Base de Datos de Fútbol: Centralización y Análisis de Datos en el Mundo del Fútbol

Introducción: Descripción de la Temática

La base de datos que estamos desarrollando se enfoca en el mundo del fútbol, un deporte con una rica y compleja estructura organizativa que involucra múltiples niveles de datos. Esta base de datos tiene el propósito de centralizar y organizar información relevante para facilitar el análisis y la gestión de datos relacionados con equipos, jugadores, competencias y estadísticas. Al integrar datos de diferentes fuentes y formatos, buscamos ofrecer una solución integral que facilite el acceso y análisis de la información, permitiendo una mejor toma de decisiones y un análisis detallado del desempeño en el ámbito del fútbol.

Objetivo

El principal objetivo de esta base de datos es proporcionar una plataforma robusta para almacenar, organizar y analizar datos del fútbol de manera eficiente. Esto incluye:

1. Centralizar la información sobre equipos, jugadores, competencias, y estadísticas.
2. Facilitar el análisis detallado del rendimiento de jugadores y equipos en diferentes competencias.
3. Mejorar la gestión de la información para clubes, asociaciones y federaciones.
4. Ofrecer una herramienta que permita el seguimiento preciso del desempeño a lo largo del tiempo.
5. Permitir la comparación y análisis de datos a nivel internacional, dado el carácter global del fútbol.

Situación Problemática

Actualmente, la información sobre fútbol se encuentra dispersa en diversas fuentes y formatos, lo que presenta varios problemas:

1. Desorganización de Datos: La información está fragmentada, dificultando el acceso y la consolidación de datos relevantes.

2. Análisis Ineficiente: Sin una estructura de datos bien definida, es complicado realizar análisis detallados sobre el desempeño de jugadores y equipos.
3. Gestión de Información: La falta de una herramienta centralizada para gestionar la información afecta la eficiencia en la toma de decisiones y planificación estratégica.
4. Seguimiento de Desempeño: Es difícil realizar un seguimiento continuo y preciso del desempeño histórico de jugadores y equipos.
5. Compatibilidad Internacional: La falta de un sistema capaz de manejar datos de competencias internacionales dificulta la comparación global y el análisis de datos a nivel mundial.

