

# LaLiga EA Sports

Proyecto Base de Datos

Base de Datos

---



**Rubén Gragera González**

**1º DAW Mañana**

**IES Alixar**

**29/03/2025**

## ÍNDICE

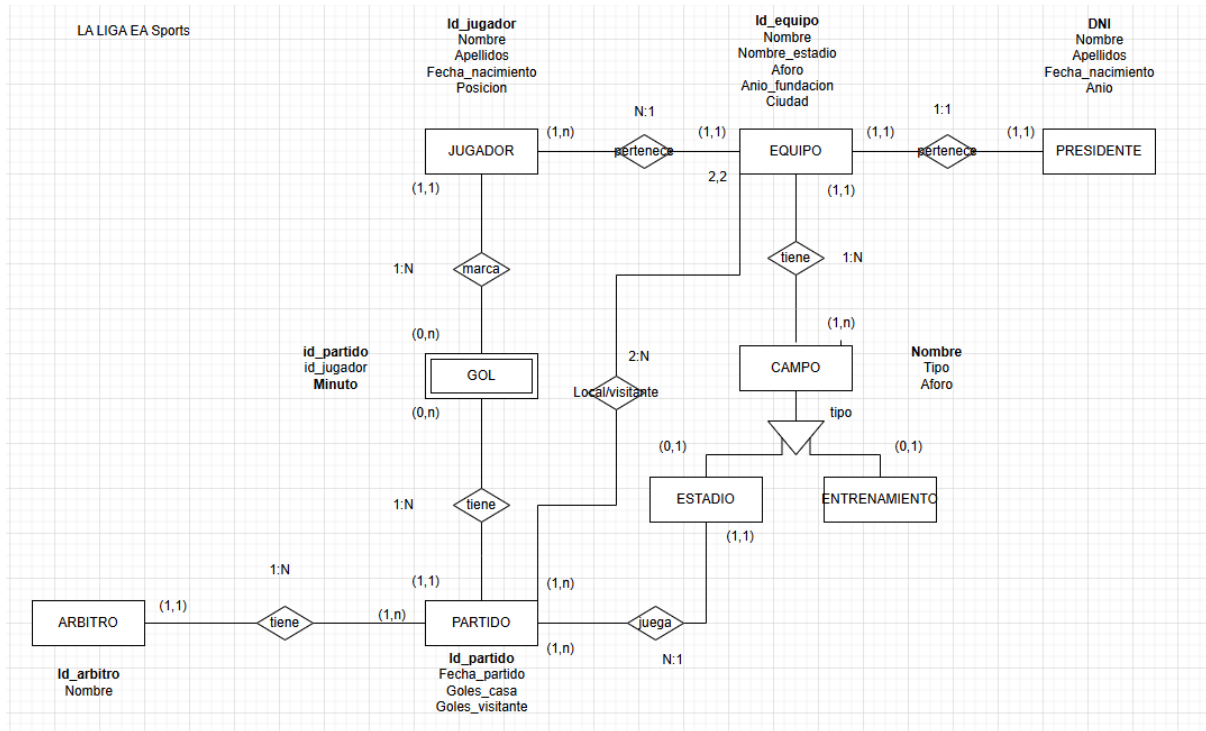
Introducción.....	3
Modelo Entidad Relación.....	4
Modelo relacional.....	5
Ejecutar Script.....	7
Carga masiva de datos.....	9
Consultas.....	11
Carga masiva en dos tablas con 1000 datos.....	14
5 consultas multitabla.....	15
Vistas.....	20
Funciones.....	22
Procedimientos.....	24
Triggers.....	28
AWS.....	31
GitHub.....	31
Conclusión.....	32

## Introducción

Este proyecto tiene como objetivo desarrollar una base de datos para gestionar información de la Liga EA Sports. La base de datos permitirá organizar y consultar de manera eficiente los datos relevantes, facilitando el análisis de rendimiento de la competición. La meta es hacer que los datos sean más fáciles de acceder y mejorar la experiencia de los jugadores y organizadores dentro del sistema de la Liga.

## Modelo Entidad Relación

Mi proyecto presenta un total de 7 entidades, una de ellas con herencia y otra débil:



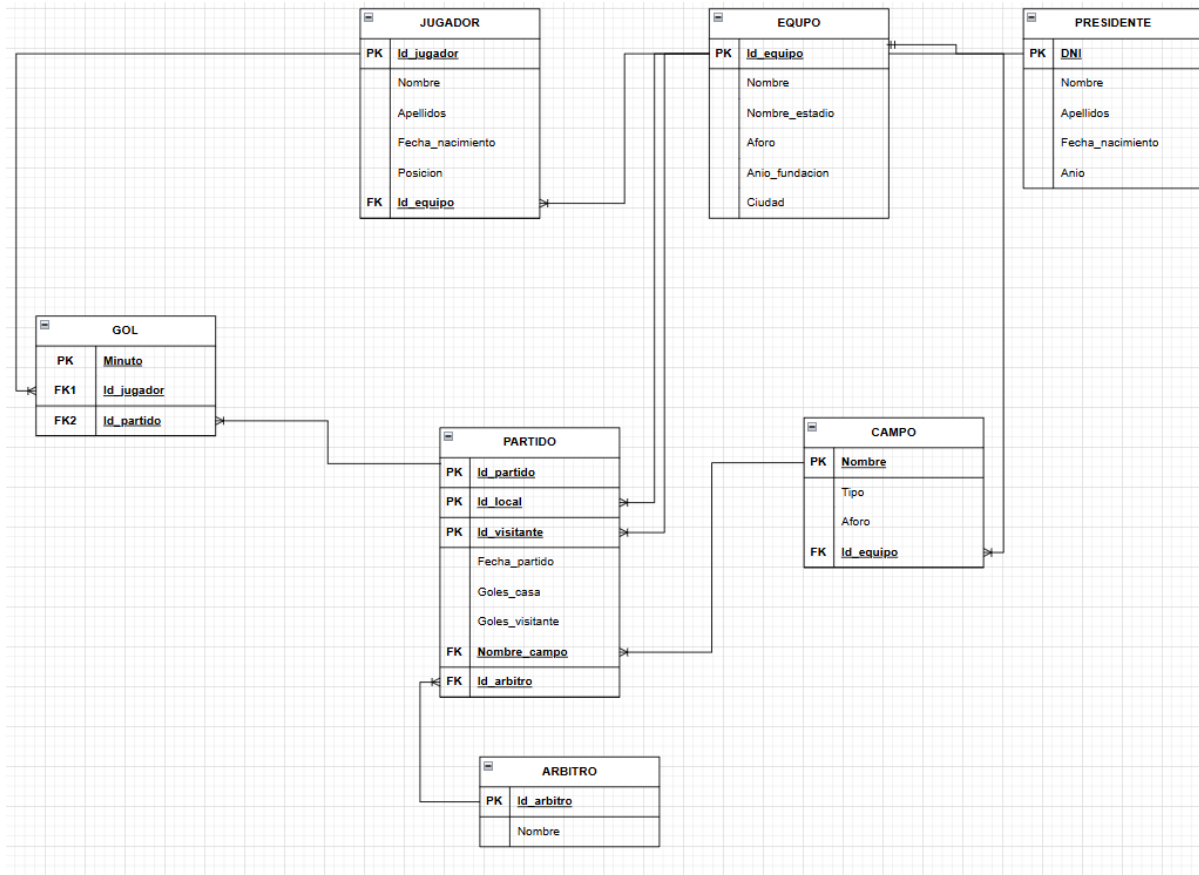
Entre la entidad PARTIDO y EQUIPO sale una relación 2:N, esto se debe a que un mismo partido lo juegan sólo dos equipos y al ser sólo dos, se puede poner así y resulta más cómodo que hacer dos relaciones, una para local y otra para visitante.

La entidad GOL es débil porque necesita la clave de otra entidad, en este caso es JUGADOR y PARTIDO, ya que son estos los que marcan y suman los goles al contador.

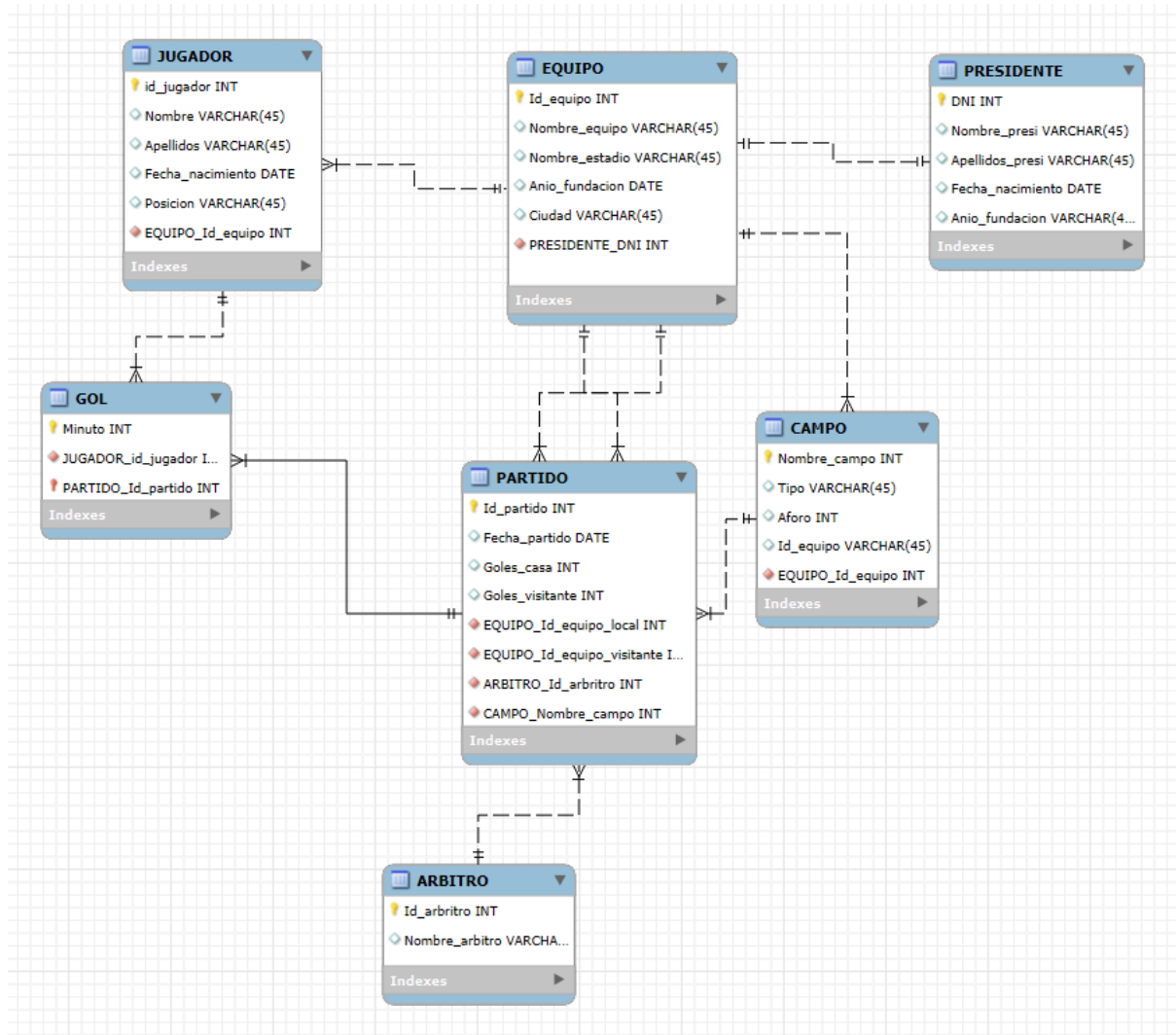
En la entidad CAMPO hay una herencia, esa herencia se refiere a que cada equipo tiene un campo donde se juegan los partidos (estadio) y suele tener uno o varios más donde entrenan para esos partidos (entrenamiento), y esos campos suelen estar ligados a estos equipos o clubes.

