(id)
);

CREATE TABLE cambiar_ruedas(
dni_m VARCHAR(9),
gp_id VARCHAR(2),
r_nombre VARCHAR(2),
PRIMARY KEY (dni_m,gp_id,r_nombre),
CONSTRAINT FK_cambiarRuedas1 FOREIGN KEY (dni_m) REFERENCES m_carrera(dni),
CONSTRAINT FK_cambiarRuedas2 FOREIGN KEY (gp_id) REFERENCES gran_premio(id),
CONSTRAINT FK_cambiarRuedas3 FOREIGN KEY (r_nombre) REFERENCES ruedas(nombre)
);


CREATE TABLE reglar(
   dni VARCHAR(9),
   modelo VARCHAR(4),
   PRIMARY KEY (dni,modelo),
   CONSTRAINT FK_reglar1 FOREIGN KEY (dni) REFERENCES m_reglaje(dni),
   CONSTRAINT FK_reglar2 FOREIGN KEY (modelo) REFERENCES coche(modelo)
);

CREATE TABLE formar(
   id VARCHAR(5),
   modelo VARCHAR(4),
```

```
    PRIMARY KEY (id,modelo),
    CONSTRAINT FK_formar1 FOREIGN KEY (id) REFERENCES piezas(id),
    CONSTRAINT FK_formar2 FOREIGN KEY (modelo) REFERENCES coche(modelo)
);

CREATE TABLE participar(
    dni VARCHAR(9),
    id VARCHAR(2),
    p_salida NUMERIC(2) CHECK (p_salida <=20),
    puntos NUMERIC(2),
    p_fin NUMERIC(2),
    PRIMARY KEY (dni,id),
    CONSTRAINT FK_participar1 FOREIGN KEY (id) REFERENCES gran_premio(id),
    CONSTRAINT FK_participar2 FOREIGN KEY (dni) REFERENCES pilotos(dni)
);

CREATE TABLE accidente (
    id VARCHAR(2),
    causa VARCHAR(50),
        id_p VARCHAR(2),
        dni VARCHAR(9),
    PRIMARY KEY (id),
CONSTRAINT FK_accidente FOREIGN KEY (dni,id_p) REFERENCES participar(dni,id)
);

CREATE TABLE comunicar(
    id VARCHAR(2),
    dni_m VARCHAR(9),
    dni_p VARCHAR(9),
    tipo VARCHAR(30),
    instrucciones VARCHAR(100),
    PRIMARY KEY(id),
```

```
        CONSTRAINT FK_comunicar1 FOREIGN KEY (dni_m) REFERENCES ingeniero_carrera(dni),
    CONSTRAINT FK_comunicar2 FOREIGN KEY (dni_p) REFERENCES pilotos(dni)

);




CREATE TABLE reparar(
    modelo VARCHAR(4),
    id_pieza VARCHAR (5),
    id_accidente VARCHAR(3),
    PRIMARY KEY(modelo, id_pieza,id_accidente),
    CONSTRAINT FK_reparar1 FOREIGN KEY (modelo) REFERENCES coche(modelo),
    CONSTRAINT FK_reparar2 FOREIGN KEY (id_pieza) REFERENCES piezas(id),
    CONSTRAINT FK_reparar3 FOREIGN KEY (id_accidente) REFERENCES accidente(id)
);

CREATE TABLE sancionar_pilotos(
    dni VARCHAR(9),
    id VARCHAR(2),
    PRIMARY KEY (dni,id),
    CONSTRAINT FK_sancionar_pilotos1 FOREIGN KEY (dni) REFERENCES pilotos(dni),
    CONSTRAINT FK_sancionar_pilotos2 FOREIGN KEY (id) REFERENCES sancion(id)

);

ALTER TABLE gran_premio ADD COLUMN circuito VARCHAR(30);
```

DML

```
INSERT INTO gran_premio VALUES
('01','GP de Bahrein','Bahrein','Sakhir'),
('02','GP de Emilia-Romagna','Italia','Imola'),
('03','GP de España','España','Montmeló'),
('04','GP de Mónaco','Mónaco','Montecarlo'),
('05','GP de Azerbaiyán','Azerbaiyán','Bakú'),
('06','GP de Francia','Francia','Paul Ricard'),
('07','GP de Estiria','Austria','Spielberg'),
('08','GP de Austria','Austria','Spielberg'),
('09','GP de Gran Bretaña','Gran Bretaña','Silverstone'),
('10','GP de Hungria','Hungria','Hungaroring'),
('11','GP de Belgica','Belgica','Spa-Francorchamps'),
('12','GP de Paises Bajos','Paises Bajos','Zandvoort'),
('13','GP de Italia','Italia','Monza'),
('14','GP de Rusia','Rusia','Sochi'),
('15','GP de Turquia','Turquia','Estanbul'),
('16','GP de EE UU','Estados Unidos','Circuito de las americas'),
('17','GP de Ciudad de México','México','Hermanos Rodriguez'),
('18','GP de Sao Paulo','Brasil','Interlagos'),
('19','GP de Qatar','Qatar','Losail'),
('20','GP de Arabia Saudi','Arabia Saudi','Yeda'),
('21','GP de Abu Dhabi','Abu Dhabi','Yas Marina')
;

INSERT INTO coche VALUES
('SF21',5651000);

INSERT INTO director_tecnico VALUES
('10296382M','Mattia','Binotto',80000,'649245676');
```

```
INSERT INTO mecanicos VALUES
('49235298O','Ricardo','Addami','650642236','10296382M',110000),
('85638135N','Jock','Clear','670422187','10296382M',110000),
('18734235N','Steve','Johnson','670422187','10296382M',45000),
('42351623J','Paco','Gomez','672187987','10296382M',45000),
('18734235A','Will','Lebron','789381047','10296382M',45000),
('18351239S','James','Smith','610934187','10296382M',45000),
('19291235J','Anthony','Davis','687653487','10296382M',50000),
('12894735H','Michael','Jordan','677985673','10296382M',50000),
('68736535C','Bill','Clinton','656412187','10296382M',50000),
('23736727G','Santiago','Matinez','777984562','10296382M',50000);

INSERT INTO ingeniero_carrera VALUES
('49235298O'),
('18734235N');

INSERT INTO m_carrera VALUES
('18734235N','Cambiar Ruedas'),
('42351623J','Levantar morro'),
('18734235A','Levantar parte trasera'),
('18351239S','Control de semaforo');

INSERT INTO m_reglaje VALUES
('19291235J'),
('12894735H'),
('68736535C'),
('23736727G');

INSERT INTO pilotos VALUES
('49926127C','Carlos','Sainz','608983634',10200000, 'Charles LeChair','16','10296382M'),
('12943872K','Charles','Leclerc','632094142',6800000,'Chili','55','10296382M');
```

```
INSERT INTO piezas VALUES
('MSF21','Motor',4100000,4),
('CCF21','Caja de cambios',570000,5),
('TEF21','Tubo de escape',230000,6),
--DE AQUI PARA ABAJO HACER UN UPDATE PARA QUE SEAN NULL
('CHF21','Chasis',105000,20),
('FSF21','Frenos',185000,25),
('ADF21','Aleron delantero',190000,10),
('ATF21','Aleron trasero',190000,10),
('VSF21','Volante',26000,10),
('ERF21', 'ERS', 200000,10);


INSERT INTO ruedas VALUES
('C1','seco','duro'),
('C2','seco','medio'),
('C3','seco','blando'),
('C4','mojado','intermedio'),
('C5','mojado','mojado');