## Modelo relacional

Aquí tenemos el modelo relacional:



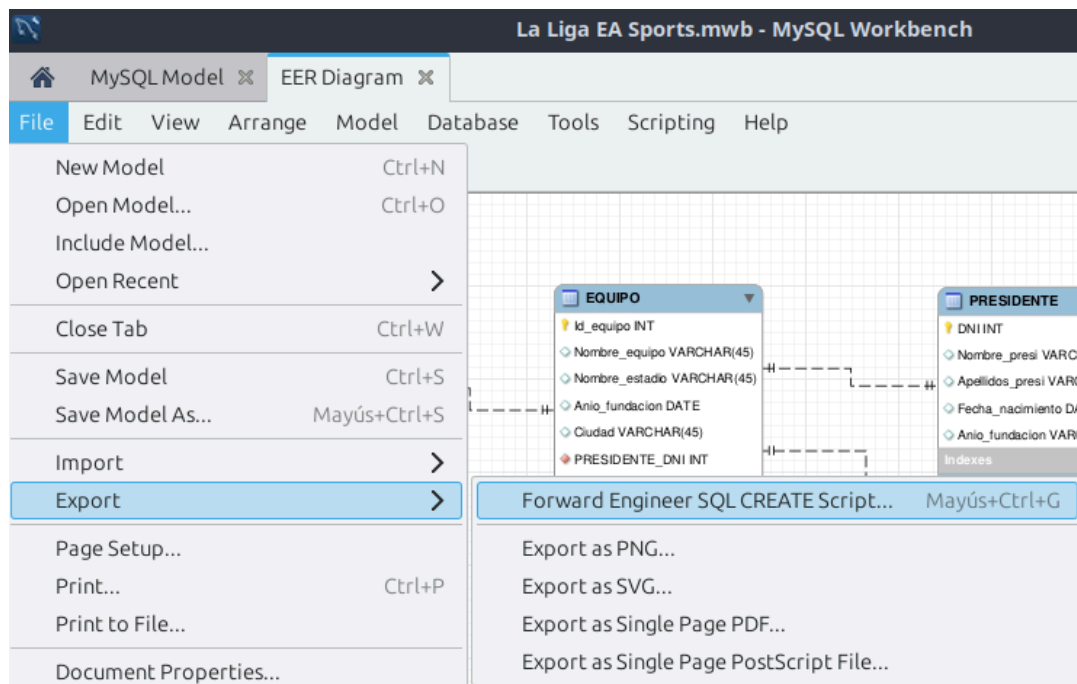
Hasta ahora hemos visto el resultado de draw.io, pero vamos a ver como quedarían estas tablas en mysql workbench:



Aquí podemos apreciar mejor las relaciones y las claves hay entre cada tabla.

## Ejecutar Script

Para pasar el script de workbench a dbeaver he exportado el archivo, y lo he copiado y pegado en dbeaver directamente:

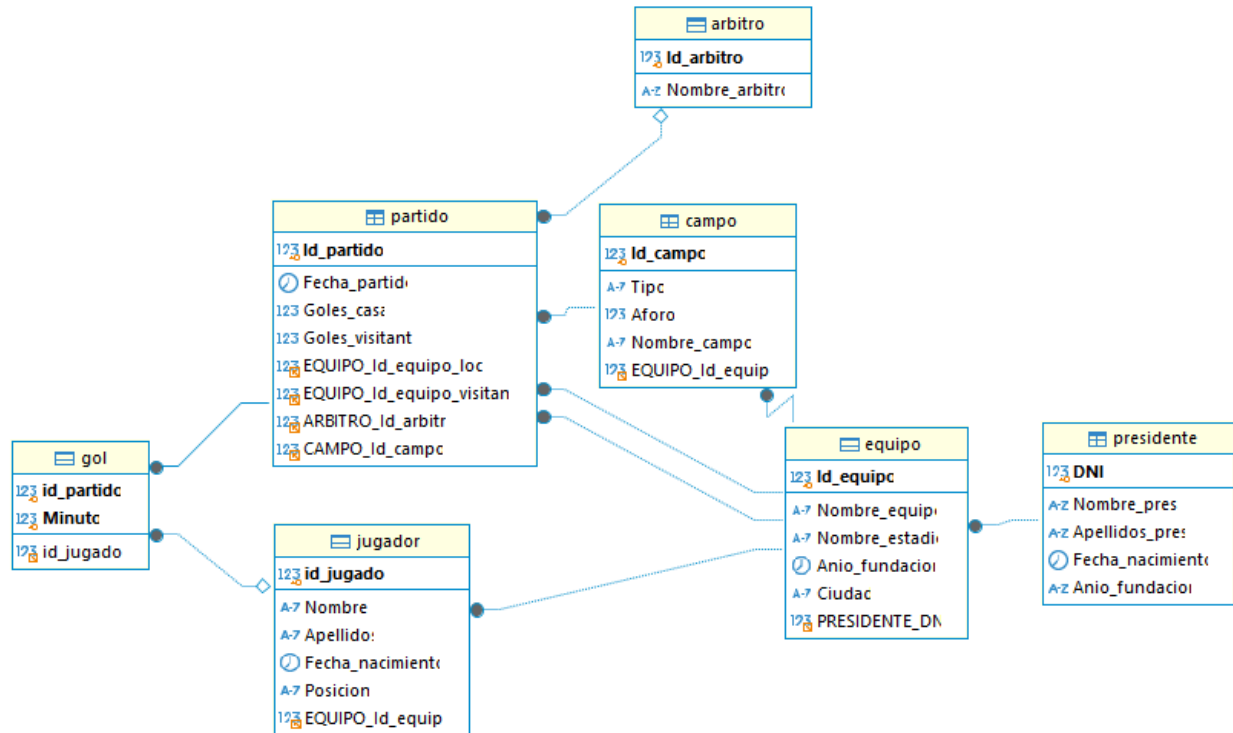


Al hacer esto, se nos genera un archivo con el Script dentro, sólo tendremos que copiar y pegar en un nuevo Script de Dbeaver:

```
-- Table `mydb`.`PRESIDENTE`
-----
CREATE TABLE IF NOT EXISTS `mydb`.`PRESIDENTE` (
  `DNI` INT NOT NULL,
  `Nombre_presi` VARCHAR(45) NULL,
  `Apellidos_presi` VARCHAR(45) NULL,
  `Fecha_nacimiento` DATE NULL,
  `Anio_fundacion` VARCHAR(45) NULL,
  PRIMARY KEY (`DNI`))
ENGINE = InnoDB;

-- Table `mydb`.`EQUIPO`
-----
CREATE TABLE IF NOT EXISTS `mydb`.`EQUIPO` (
  `Id_equipo` INT NOT NULL,
  `Nombre_equipo` VARCHAR(45) NULL,
  `Nombre_estadio` VARCHAR(45) NULL,
  `Anio_fundacion` DATE NULL,
  `Ciudad` VARCHAR(45) NULL,
  `PRESIDENTE_DNI` INT NOT NULL,
  PRIMARY KEY (`Id_equipo`),
  INDEX `fk_EQUIPO_PRESIDENTE1_idx` (`PRESIDENTE_DNI` ASC) VISIBLE,
  CONSTRAINT `fk_EQUIPO_PRESIDENTE1`
    FOREIGN KEY (`PRESIDENTE_DNI`)
      REFERENCES `mydb`.`PRESIDENTE` (`DNI`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

Al generar el Script en Dbeaver, si no hay ningún fallo, podremos refrescar la base de datos y nos saldrá:





## Carga masiva de datos

Una vez que tenemos nuestra base de datos lista, podemos proceder a hacer la carga de datos, en mi caso he usado mockaroo.

Este es el ejemplo de la tabla PRESIDENTE:

The screenshot shows the Mockaroo website interface. At the top, there's a navigation bar with links: SCHEMAS, DATASETS, APIS, DATABASES (highlighted with a 'NEW' badge), SCENARIOS, PROJECTS, and FUNCTIONS. Below the navigation bar, there's a promotional banner for TON. The main area is a configuration form for generating fake data. It has a table with columns: Field Name, Type, and Options. The fields are: DNI (Row Number), Nombre\_presi (First Name), Apellidos\_presi (Last Name), Fecha\_nacimiento (Datetime), and Anio\_fundacion (Mobile Device Relea...). Each field has a dropdown for Type and a set of options for blank percentage and a sum icon. At the bottom, there's a section for # Rows (500), Format (SQL), Table Name (PRESIDENTE), and a checkbox for 'include CREATE TABLE'.

Field Name	Type	Options
DNI	Row Number	blank: 0 %
Nombre_presi	First Name	blank: 0 %
Apellidos_presi	Last Name	blank: 0 %
Fecha_nacimiento	Datetime	01/21/2024 to 01/21/2025 format: yyyy-mm-dd blank: 0 %
Anio_fundacion	Mobile Device Relea...	blank: 0 %

+ ADD ANOTHER FIELD GENERATE FIELDS USING AI...

# Rows: 500 Format: SQL Table Name: PRESIDENTE ☐ include CREATE TABLE

Cuando tengo los atributos listos, genero el script sql y copio el contenido y lo pego en un Script de Dbeaver:

The screenshot shows a Dbeaver SQL script editor window. The title bar says 'mydb' and the script is named '\*<localhost> Script-1'. The script contains 15 'insert into PRESIDENTE' statements, each with a set of values in parentheses. The values are: (DNI, Nombre\_presi, Apellidos\_presi, Fecha\_nacimiento, Anio\_fundacion). The script is as follows:

```
insert into PRESIDENTE (DNI, Nombre_presi, Apellidos_presi, Fecha_nacimiento, Anio_fundacion)
insert into PRESIDENTE (DNI, Nombre_presi, Apellidos_presi, Fecha_nacimiento, Anio_fundacion)
insert into PRESIDENTE (DNI, Nombre_presi, Apellidos_presi, Fecha_nacimiento, Anio_fundacion)
insert into PRESIDENTE (DNI, Nombre_presi, Apellidos_presi, Fecha_nacimiento, Anio_fundacion)
insert into PRESIDENTE (DNI, Nombre_presi, Apellidos_presi, Fecha_nacimiento, Anio_fundacion)
insert into PRESIDENTE (DNI, Nombre_presi, Apellidos_presi, Fecha_nacimiento, Anio_fundacion)
insert into PRESIDENTE (DNI, Nombre_presi, Apellidos_presi, Fecha_nacimiento, Anio_fundacion)
insert into PRESIDENTE (DNI, Nombre_presi, Apellidos_presi, Fecha_nacimiento, Anio_fundacion)
insert into PRESIDENTE (DNI, Nombre_presi, Apellidos_presi, Fecha_nacimiento, Anio_fundacion)
insert into PRESIDENTE (DNI, Nombre_presi, Apellidos_presi, Fecha_nacimiento, Anio_fundacion)
insert into PRESIDENTE (DNI, Nombre_presi, Apellidos_presi, Fecha_nacimiento, Anio_fundacion)
insert into PRESIDENTE (DNI, Nombre_presi, Apellidos_presi, Fecha_nacimiento, Anio_fundacion)
insert into PRESIDENTE (DNI, Nombre_presi, Apellidos_presi, Fecha_nacimiento, Anio_fundacion)
insert into PRESIDENTE (DNI, Nombre_presi, Apellidos_presi, Fecha_nacimiento, Anio_fundacion)
insert into PRESIDENTE (DNI, Nombre_presi, Apellidos_presi, Fecha_nacimiento, Anio_fundacion)
```

Ahora solo tendremos que ejecutar el script y nos saldrá la tabla llena:

mydb \*<localhost> Script-1 PRESIDENTE X

Propiedades Datos Diagrama ER

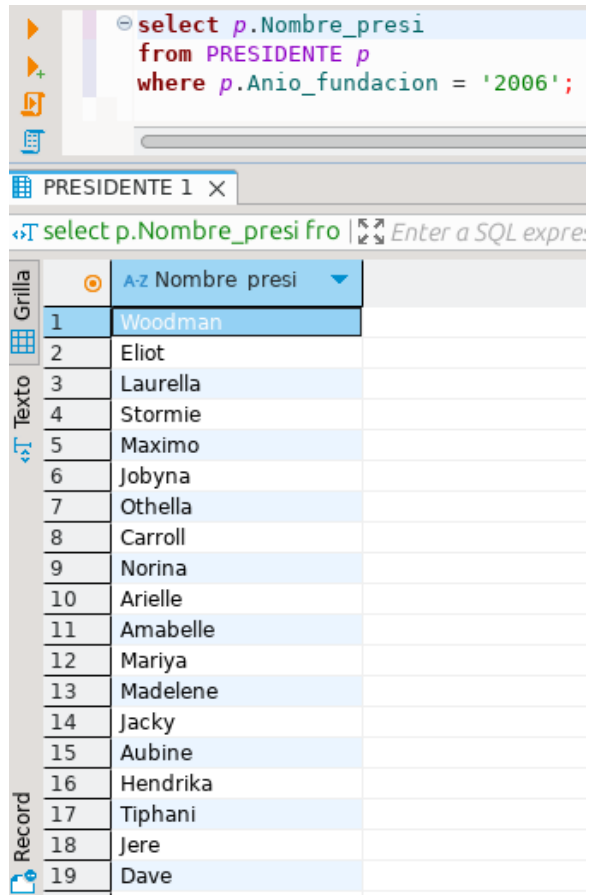
localhost Databases mydb Tables PRESIDENTE

PRESIDENTE Enter a SQL expression to filter results (use Ctrl+Space)

	123 DNI	A-Z Nombre presi	A-Z Apellidos presi	Fecha nacimiento	A-Z Anio fundacion
481	481	Kara-lynn	Dikes	2024-08-31	2004
482	482	Dana	Musson	2024-01-25	2008
483	483	Emiline	Tumpane	2024-03-30	2011
484	484	Sibilla	Twiddy	2024-03-24	2019
485	485	Ware	Hasnip	2024-01-23	2014
486	486	Tucker	Chamberlaine	2024-05-08	2013
487	487	Davin	MacCarter	2024-07-16	2005
488	488	Shelby	Sabattier	2024-04-15	2012
489	489	Lari	Campe	2024-09-24	2010
490	490	Philippine	Henriksson	2024-02-07	2013
491	491	Emeline	Boyett	2024-11-22	2011
492	492	Laverna	Schaffel	2024-02-26	2018
493	493	Kanya	Cyseley	2024-07-04	2002
494	494	Traver	Bravey	2024-02-13	2005
495	495	Bordie	Cloke	2024-04-21	2016
496	496	Bren	Beatty	2024-05-08	2018
497	497	Elsie	Cristou	2024-03-06	2018
498	498	Dulcea	McKevin	2024-05-21	2011
499	499	Latrena	Kittle	2025-01-16	2014
500	500	Frasier	Mapston	2024-10-07	2012

## Consultas

1ª Consulta Una tabla con where:



The screenshot shows a database query tool interface. At the top, a SQL query is entered in a text area: `select p.Nombre_presi  
from PRESIDENTE p  
where p.Anio_fundacion = '2006';`. Below the query area, a tab labeled 'PRESIDENTE 1' is active. The results are displayed in a table with a sidebar on the left containing icons for 'Grilla' (Grid), 'Texto' (Text), and 'Record'. The 'Grilla' icon is selected. The table has two columns: 'A-Z Nombre presi' and an empty column. The results are as follows:

	A-Z Nombre presi	
1	Woodman	
2	Eliot	
3	Laurella	
4	Stormie	
5	Maximo	
6	Jobyna	
7	Othella	
8	Carroll	
9	Norina	
10	Arielle	
11	Amabelle	
12	Mariya	
13	Madelene	
14	Jacky	
15	Aubine	
16	Hendrika	
17	Tiphani	
18	Jere	
19	Dave	

Nombre de los presidentes cuyo equipo ha sido fundado en 2006.

## 2ª Consulta Más de una tabla:

```
SELECT a.Nombre_arbitro, p.Id_partido
from ARBITRO a inner join PARTIDO p
on a.Id_arbitro = p.ARBITRO_Id_arbitro;
```

ARBITRO(+) 1 X

SELECT a.Nombre\_arbitro, Enter a SQL expression to filter i

	A-Z Nombre arbitro	123 Id partido
1	Karoly	1
2	Carmita	2
3	Verla	3
4	Mercy	4
5	Gerrilee	5
6	Melony	6
7	Jeralee	7
8	Nils	8
9	Faina	9
10	Neville	10
11	Emelita	11
12	Sharleen	12
13	Fredi	13
14	Drusilla	14

Nombre de los árbitros y los id de los partidos.

## 3ª Consulta con agrupación:

```
SELECT COUNT(j.Nombre) , e.Nombre_equipo
FROM JUGADOR j inner join EQUIPO e
on j.EQUIPO_Id_equipo = e.Id_equipo
group by e.Nombre_equipo;
```

EQUIPO 1 X

SELECT COUNT(j.Nombre) Enter a SQL expression to filter

	123 COUNT(j.Nombre)	A-Z Nombre equipo
1	3	Interessant
2	6	Judicaël
3	1	Liè
4	2	Léa
5	2	Örjan
6	5	Gisèle
7	5	Andrée

Los equipos y sus jugadores.

4ª Consulta con subconsultas:

```
SELECT e.Nombre_equipo, e.Id_equipo
FROM EQUIPO e
where e.Id_equipo in (
  SELECT MAX(e.Id_equipo)
  from EQUIPO e
);
```

EQUIPO 1 X

	A-Z Nombre equipo	123 Id equipo
1	Maílís	500

Nombre del equipo con el id más alto.

5ª Consulta combina varias anteriores:

```
SELECT p.Nombre_presi, e.Id_equipo
FROM PRESIDENTE p inner join EQUIPO e
  on p.DNI = e.PRESIDENTE_DNI
where e.Id_equipo in (
  SELECT MAX(e.Id_equipo)
  from EQUIPO e
);
```

PRESIDENTE(+) 1 X

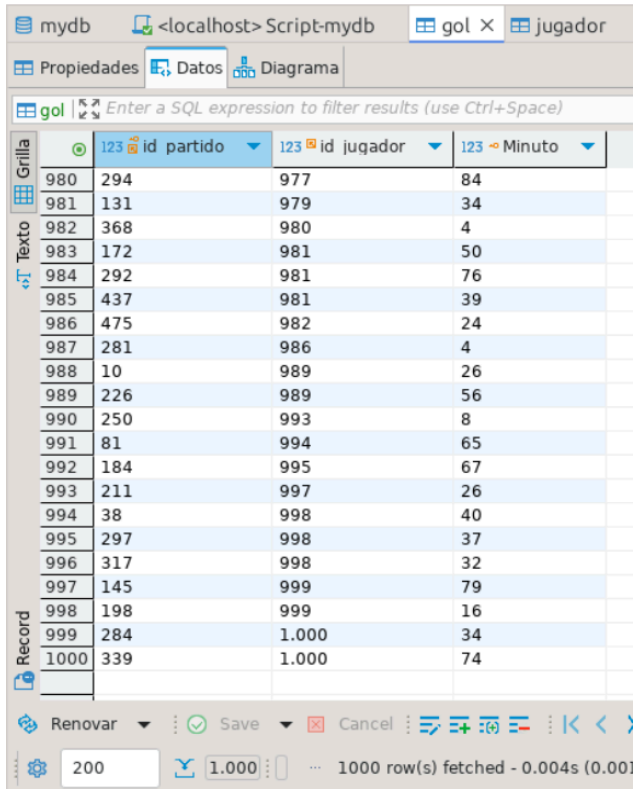
	A-Z Nombre presi	123 Id equipo
1	Frasier	500

Nombre del presidente del equipo con mayor id, con dos tablas y subconsulta.

2ª entrega

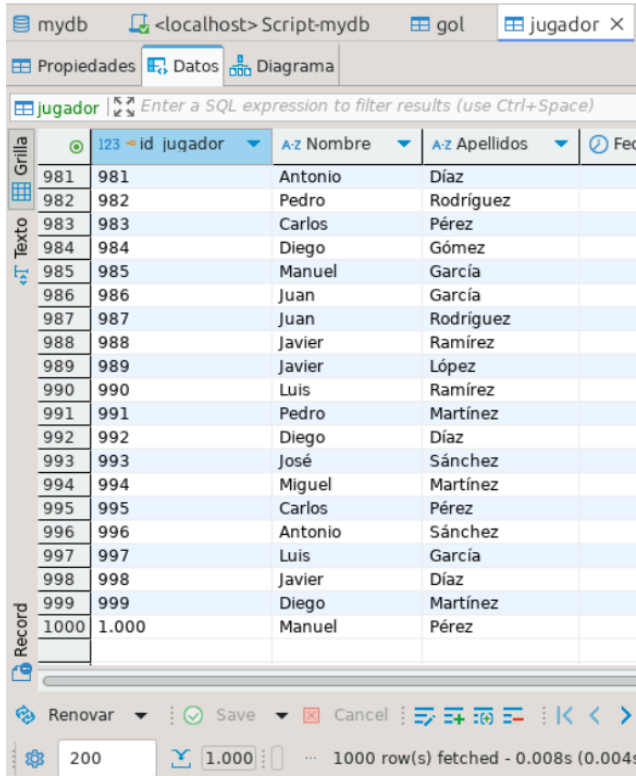
Carga masiva en dos tablas con 1000 datos

Gol:



	123 id partido	123 id jugador	123 Minuto
980	294	977	84
981	131	979	34
982	368	980	4
983	172	981	50
984	292	981	76
985	437	981	39
986	475	982	24
987	281	986	4
988	10	989	26
989	226	989	56
990	250	993	8
991	81	994	65
992	184	995	67
993	211	997	26
994	38	998	40
995	297	998	37
996	317	998	32
997	145	999	79
998	198	999	16
999	284	1.000	34
1000	339	1.000	74

Jugador:

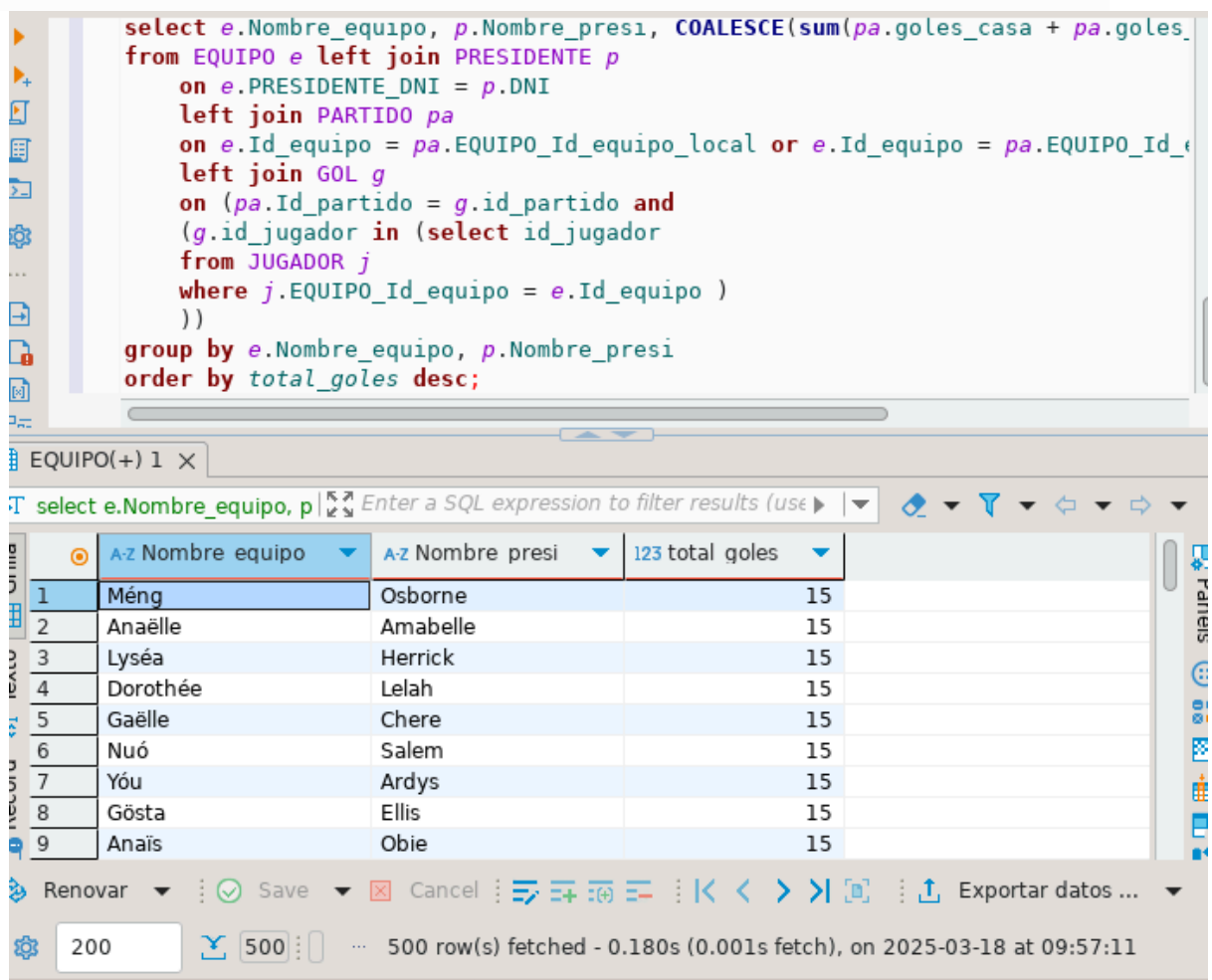


	123 id jugador	A-Z Nombre	A-Z Apellidos	Fecha
981	981	Antonio	Díaz	
982	982	Pedro	Rodríguez	
983	983	Carlos	Pérez	
984	984	Diego	Gómez	
985	985	Manuel	García	
986	986	Juan	García	
987	987	Juan	Rodríguez	
988	988	Javier	Ramírez	
989	989	Javier	López	
990	990	Luis	Ramírez	
991	991	Pedro	Martínez	
992	992	Diego	Díaz	
993	993	José	Sánchez	
994	994	Miguel	Martínez	
995	995	Carlos	Pérez	
996	996	Antonio	Sánchez	
997	997	Luis	García	
998	998	Javier	Díaz	
999	999	Diego	Martínez	
1000	1.000	Manuel	Pérez	

## 5 consultas multitable

1. Obtener el nombre de cada equipo, el nombre de su presidente y el total de goles anotados por sus jugadores en todos los partidos:

```
select e.Nombre_equipo, p.Nombre_presi, COALESCE(sum(pa.goles_casa +
pa.goles_visitante), 0) as total_goles
from EQUIPO e left join PRESIDENTE p
on e.PRESIDENTE_DNI = p.DNI
left join PARTIDO pa
on e.Id_equipo = pa.EQUIPO_Id_equipo_local or e.Id_equipo =
pa.EQUIPO_Id_equipo_visitante
left join GOL g
on (pa.Id_partido = g.id_partido and
(g.id_jugador in (select id_jugador
from JUGADOR j
where j.EQUIPO_Id_equipo = e.Id_equipo )
))
group by e.Nombre_equipo, p.Nombre_presi
order by total_goles desc;
```



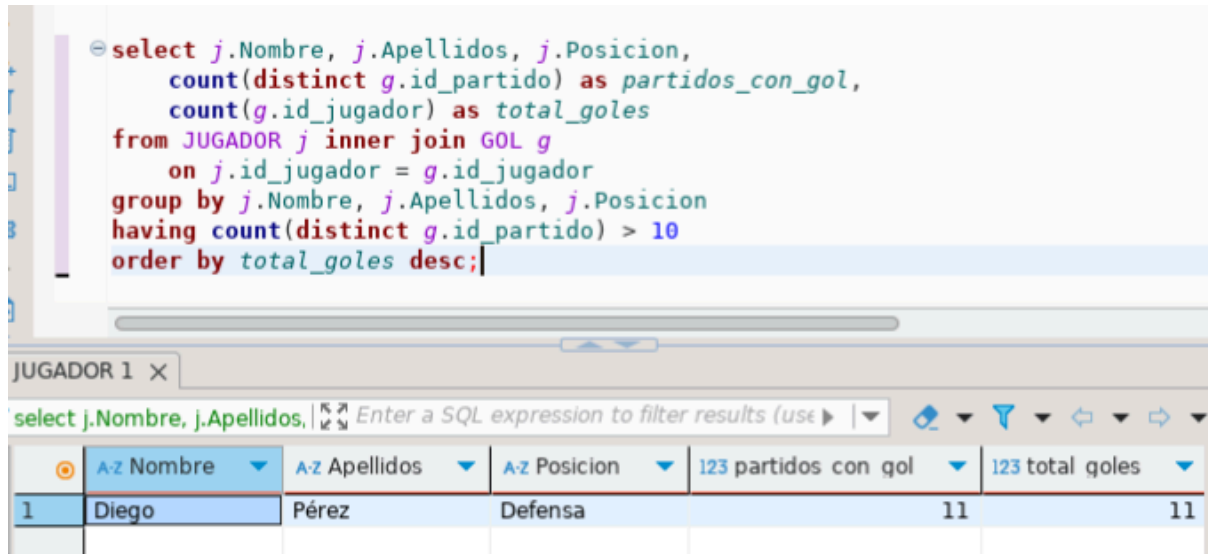
The screenshot shows a SQL IDE interface. The top pane displays the SQL query, which is identical to the one in the previous block. The bottom pane shows the results of the query in a table format. The table has three columns: 'A-Z Nombre equipo', 'A-Z Nombre presi', and '123 total goles'. There are 9 rows of data, each representing a team and its president, with a total of 15 goals for each team.

	A-Z Nombre equipo	A-Z Nombre presi	123 total goles
1	Méng	Osborne	15
2	Anaëlle	Amabelle	15
3	Lyséa	Herrick	15
4	Dorothée	Lelah	15
5	Gaëlle	Chere	15
6	Nuó	Salem	15
7	Yóu	Ardys	15
8	Gösta	Ellis	15
9	Anaïs	Obie	15

At the bottom of the IDE, there is a status bar showing '500 row(s) fetched - 0.180s (0.001s fetch), on 2025-03-18 at 09:57:11'.

2. Jugadores que han anotado al menos un gol en más de 10 partidos diferentes, con su nombre, posición y el total de goles anotados:

```
select j.Nombre, j.Apellidos, j.Posicion,  
       count(distinct g.id_partido) as partidos_con_gol,  
       count(g.id_jugador) as total_goles  
from JUGADOR j inner join GOL g  
     on j.id_jugador = g.id_jugador  
group by j.Nombre, j.Apellidos, j.Posicion  
having count(distinct g.id_partido) > 10  
order by total_goles desc;
```



The screenshot shows a SQL IDE interface. The top pane displays the SQL query from the previous block. The bottom pane shows the results of the query in a table format. The table has five columns: an index, Name, Surname, Position, and Total Goals. The first row shows a player named Diego Pérez in the Defense position with 11 total goals. The table is titled 'JUGADOR 1'.

	A-Z Nombre	A-Z Apellidos	A-Z Posicion	123 partidos con gol	123 total goles
1	Diego	Pérez	Defensa	11	11



3. Árbitros y el número de partidos dirigidos en campos con un nombre en específico, junto con el total de goles en esos partidos:

```
Select a.Nombre_arbitro, COUNT(pa.id_partido) as partidos_dirigidos,  
SUM(COALESCE(pa.goles_casa, 0) + COALESCE(pa.goles_visitante, 0)) as  
total_goles  
from ARBITRO a inner join PARTIDO pa  
on a.id_arbitro = pa.ARBITRO_id_arbitro  
inner join CAMPO c  
on pa.CAMPO_Id_campo = c.Id_campo  
left join GOL g  
on pa.id_partido = g.id_partido  
where  
c.Nombre_campo = 'Lorène'  
group by  
a.Nombre_arbitro  
having  
COUNT(pa.id_partido) > 0  
order by  
partidos_dirigidos DESC;
```

The screenshot shows a SQL query editor with the following query:

```
Select a.Nombre_arbitro, COUNT(pa.id_partido) as partidos_dirigidos,  
SUM(COALESCE(pa.goles_casa, 0) + COALESCE(pa.goles_visitante, 0)) as total_goles  
from ARBITRO a inner join PARTIDO pa  
on a.id_arbitro = pa.ARBITRO_id_arbitro  
inner join CAMPO c  
on pa.CAMPO_Id_campo = c.Id_campo  
left join GOL g  
on pa.id_partido = g.id_partido  
where  
c.Nombre_campo = 'Lorène'  
group by  
a.Nombre_arbitro  
having  
COUNT(pa.id_partido) > 0  
order by  
partidos_dirigidos DESC;
```

Below the query editor, the results are displayed in a table with the following columns: "A-z Nombre arbitro", "123 partidos dirigidos", and "123 total goles". The table contains three rows of data:

	A-z Nombre arbitro	123 partidos dirigidos	123 total goles
1	Hazlett	2	22
2	Tonie	2	14
3	Faina	1	6

At the bottom of the interface, there is a status bar indicating "3 row(s) fetched - 0.004s, on 2025-03-18 at 10:55:58".

4. Equipos con el total de victorias como local, incluyendo sólo aquellos que tengan más de 3:

```
select e.Nombre_equipo, count(*) as victorias_local
from EQUIPO e inner join PARTIDO p
  on e.id_equipo = p.EQUIPO_id_equipo_local
where p.id_partido in (
  select id_partido
  from PARTIDO p
  where p.Goles_casa > p.Goles_visitante
)
group by e.Nombre_equipo
having victorias_local > 3
order by victorias_local desc;
```

The screenshot shows a SQL query editor with the following query:

```
select e.Nombre_equipo, count(*) as victorias_local
from EQUIPO e inner join PARTIDO p
  on e.id_equipo = p.EQUIPO_id_equipo_local
where p.id_partido in (
  select id_partido
  from PARTIDO p
  where p.Goles_casa > p.Goles_visitante
)
group by e.Nombre_equipo
having victorias_local > 3
order by victorias_local desc;
```

Below the query editor, the results are displayed in a table. The table has two columns: 'A-Z Nombre equipo' and '123 victorias local'. The results are as follows:

	A-Z Nombre equipo	123 victorias local
1	Gisèle	5
2	Judicaël	5
3	Rébecca	5
4	Anaëlle	4
5	Pélagie	4
6	Andréanne	4
7	Märta	4
8	Véronique	4

The interface also shows a status bar at the bottom indicating '19 row(s) fetched - 0.020s (0.001s fetch), on 2025-03-18 at 18:28:00'.

5. Campos donde se jugaron partidos con un total de goles superior al promedio general:

```
select c.Nombre_campo, AVG(p.Goles_casa + p.Goles_visitante) as promedio_goles
from CAMPO c inner join PARTIDO p
    on c.id_campo = p.CAMPO_id_campo
group by c.Nombre_campo
having AVG(p.Goles_casa + p.Goles_visitante) > (
    select avg(p.Goles_casa + p.Goles_visitante)
    from PARTIDO p
)
order by promedio_goles desc;
```

The screenshot shows a SQL IDE interface. The top pane displays the SQL query from the previous block. The bottom pane shows the results of the query in a table. The table has two columns: 'Nombre campo' and 'promedio\_goles'. The results are sorted in descending order of the average goals. The first row is 'Angélique' with a value of 14. The last row is 'Crééz' with a value of 12,6667. The interface also shows a status bar at the bottom indicating '108 row(s) fetched'.

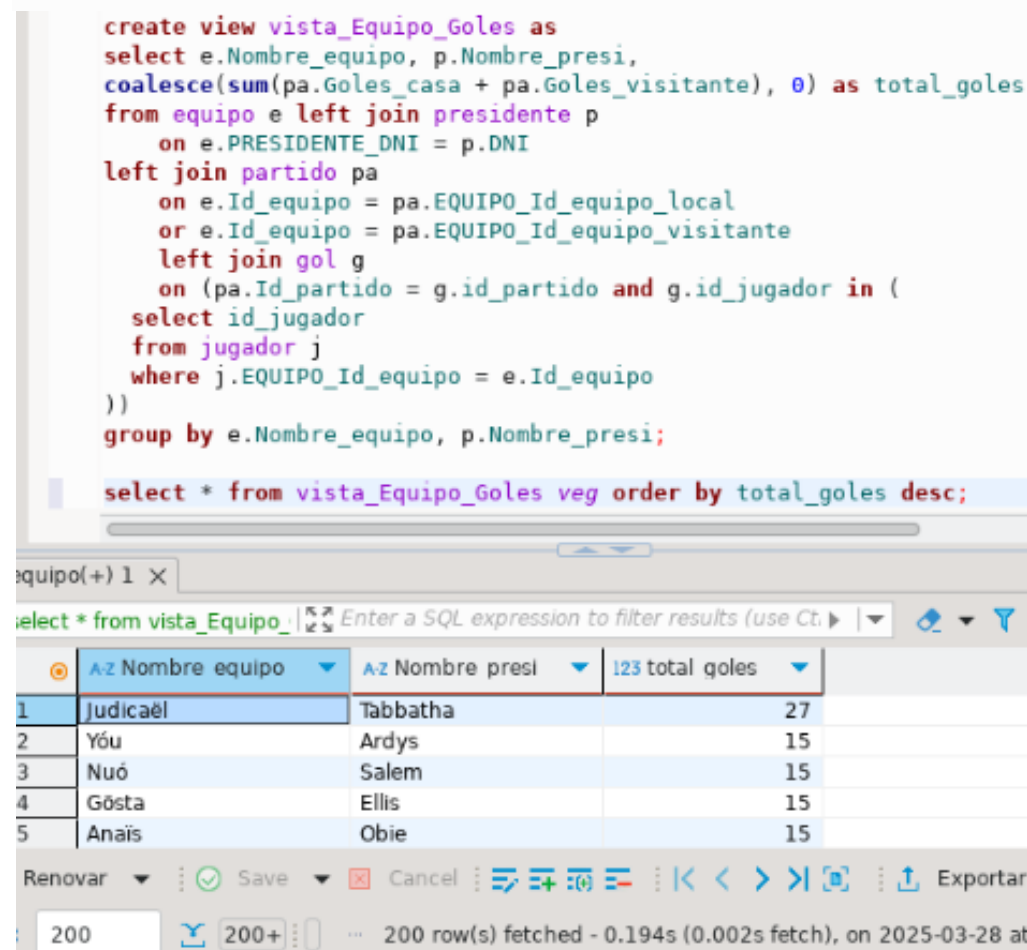
	A:Z Nombre campo	promedio_goles
1	Angélique	14
2	Gaïa	14
3	Pélagie	14
4	Personnalisée	14
5	Cléa	13
6	Intéressant	13
7	Styrbjörn	13
8	Liè	13
9	Crééz	12,6667

## Vistas

### 1. Equipos con nombre, presidente y total de goles anotados:

Esta vista muestra el nombre de cada equipo, el nombre de su presidente y el total de goles anotados por sus jugadores en los partidos. Es útil para evaluar el rendimiento de los partidos.