INSERT INTO participar VALUES
('49926127C','01',8,4,8),
('12943872K','01',4,8,6),
('49926127C','02',11,10,5),
('12943872K','02',4,12,4),
('49926127C','03',6,6,7),
('12943872K','03',4,12,4),
('49926127C','04',4,18,2),
('12943872K','04',null,0,20),
('49926127C','05',5,4,8),
```

('12943872K','05',1,12,4),

('49926127C','06',5,0,11),

('12943872K','06',7,0,16),

('49926127C','07',12,8,6),

('12943872K','07',7,6,7),

('49926127C','08',10,10,5),

('12943872K','08',12,4,8),

('49926127C','09',10,8,6),

('12943872K','09',4,18,2),

('49926127C','10',15,15,3),

('12943872K','10',7,0,18),

('49926127C','11',11,1,10),

('12943872K','11',9,2,8),

('49926127C','12',6,6,7),

('12943872K','12',5,10,5),

('49926127C','13',6,8,6),

('12943872K','13',5,12,4),

('49926127C','14',2,15,3),

('12943872K','14',19,0,15),

('49926127C','15',19,4,8),

('12943872K','15',3,12,4),

('49926127C','16',5,6,7),

('12943872K','16',4,12,4),

('49926127C','17',6,8,6),

('12943872K','17',8,10,5),

('49926127C','18',3,9,6),

('12943872K','18',6,10,5),

('49926127C','19',5,6,7),

('12943872K','19',13,4,8),

('49926127C','20',15,4,8),

('12943872K','20',4,6,7),

('49926127C','21',5,15,3),

('12943872K','21',7,1,10);

INSERT INTO accidente VALUES

('01','Choque contra una barrera','04','12943872K'),

('02', 'Golpe en el aleron trasero con otro piloto','06','12943872K'),

('03', 'Golpe contra un muro tras un subviraje','07','12943872K'),

('04', 'Pinchazo tras pisar un trozo de chasis', '20','49926127C')

;

INSERT INTO comunicar VALUES

('01','49235298O','49926127C','Entrar a boxes','Tienes que entrar a boxes en la siguiente vuelta'),

('02','18734235N','12943872K','Fallo motor','Estas teniendo fallos en el motor en bajas revoluciones evita bajarlas demasiado'),

('03','18734235N','12943872K','Bandera amarilla','Ten cuidado hay bandera amarilla en el tercer sector'),

('04','18734235N','12943872K','Golpe','Te encuentras bien, ha sido un duro golpe, quédate hay esta llegando el safety car');

INSERT INTO sancion VALUES

('01','10 posiciones de penalizacion','Por exceder el limite de motores posibles en un mundial'),

('02','10 segundos en carrera','Por adelantar por fuera de pista y no devolver la posicion'),

('03','8 posiciones de penalizacion','Por cambiar el sistema electrico del motor '),

('04','Multa de 10.000 euros','Debido a tocar un Formula 1 de un equipo contrario');

INSERT INTO cambiar_ruedas VALUES

('18734235N','01','C2'),

('18734235N','02','C1'),

('18734235N','02','C4'),

('18734235N','03','C2'),

('18734235N','04','C3'),

('18734235N','05','C2'),

('18734235N','06','C1'),

('18734235N','06','C4'),

('18734235N','07','C1'),

('18734235N','08','C1'),

('18734235N','08','C2'),

('18734235N','09','C1'),

('18734235N','10','C5'),

('18734235N','11','C2'),

('18734235N','12','C3'),

('18734235N','13','C2'),

('18734235N','13','C1'),

('18734235N','14','C2'),

('18734235N','14','C4'),

('18734235N','15','C2'),

('18734235N','16','C4'),

('18734235N','16','C2'),

('18734235N','17','C1'),

('18734235N','18','C1'),

('18734235N','19','C2'),

('18734235N','20','C2'),

('18734235N','21','C2');

INSERT INTO sancionar_pilotos VALUES

('49926127C','01'),

('12943872K','02'),

('12943872K','03'),

('12943872K','04');

```
INSERT INTO reparar VALUES
('SF21','CHF21','01'),
('SF21','ATF21','02'),
('SF21','ADF21','03');


INSERT INTO formar VALUES
('MSF21','SF21'),
('CCF21','SF21'),
('TEF21','SF21'),
('CHF21','SF21'),
('FSF21','SF21'),
('ADF21','SF21'),
('ATF21','SF21'),
('VSF21','SF21');


INSERT INTO reglar VALUES
('19291235J','SF21'),
('12894735H','SF21'),
('68736535C','SF21'),
('23736727G','SF21');