```
create view vista_Equipo_Goles as
select e.Nombre_equipo, p.Nombre_presi,
coalesce(sum(pa.Goles_casa + pa.Goles_visitante), 0) as total_goles
from EQUIPO e left join PRESIDENTE p
on e.PRESIDENTE_DNI = p.DNI
left join PARTIDO pa
on e.Id_equipo = pa.EQUIPO_Id_equipo_local
or e.Id_equipo = pa.EQUIPO_Id_equipo_visitante
left join GOL g
on (pa.Id_partido = g.id_partido and g.id_jugador in (
select id_jugador
from JUGADOR j
where j.EQUIPO_Id_equipo = e.Id_equipo
))
group by e.Nombre_equipo, p.Nombre_presi;
```



The screenshot shows a SQL IDE interface. The top pane displays the SQL code for creating the view 'vista\_Equipo\_Goles' and a query to select all data from it, ordered by total goals in descending order. The bottom pane shows the results of the query in a table format.

```
create view vista_Equipo_Goles as
select e.Nombre_equipo, p.Nombre_presi,
coalesce(sum(pa.Goles_casa + pa.Goles_visitante), 0) as total_goles
from equipo e left join presidente p
on e.PRESIDENTE_DNI = p.DNI
left join partido pa
on e.Id_equipo = pa.EQUIPO_Id_equipo_local
or e.Id_equipo = pa.EQUIPO_Id_equipo_visitante
left join gol g
on (pa.Id_partido = g.id_partido and g.id_jugador in (
select id_jugador
from jugador j
where j.EQUIPO_Id_equipo = e.Id_equipo
))
group by e.Nombre_equipo, p.Nombre_presi;

select * from vista_Equipo_Goles veg order by total_goles desc;
```

	A-Z Nombre equipo	A-Z Nombre presi	total goles
1	Judicaël	Tabbatha	27
2	Yóu	Ardys	15
3	Nuó	Salem	15
4	Gôsta	Ellis	15
5	Anaïs	Obie	15

At the bottom of the IDE, there is a status bar showing '200 row(s) fetched - 0.194s (0.002s fetch), on 2025-03-28 at'.

## 2. Campo con promedio de goles superior al general

Esta vista identifica los campos donde el promedio de goles por partido supera al promedio general de todos los partidos. Es útil para analizar qué campos tienen más actividad goleadora.

```
create view vista_Campos_Promedio_Goles_Superior as
select c.Nombre_campo,
       avg(p.Goles_casa + p.Goles_visitante) as promedio_goles
from CAMPO c inner join PARTIDO p
  on c.id_campo = p.CAMPO_id_campo
group by c.Nombre_campo
having avg(p.Goles_casa + p.Goles_visitante) > (
  select avg(p.Goles_casa + p.Goles_visitante)
  from PARTIDO p
);
```

The screenshot shows a SQL IDE interface. The top pane displays the SQL code for creating a view and querying it. The bottom pane shows the results of the query in a table format.

```
create view vista_Campos_Promedio_Goles_Superior as
select c.Nombre_campo,
       avg(p.Goles_casa + p.Goles_visitante) as promedio_goles
from CAMPO c inner join PARTIDO p
  on c.id_campo = p.CAMPO_id_campo
group by c.Nombre_campo
having avg(p.Goles_casa + p.Goles_visitante) > (
  select avg(p.Goles_casa + p.Goles_visitante)
  from PARTIDO p
);

select *
from vista_Campos_Promedio_Goles_Superior vcpgs order by promedio_goles desc;
```

The results table shows the following data:

	A-Z Nombre campo	promedio goles
1	Angélique	14
2	Gaïa	14
3	Pélagie	14
4	Personnalisée	14
5	Cléa	13
6	Intéressant	13

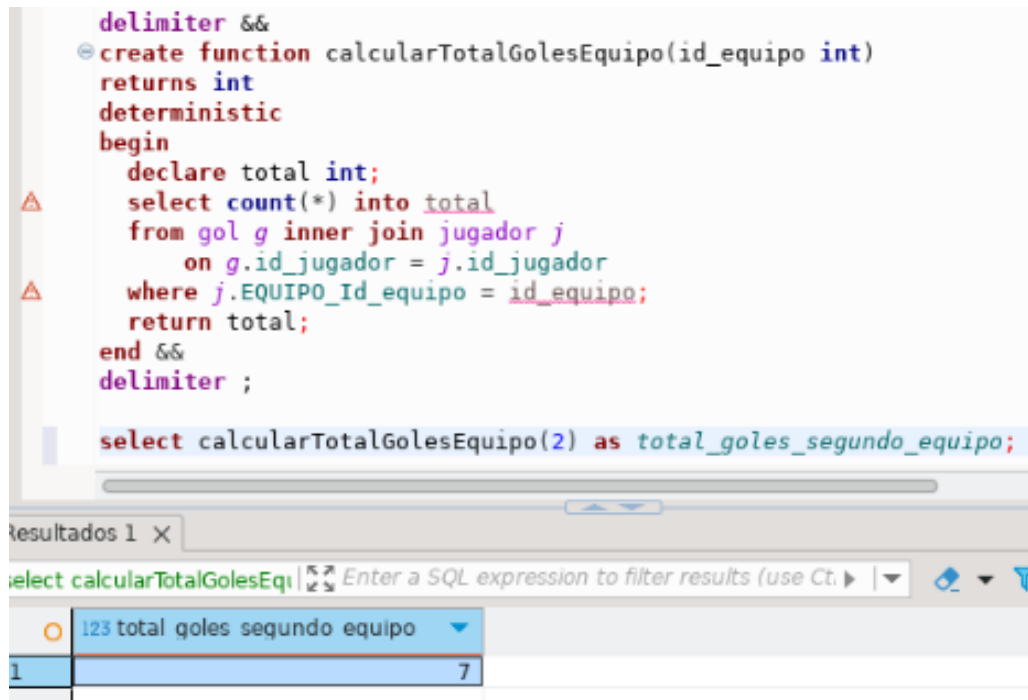
The interface also shows a status bar at the bottom indicating "108 row(s) fetched - 0.021s (0.001s fetch), on 2025-03-18 at 20:22:53".

## Funciones

### 1. calcularTotalGolesEquipo

Calcula el total de goles anotados por los jugadores de un equipo en todos los partidos.

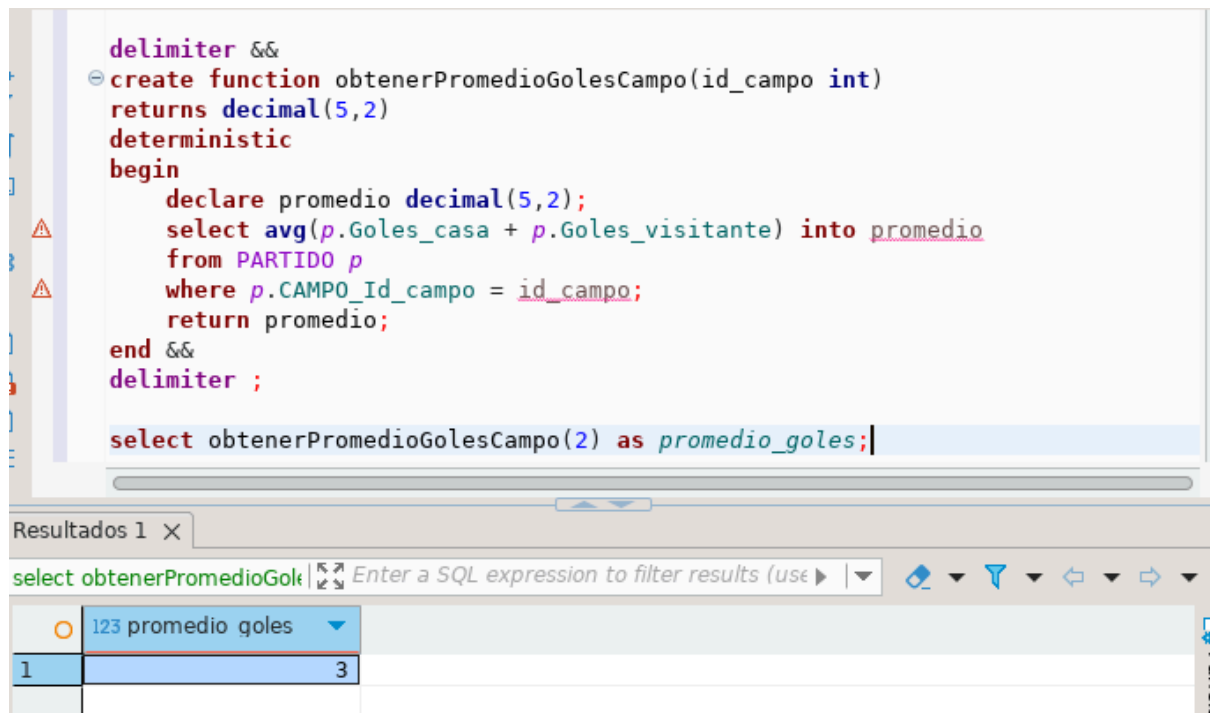
```
delimiter &&
create function calcularTotalGolesEquipo(id_equipo int)
returns int
deterministic
begin
    declare total int;
    select count(g.id_gol) into total
    from GOL g inner join JUGADOR j
        on g.id_jugador = j.id_jugador
    where j.EQUIPO_Id_equipo = id_equipo;
    return total;
end &&
delimiter ;
```



## 2. obtenerPromedioGolesCampo

Calcula el promedio de goles por partido en un campo específico.

```
delimiter &&
create function obtenerPromedioGolesCampo(id_campo int)
returns decimal(5,2)
deterministic
begin
    declare promedio decimal(5,2);
    select avg(p.Goles_casa + p.Goles_visitante) into promedio
    from PARTIDO p
    where p.CAMPO_Id_campo = id_campo;
    return promedio;
end &&
delimiter ;
```



```
delimiter &&
create function obtenerPromedioGolesCampo(id_campo int)
returns decimal(5,2)
deterministic
begin
    declare promedio decimal(5,2);
    select avg(p.Goles_casa + p.Goles_visitante) into promedio
    from PARTIDO p
    where p.CAMPO_Id_campo = id_campo;
    return promedio;
end &&
delimiter ;

select obtenerPromedioGolesCampo(2) as promedio_goles;
```

Resultados 1 x

select obtenerPromedioGolesCampo(2) as promedio\_goles

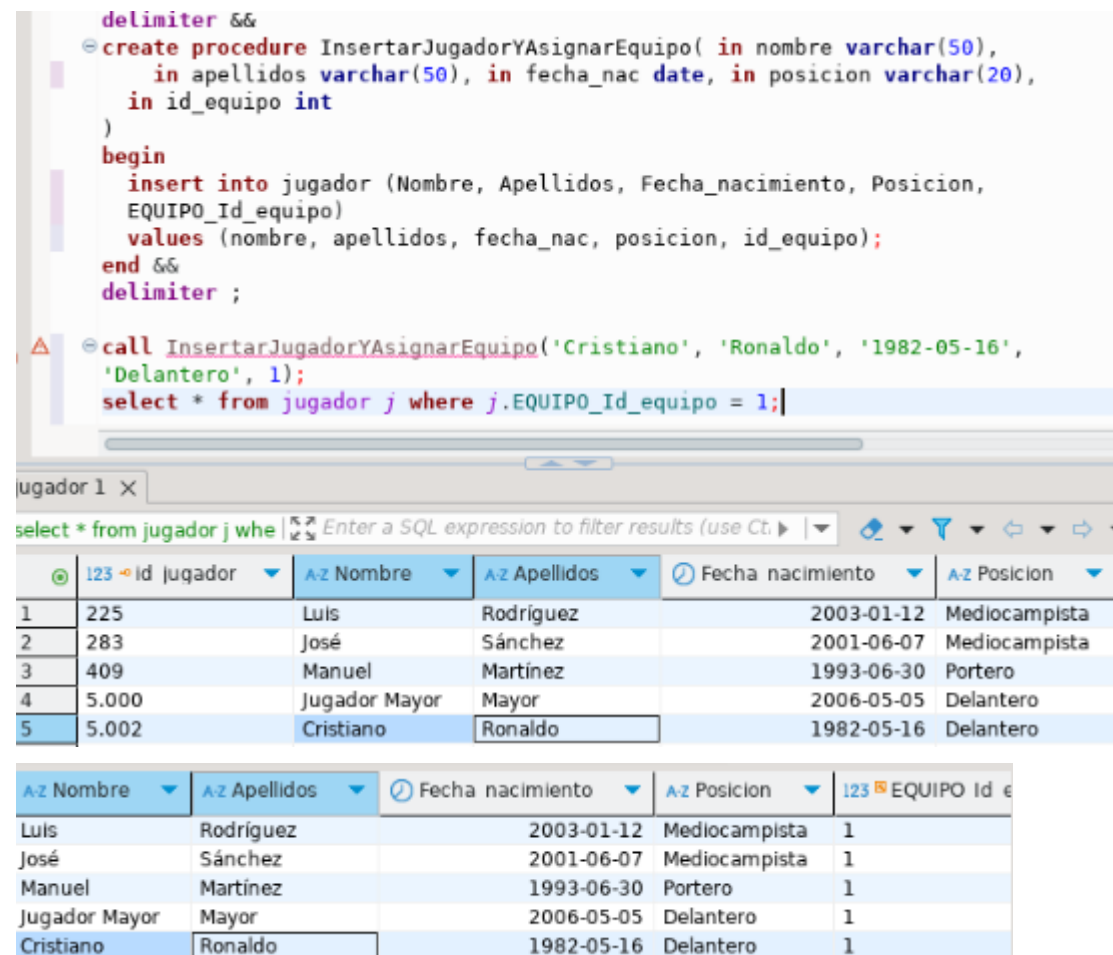
	123 promedio goles
1	3

## Procedimientos

### 1. insertarJugadorYAsignarEquipo

Insertar un nuevo jugador en la tabla JUGADOR.

```
delimiter &&
create procedure InsertarJugadorYAsignarEquipo( in nombre varchar(50),
    in apellidos varchar(50), in fecha_nac date, in posicion varchar(20),
    in id_equipo int
)
begin
    insert into jugador (Nombre, Apellidos, Fecha_nacimiento, Posicion,
    EQUIPO_Id_equipo)
    values (nombre, apellidos, fecha_nac, posicion, id_equipo);
end &&
delimiter ;
```



The screenshot shows a SQL IDE with the following SQL code executed:

```
delimiter &&
create procedure InsertarJugadorYAsignarEquipo( in nombre varchar(50),
    in apellidos varchar(50), in fecha_nac date, in posicion varchar(20),
    in id_equipo int
)
begin
    insert into jugador (Nombre, Apellidos, Fecha_nacimiento, Posicion,
    EQUIPO_Id_equipo)
    values (nombre, apellidos, fecha_nac, posicion, id_equipo);
end &&
delimiter ;

call InsertarJugadorYAsignarEquipo('Cristiano', 'Ronaldo', '1982-05-16',
'Delantero', 1);
select * from jugador j where j.EQUIPO_Id_equipo = 1;
```

Below the code, the results of the query are displayed in a table. The table has 5 columns: Id jugador, Nombre, Apellidos, Fecha nacimiento, and Posicion. The data is as follows:

Id jugador	Nombre	Apellidos	Fecha nacimiento	Posicion
225	Luis	Rodríguez	2003-01-12	Mediocampista
283	José	Sánchez	2001-06-07	Mediocampista
409	Manuel	Martínez	1993-06-30	Portero
5.000	Jugador Mayor	Mayor	2006-05-05	Delantero
5.002	Cristiano	Ronaldo	1982-05-16	Delantero

Below the table, there is a summary table with 5 columns: Nombre, Apellidos, Fecha nacimiento, Posicion, and EQUIPO Id. The data is as follows:

Nombre	Apellidos	Fecha nacimiento	Posicion	EQUIPO Id
Luis	Rodríguez	2003-01-12	Mediocampista	1
José	Sánchez	2001-06-07	Mediocampista	1
Manuel	Martínez	1993-06-30	Portero	1
Jugador Mayor	Mayor	2006-05-05	Delantero	1
Cristiano	Ronaldo	1982-05-16	Delantero	1



## 2. mostrarGolesJugador

Verifica si un jugador ha marcado goles. Si no ha marcado, muestra un mensaje “NO HA MARCADO NINGÚN GOL”. Si ha marcado, lista los partidos y los minutos en los que anotó.

```
delimiter &&
create procedure mostrarGolesJugador(in id_jugador int)
begin
    declare total_goles int;
    select count(*) into total_goles
    from GOL g
    where g.id_jugador = id_jugador;
    if total_goles = 0 then
        select 'NO HA MARCADO NINGÚN GOL' as mensaje;
    else
        select g.id_partido, g.Minuto
        from GOL g
        where g.id_jugador = id_jugador
        order by g.id_partido , g.Minuto;
    end if;
end &&
delimiter ;
```

The screenshot shows a SQL IDE interface with two execution results. The first result, for `call mostrarGolesJugador(1);`, displays a table with 3 rows of goals scored by player 1. The second result, for `call mostrarGolesJugador(2);`, displays a single row with the message "NO HA MARCADO NINGÚN GOL".

**Execution 1: `call mostrarGolesJugador(1);`**

	123 id partido	123 Minuto
1	65	5
2	311	44
3	445	3

**Execution 2: `call mostrarGolesJugador(2);`**

	A-z mensaje
1	NO HA MARCADO NINGÚN GOL

### 3. actualizarAforoCampo

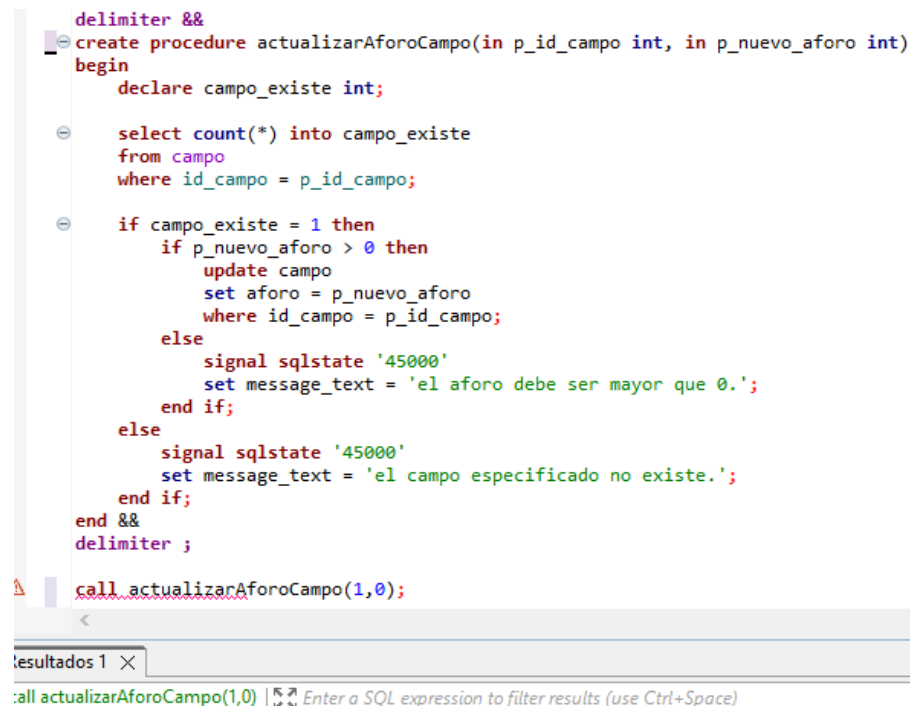
Actualiza el aforo de un campo específico. Si el campo no existe o al introducir el aforo es igual o menor que 0 da un error.