INSERT INTO sancionar_piezas VALUES
('MSF21','01'),
('ERF21','02');
```

SQL

--h

UPDATE pilotos set n_sanciones = (SELECT count(id) FROM sancionar_pilotos WHERE dni = '49926127C') WHERE nombre = 'Carlos';

UPDATE pilotos set n_sanciones = (SELECT count(id) FROM sancionar_pilotos WHERE dni != '49926127C') WHERE nombre = 'Charles';

UPDATE piezas set n_sanciones = (SELECT count(sancion_id) FROM sancionar_piezas WHERE piez_id = 'ERF21') WHERE id = 'ERF21';

UPDATE piezas set n_sanciones = (SELECT count(sancion_id) FROM sancionar_piezas WHERE piez_id = 'VSF21') WHERE id = 'VSF21';

UPDATE piezas set n_sanciones = (SELECT count(sancion_id) FROM sancionar_piezas WHERE piez_id = 'ATF21') WHERE id = 'ATF21';

UPDATE piezas set n_sanciones = (SELECT count(sancion_id) FROM sancionar_piezas WHERE piez_id = 'ADF21') WHERE id = 'ADF21';

UPDATE piezas set n_sanciones = (SELECT count(sancion_id) FROM sancionar_piezas WHERE piez_id = 'FSF21') WHERE id = 'FSF21';

UPDATE piezas set n_sanciones = (SELECT count(sancion_id) FROM sancionar_piezas WHERE piez_id = 'CHF21') WHERE id = 'CHF21';

UPDATE piezas set n_sanciones = (SELECT count(sancion_id) FROM sancionar_piezas WHERE piez_id = 'TEF21') WHERE id = 'TEF21';

UPDATE piezas set n_sanciones = (SELECT count(sancion_id) FROM sancionar_piezas WHERE piez_id = 'CCF21') WHERE id = 'CCF21';

UPDATE piezas set n_sanciones = (SELECT count(sancion_id) FROM sancionar_piezas WHERE piez_id = 'MSF21') WHERE id = 'MSF21';

--a. 5 Consultas simples de una sola tabla

--Muestra el id y el nombre del gran premio que contiene en su nombre Estiria

SELECT id, nombre

FROM gran_premio

WHERE nombre

```
LIKE '%Estiria%';

--Muestra el nombre de los mecanicos que cobren mas de 50000 euros
SELECT nombre
FROM mecanicos
WHERE sueldo > 50000;

--Muestra el nombre de los mecanicos que su nombre empiece por J
SELECT nombre
FROM mecanicos
WHERE nombre
LIKE 'J%';

--Muestra las piezas con alguna una sanción
SELECT nombre
FROM piezas
WHERE n_sanciones>0;

--Muestra los numeros de telefono de los pilotos
SELECT telefono
FROM pilotos;


--b. 2 Actualizaciones y 2 Borrados en cualquier tabla

DELETE FROM sancionar_pilotos WHERE id = '04'
DELETE FROM sancion WHERE id = '04'


--c. 3 Consultas con más de 1 tabla
```

```
--Muestra el id del gran premio y toda la información de las ruedas cambiadas en el gran premio de Francia
SELECT gran_premio.id, ruedas.nombre, ruedas.tipo, ruedas.durabilidad
FROM ruedas, gran_premio, cambiar_ruedas
WHERE ruedas.nombre = cambiar_ruedas.r_nombre
AND cambiar_ruedas.gp_id = gran_premio.id
AND gran_premio.pais = 'Francia';

--Muestrame las piezas afectadas por una sancion

SELECT sancion.id, piezas.*
FROM sancion, sancionar_piezas,piezas
WHERE piezas.id = sancionar_piezas.piez_id
AND sancionar_piezas.sancion_id = sancion.id;

--Muestrame los datos de los accidentes de cada gran premio

SELECT gran_premio.id, accidente.*
FROM accidente,gran_premio,participar
WHERE accidente.id = participar.id
AND participar.id = gran_premio.id;



--d. 3 Consultas usando funciones

--Muestrame el la cantidad de mecanicos que hay
SELECT count(nombre) AS Cantidad_Mecanicos
FROM mecanicos;

--Muestrame la cantidad de veces usada los neumaticos blandos en el gran premio de brasil
SELECT count(r_nombre)
```

```
FROM cambiar_ruedas
WHERE r_nombre  =
   (SELECT nombre
   FROM ruedas
   WHERE durabilidad = 'blando')
AND gp_id =
   (SELECT id
   FROM gran_premio
   WHERE pais = 'Brasil')

--Muestrame la suma de puntos de charles leclerc
SELECT sum(puntos)
FROM participar
WHERE dni =
   (SELECT dni
   FROM pilotos
   WHERE nombre = 'Charles')


--AVG la posición media en la que queda cada piloto al acabar el gran premio

SELECT pilotos.nombre, AVG(p_fin) AS posicion
FROM pilotos, participar
WHERE pilotos.dni = participar.dni
GROUP BY pilotos.nombre



--e. 2 Consultas usando group by

--Saber cantidad de neumaticos utilizados a lo largo de todos los grandes premios
SELECT r_nombre, count(gp_id)
```

```
FROM cambiar_ruedas
GROUP BY r_nombre;

--Muestra los mecanicos que su nombre empiece por vocal, su sueldo esté entre 10000 y 70000 euros y agrupados por su nombre, apellido y dni
SELECT *
FROM mecanicos
WHERE UPPER(nombre) SIMILAR TO '(A|E|I|O|U)%'
AND sueldo BETWEEN 10000 AND 70000
GROUP BY nombre, apellido,dni;

--f. 2 Consultas utilizando subconsultas

--Muestra los gran permios en los que Carlos Sainz haya quedado en el podio
SELECT gran_premio.nombre
FROM gran_premio, participar
WHERE gran_premio.id = participar.id
AND p_fin < 4
AND dni =
   (SELECT dni
   FROM pilotos
   WHERE nombre = 'Carlos')

--Muestra las sanciones que fueron culpa del motor
SELECT sancion.*
FROM sancionar_piezas, sancion
WHERE sancionar_piezas.sancion_id = sancion.id
AND piez_id =
   (SELECT id
   FROM piezas
   WHERE nombre = 'Motor');
```

--g. 2 Consultas usando group by con having

--Saber cantidad de neumaticos utilizados a lo largo de todos los grandes premios que sean una cantidad mayor a 5

```
SELECT r_nombre, count(gp_id)
FROM cambiar_ruedas
GROUP BY r_nombre;
having count(*) >5
```

--Saber las veces que cada piloto quedó último

```
SELECT pilotos.nombre, count(p_fin)
FROM pilotos, participar
WHERE pilotos.dni = participar.dni
GROUP BY pilotos.nombre, p_fin
HAVING p_fin = 20
```

Vistas

Vista en la que muestra los datos generales de los pilotos sin datos privados de los mismos, sería una vista enfocada para personal diferente al gestor de la base de datos

```
CREATE OR REPLACE VIEW g_pilotos

AS

SELECT CONCAT(nombre, ', ' ,apellido) AS nombre, apodo, numero

FROM pilotos;
```

Vista en la que se muestra el nombre de los pilotos junto a sus respectivas sanciones

```
CREATE OR REPLACE VIEW faltas_pilotos

AS

SELECT CONCAT(nombre, ', ' ,apellido) AS nombre, sancion.descripcion AS accion ,sancion.sancion AS penalizacion

FROM pilotos, sancion, sancionar_pilotos

WHERE sancion.id = sancionar_pilotos.id

AND pilotos.dni = sancionar_pilotos.dni;
```

Funciones

Funcion que indica la diferencia de sueldo entre los 2 pilotos de la escuderia

```
CREATE OR REPLACE FUNCTION diferencia_sueldo()
   RETURNS void AS $$
   DECLARE
      rec_pil RECORD;
                  sueldoCarlos FLOAT DEFAULT 0;
                  sueldoCharles FLOAT DEFAULT 0;
                  diferencia FLOAT DEFAULT 0;
                  cur_piloto CURSOR FOR SELECT * FROM pilotos;

   BEGIN

                  OPEN cur_piloto;

         LOOP

                  FETCH cur_piloto INTO rec_pil;
            EXIT WHEN NOT FOUND;

                  if(rec_pil.nombre = 'Carlos') THEN
                        sueldoCarlos = rec_pil.sueldo;
                  END IF;

                  if(rec_pil.nombre = 'Charles') THEN
                        sueldoCharles = rec_pil.sueldo;
                  END IF;

         END LOOP;

                  CLOSE cur_piloto;
```

```
        diferencia := sueldoCharles - sueldoCarlos;

        if(diferencia < 0) THEN
        diferencia := diferencia * -1;
        END IF;

        IF(sueldoCharles > sueldoCarlos) THEN
                RAISE INFO'Tienen una diferencia de % euros, el piloto que mas cobra es Charles' , diferencia;
        ELSIF (sueldoCharles < sueldoCarlos) THEN
                RAISE INFO 'Tienen una diferencia de % euros, el piloto que mas cobra es Carlos' , diferencia;
        ELSE
                RAISE INFO 'Tienen el mismo sueldo';
        END IF;