```
delimiter &&
create procedure actualizarAforoCampo(in p_id_campo int, in p_nuevo_aforo int)
begin
    declare campo_existe int;

    select count(*) into campo_existe
    from campo
    where id_campo = p_id_campo;

    if campo_existe = 1 then
        if p_nuevo_aforo > 0 then
            update campo
            set aforo = p_nuevo_aforo
            where id_campo = p_id_campo;
        else
            signal sqlstate '45000'
            set message_text = 'el aforo debe ser mayor que 0.';
        end if;
    else
        signal sqlstate '45000'
        set message_text = 'el campo especificado no existe.';
    end if;
end &&
delimiter ;
```

Si el aforo es 0 o menor:



```
delimiter &&
create procedure actualizarAforoCampo(in p_id_campo int, in p_nuevo_aforo int)
begin
    declare campo_existe int;

    select count(*) into campo_existe
    from campo
    where id_campo = p_id_campo;

    if campo_existe = 1 then
        if p_nuevo_aforo > 0 then
            update campo
            set aforo = p_nuevo_aforo
            where id_campo = p_id_campo;
        else
            signal sqlstate '45000'
            set message_text = 'el aforo debe ser mayor que 0.';
        end if;
    else
        signal sqlstate '45000'
        set message_text = 'el campo especificado no existe.';
    end if;
end &&
delimiter ;

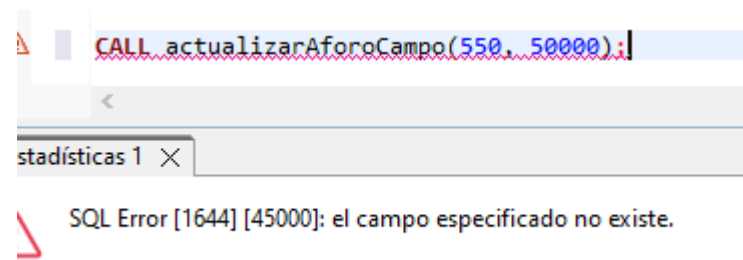
call actualizarAforoCampo(1,0);
```

Resultados 1 X

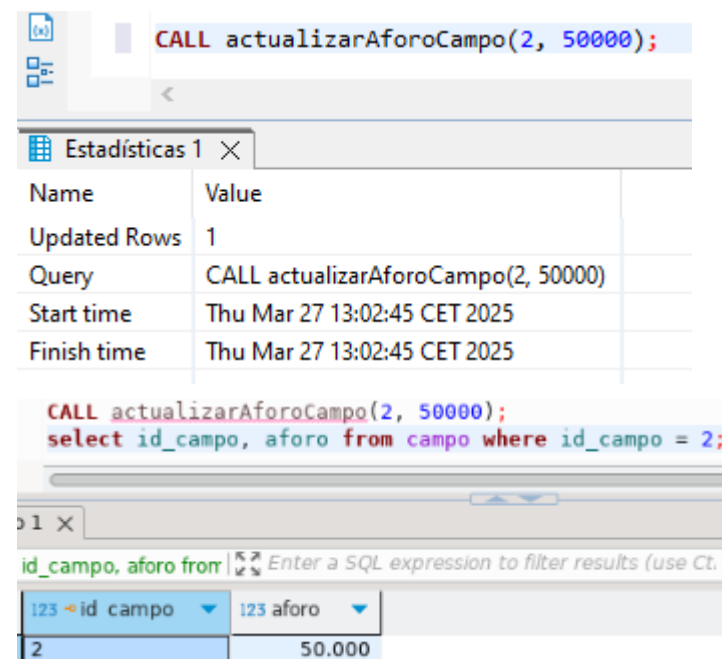
call actualizarAforoCampo(1,0) Enter a SQL expression to filter results (use Ctrl+Space)

SQL Error [1644] [45000]: el aforo debe ser mayor que 0.

Si la id del campo no existe:



Si todo está bien:



## Triggers

### 1. actualizarGolesPartidoDespuesGol

Update. Al insertar un nuevo gol en la tabla GOL, se actualiza automáticamente los goles del partido en la tabla PARTIDO.

```
create trigger actualizarGolesPartidoDespuesGol
after insert on GOL
for each row
begin
    declare equipo_jugador int;
    declare equipo_local int;
    declare equipo_visitante int;

    select EQUIPO_Id_equipo into equipo_jugador
    from JUGADOR
    where id_jugador = NEW.id_jugador;

    select EQUIPO_Id_equipo_local,
    EQUIPO_Id_equipo_visitante
    into equipo_local, equipo_visitante
    from PARTIDO
    where Id_partido = NEW.id_partido;

    if equipo_jugador = equipo_local then
        update PARTIDO
        set Goles_casa = Goles_casa + 1
        where Id_partido = NEW.id_partido;
    elseif equipo_jugador = equipo_visitante then
        update PARTIDO
        set Goles_visitante = Goles_visitante + 1
        where Id_partido = NEW.id_partido;
    end if;
end &&
delimiter ;
```

```
delimiter &&
create trigger actualizarGolesPartidoDespuesGol
after insert on GOL
for each row
begin
    declare equipo_jugador int;
    declare equipo_local int;
    declare equipo_visitante int;
    select EQUIPO_Id_equipo into equipo_jugador
    from JUGADOR
    where id_jugador = NEW.id_jugador;
    select EQUIPO_Id_equipo_local, EQUIPO_Id_equipo_visitante
    into equipo_local, equipo_visitante
    from PARTIDO
    where Id_partido = NEW.id_partido;
    if equipo_jugador = equipo_local then
        update PARTIDO
        set Goles_casa = Goles_casa + 1
        where Id_partido = NEW.id_partido;
    elseif equipo_jugador = equipo_visitante then
        update PARTIDO
        set Goles_visitante = Goles_visitante + 1
        where Id_partido = NEW.id_partido;
    end if;
end &&
delimiter ;

select Goles casa, Goles visitante from PARTIDO where Id partido = 10;
```

partido 1	123 Goles_casa	123 Goles_visitante
1	4	3

Al hacer insert de un nuevo gol en la tabla GOL y volvemos a hacer el select para el partido 10:

```
select Goles_casa, Goles_visitante from PARTIDO where Id_partido = 10;

insert into GOL (id_partido, id_jugador, Minuto)
values (10, 51, 11);
```

partido 1	123 Goles_casa	123 Goles_visitante
1	5	3

## 2. verificarEdadJugadorAntesInsertar

Insert. Verifica que el jugador a insertar tenga al menos 16 años, si no cumple esta condición, no lo inserta e inserta un mensaje de error en otra tabla.

1º Crear la nueva tabla:

```
create table Error_insert (  
    mensaje varchar(255),  
    fecha_error datetime  
);
```

2º Hacer el trigger:

```
delimiter &&  
create trigger verificarEdadJugadorAntesInsertar  
before insert on JUGADOR  
for each row  
begin  
    declare edad int;  
    set edad = TIMESTAMPDIFF(YEAR, new.Fecha_nacimiento, CURDATE());  
    if edad < 16 then  
        insert into Error_insert (mensaje, fecha_error)  
        values (  
            concat('Error: El jugador ', new.Nombre, ' ', new.Apellidos, ' tiene ', edad, ' años y no cumple  
con la edad mínima de 16 años.'),  
            now()  
        );  
        set new.id_jugador = NULL;  
    end if;  
end &&  
delimiter ;
```

```
delimiter &&  
create trigger verificarEdadJugadorAntesInsertar  
before insert on JUGADOR  
for each row  
begin  
    declare edad int;  
  
    set edad = TIMESTAMPDIFF(YEAR, new.Fecha_nacimiento, CURDATE());  
  
    if edad < 16 then  
        insert into Error_insert (mensaje, fecha_error)  
        values (  
            concat('Error: El jugador ', new.Nombre, ' ', new.Apellidos,  
                ' tiene ', edad, ' años y no cumple con la edad mínima de 16 años.'),  
            now()  
        );  
  
        set new.id_jugador = NULL;  
    end if;  
end &&  
delimiter ;  
  
insert into JUGADOR (id_jugador, Nombre, Apellidos, Posicion, EQUIPO_Id_equipo, Fecha_nacimiento)  
values (1002, 'Jugador', 'Menor', 'Delantero', 1, '2010-05-05');  
  
select id_jugador, Nombre, Fecha_nacimiento from JUGADOR where id_jugador = 1002;  
select mensaje, fecha_error from error_insert ei;
```

jugador 1 X

select id\_jugador, Nombre, Fecha\_nacimiento from JUGADOR where

Enter a SQL expression to filter results (use Ctrl+Space)

123 id_jugador	A-Z Nombre	Fecha_nacimiento

```
insert into JUGADOR (id_jugador, Nombre, Apellidos, Posicion, EQUIPO_Id_equipo, Fecha_nacimiento)
values (1002, 'Jugador', 'Menor', 'Delantero', 1, '2010-05-05');

select id_jugador, Nombre, Fecha_nacimiento from JUGADOR where id_jugador = 1002;
select mensaje, fecha_error from error_insert ei;
```

error\_insert 1 X

select mensaje, fecha\_error from error\_insert ei | Enter a SQL expression to filter results (use Ctrl+Space)

A-z mensaje	fecha_error
Error: El jugador Jugador Menor tiene 14 años y no cumple con la edad mínima de 16 años.	2025-03-26 13:30:01

Pero si introducimos un jugador que tenga más de 16 años:

```
insert into JUGADOR (id_jugador, Nombre, Apellidos, Posicion, EQUIPO_Id_equipo, Fecha_nacimiento)
values (1003, 'Jugador', 'Mayor', 'Delantero', 1, '2006-05-05');

select id_jugador, Nombre, Fecha_nacimiento from JUGADOR where id_jugador = 1003;
select mensaje, fecha_error from error_insert ei;
```

jugador 1 X

select id\_jugador, Nombre, Fecha\_nacimiento from JUGADOR where id\_jugador = 1003 | Enter a SQL expression to filter results (use Ctrl+Space)

id_jugador	A-z Nombre	Fecha_nacimiento
1	1.003 Jugador	2006-05-05

## **AWS**

IP: 3.232.242.131

## **GitHub**

<https://github.com/rgragon/Mi-Proyecto-LaLiga-EA-Sports>

## Conclusión

En conclusión, este proyecto me ha permitido trabajar las bases de datos con relaciones entre entidades, cargas masivas, todo tipo de consultas, vistas, funciones, procedimientos y triggers, utilizando, Draw.io para el inicio de la estructura junto con las relaciones, MySQL Workbench para pasar esa estructura a SQL y finalmente subirlo y trabajar en ella en DBeaver, además, la base de datos está subida a AWS.

En mi opinión, es un proyecto que no ha sido fácil, pero me ha ayudado a entender mejor algunas relaciones entre entidades y a mejorar en ellas, y además a tener un buen inicio de una base de datos.