END;
$$ LANGUAGE plpgsql;
```

Funcion que al pasarle la informacion necesaria introduzca una nueva linea a la base de datos a la tabla de gran_premio

```
CREATE OR REPLACE FUNCTION inserta_gran_premio(nombre varchar, pais varchar, circuito varchar)
   RETURNS void AS $$
   DECLARE
   id VARCHAR;
   BEGIN
                id:= (SELECT count(*) FROM gran_premio)+1;

      INSERT INTO gran_premio VALUES (id,nombre,pais,circuito);
```

```
END;
$$ LANGUAGE plpgsql;
```

Funcion que se encargue de mostrar una tabla desde una linea de inicio y una linea final

```
CREATE OR REPLACE FUNCTION mostrarInfoTabla(tabla VARCHAR, empieza int, final int)
        RETURNS void AS $$
        DECLARE
                rec_tabla RECORD;
                cur_tabla refcursor;
                count int DEFAULT 0;
        BEGIN

      if final < empieza THEN
        RAISE EXCEPTION 'El numero de la linea final es menor que la de inicio';
      END IF;



                OPEN cur_tabla FOR EXECUTE 'SELECT * FROM '|| tabla;
                LOOP
                        FETCH cur_tabla INTO rec_tabla;
                        EXIT WHEN NOT FOUND OR count = final;
                        count := count +1;

        IF count <= final AND count >= empieza THEN
           RAISE INFO '% la tabla %: %', count,tabla, rec_tabla;
        END IF;

                END LOOP;
                CLOSE cur_tabla;
```

```
            EXCEPTION
                WHEN undefined_table THEN
                RAISE EXCEPTION 'la tabla % no existe',tabla;
        END;
$$ LANGUAGE plpgsql;
```

Funcion que muestre la diferencia de posiciones entre el inicio y el final de carrera

```
CREATE OR REPLACE FUNCTION diferencias_posiciones(piloto varchar, gPremio varchar)
   RETURNS void AS $$
   DECLARE

   idG VARCHAR DEFAULT null;
   idP VARCHAR DEFAULT null;
   pFin int DEFAULT 0;
   pInicio int DEFAULT 0;
   diferencia int DEFAULT 0;

      cur_gp CURSOR FOR
          SELECT * FROM gran_premio;
      rec_gp RECORD;

   BEGIN

   if piloto = 'Carlos' THEN
      idP = '49926127C';
   END IF;

   if piloto = 'Charles' THEN
      idP = '12943872K';
   END IF;
```

```
if idP = null THEN

   RAISE EXCEPTION 'No ha escrito un nombre de piloto valido';

END IF;

OPEN cur_gp;

LOOP

   FETCH cur_gp INTO rec_gp;
        EXIT WHEN NOT FOUND;

   if rec_gp.nombre = gPremio THEN

      idG := rec_gp.id;

   END if;


END LOOP;

CLOSE cur_gp;

if idG = null THEN

   RAISE EXCEPTION 'No ha escrito un nombre de gran premio valido';

END IF;


pInicio := (SELECT p_salida FROM participar WHERE dni = idP AND id = idG);
```

```
    pFin := (SELECT p_fin FROM participar WHERE dni = idP AND id = idG);

    IF pInicio = null  THEN

       RAISE INFO 'El piloto % no ha ha corrido en la carrera de %',piloto,gPremio;

    END IF;

    diferencia = pInicio -pFin;

    if diferencia <0 THEN

       RAISE INFO 'El piloto % Ha bajado % posiciones',piloto,(diferencia * -1);

    ELSIF diferencia = 0 THEN

          RAISE INFO 'El piloto % Ha quedado en la misma posicion',piloto;
    ELSE

    RAISE INFO 'El piloto % Ha subido % posiciones',piloto,diferencia ;

    END IF;

END;
$$ LANGUAGE plpgsql;
```

Triggers

Trigger que cuando eliminas al director tecnico de la base de datos iguala a null los pilotos

```
CREATE OR REPLACE FUNCTION borrado_director()
	RETURNS TRIGGER AS $$
	BEGIN

			UPDATE pilotos
			SET dni_dt = NULL;
			RETURN OLD;
	 RAISE INFO 'Se ha ejecutado un trigger correctamente Y se ha modificado la tabal pilotos exitosamente';

	END;
$$ LANGUAGE plpgsql;

CREATE OR REPLACE TRIGGER delete_director
BEFORE DELETE ON director_tecnico
FOR EACH ROW
EXECUTE PROCEDURE borrado_director();
```

Triggers que cuando cambias el dni del director iguala el dni_dt a los pilotos al nuevo

```
CREATE OR REPLACE FUNCTION actualización_director_primero()
	RETURNS TRIGGER AS $$
	BEGIN

			UPDATE pilotos
			SET dni_dt = null;
			RETURN new;
	RAISE INFO 'Se ha ejecutado un trigger correctamente al actualizar la información';
```

```
        END;
$$ LANGUAGE plpgsql;

CREATE OR REPLACE TRIGGER update_director1
BEFORE UPDATE OF dni ON director_tecnico
FOR EACH ROW
EXECUTE PROCEDURE actualización_director_primero();



CREATE OR REPLACE FUNCTION actualización_director_segundo()
        RETURNS TRIGGER AS $$
        BEGIN

                UPDATE pilotos
                SET dni_dt = NEW.dni;
                RETURN NULL;
        RAISE INFO 'Se ha ejecutado un trigger correctamente al actualizar la información';



        END;
$$ LANGUAGE plpgsql;

CREATE OR REPLACE TRIGGER update_director2
AFTER UPDATE OF dni ON director_tecnico
FOR EACH ROW
EXECUTE PROCEDURE actualización_director_segundo();

Trigger que comprueba si existe o no un director tecnico, si existe no se realizara el insert
CREATE OR REPLACE FUNCTION numero_director()
        RETURNS TRIGGER AS $$
   DECLARE
```

```
rec_dt RECORD;
        cur_dt CURSOR FOR
    SELECT * FROM director_tecnico;

count int DEFAULT 0;

  BEGIN

        OPEN cur_dt;
        LOOP

            FETCH cur_dt INTO rec_dt;
            EXIT WHEN NOT FOUND;

  count := count +1;

        END LOOP;

CLOSE cur_dt;

IF count = 1 THEN

  RAISE EXCEPTION 'Ya existe un director tecnico en la base de datos';

ELSE

  return new;

END IF;

  END;
```

```
$$ LANGUAGE plpgsql;

CREATE OR REPLACE TRIGGER insertar_director
BEFORE Insert ON director_tecnico
FOR EACH ROW
EXECUTE PROCEDURE numero_director();



Trigger que actualiza las tablas de mecanicos al borrarse
CREATE OR REPLACE FUNCTION actualizaciónes_mecanicos()
        RETURNS TRIGGER AS $$
   DECLARE

      rec_r RECORD;
      rec_c RECORD;
      rec_i RECORD;

            cur_r CURSOR FOR
         SELECT * FROM m_reglaje;

      cur_c CURSOR FOR
         SELECT * FROM m_carrera;

      cur_i CURSOR FOR
         SELECT * FROM ingeniero_carrera;

        BEGIN

      OPEN cur_r;

            LOOP
```

```
    FETCH cur_r INTO rec_r;
    EXIT WHEN NOT FOUND;

    IF rec_r.dni = old.dni THEN
                        DELETE FROM reglar WHERE dni = old.dni;
       DELETE FROM m_reglaje WHERE dni = old.dni;
    END IF;

END LOOP;

CLOSE cur_r;
OPEN cur_c;

LOOP

    FETCH cur_c INTO rec_c;
    EXIT WHEN NOT FOUND;

    IF rec_c.dni = old.dni THEN
                        DELETE FROM cambiar_ruedas WHERE dni_m = old.dni;
       DELETE FROM m_carrera WHERE dni = old.dni;
    END IF;

END LOOP;

CLOSE cur_c;
OPEN cur_i;

LOOP

    FETCH cur_i INTO rec_i;
    EXIT WHEN NOT FOUND;
```

```
        IF rec_i.dni = old.dni THEN

                                DELETE FROM comunicar WHERE dni_m = old.dni;
           DELETE FROM ingeniero_carrera WHERE dni = old.dni;
        END IF;

    END LOOP;

    CLOSE cur_i;

    return old;

        END;
$$ LANGUAGE plpgsql;

CREATE OR REPLACE TRIGGER mecanicos_actualizacion
BEFORE DELETE ON mecanicos
FOR EACH ROW
EXECUTE PROCEDURE actualizaciónes_mecanicos();
```