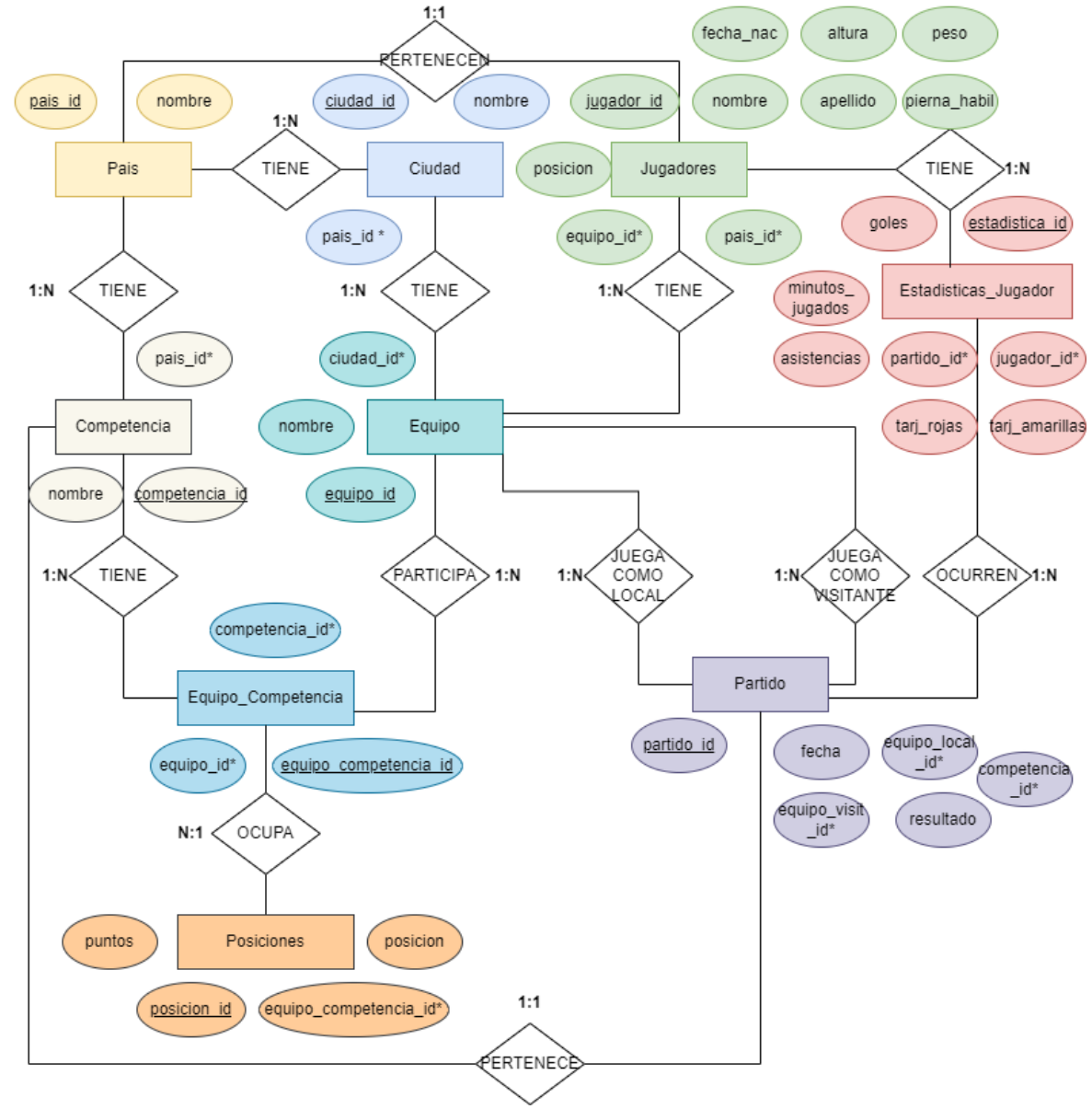
Modelo de Negocio

El modelo de negocio para esta base de datos se basa en ofrecer una solución integral para la gestión y análisis de información en el ámbito del fútbol. Este modelo incluye:

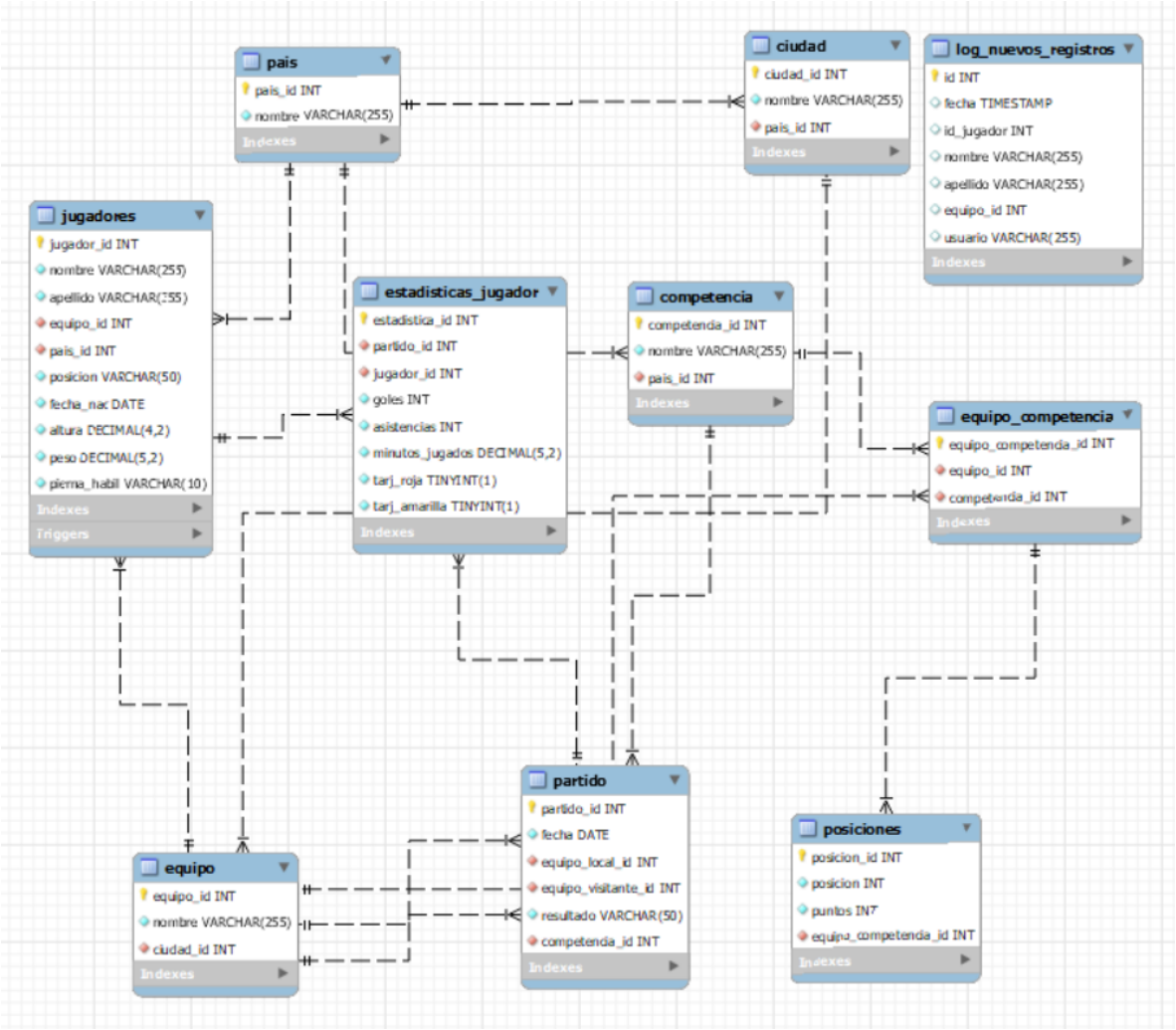
1. Centralización de Datos: Proporcionar una plataforma única donde se consolida toda la información relevante sobre fútbol, facilitando su gestión y consulta.
2. Servicios de Análisis: Ofrecer herramientas avanzadas de análisis para evaluar el rendimiento de jugadores y equipos, proporcionando insights valiosos para clubes y asociaciones.
3. Gestión de Información: Facilitar la administración eficiente de datos para la toma de decisiones estratégicas, mejorando la planificación y la gestión operativa.
4. Acceso Internacional: Permitir la integración y comparación de datos a nivel mundial, satisfaciendo las necesidades de competencias internacionales y comparativas globales.
5. Valor Añadido: Ofrecer valor añadido a través de informes, análisis y visualizaciones que mejoren la comprensión y la toma de decisiones en el ámbito del fútbol.

Esta base de datos no solo busca resolver los problemas actuales de desorganización y análisis ineficiente, sino que también pretende proporcionar una herramienta valiosa para la gestión y el análisis del fútbol a nivel global.

Diagrama de Entidad- Relación (DER)



DER Obtenido por Reverse Engineer en MySQL



Listado de tablas

En este apartado se hará mención de cada una de las tablas junto a una descripción de las mismas.

1- **Tabla país:** Contiene información del nombre de los países involucrados en la base de datos. Ésta tabla se relaciona por medio de su Primary Key (PK) con las tablas CIUDAD, JUGADORES y COMPETENCIA siendo una Foreign Key (FK) en éstas.

Campo	Tipo de Dato	Tipo de Clave
pais_id	INT NOT NULL AUTO_INCREMENT	PK
nombre	VARCHAR(255) NOT NULL	

2- **Tabla ciudad:** Contiene nombres de ciudades que pertenecen a distintos países en las cuales se localizan los distintos equipos. Se relaciona con la tabla PAÍS (por intermedio de la FK pais_id) y con la tabla EQUIPO (por intermedio de su PK que se presenta como FK en la tabla equipo).

Campo	Tipo de Dato	Tipo de Clave
ciudad_id	INT NOT NULL AUTO_INCREMENT	PK
nombre	VARCHAR(255) NOT NULL	
pais_id	INT NOT NULL	FK

3- **Tabla equipo:** Incluye información sobre los equipos de fútbol involucrados. Se relaciona con la tabla CIUDADES (por intermedio de la FK ciudad_id) y con la tabla EQUIPOCOMPETENCIA (por intermedio de su PK equipo_id), con la tabla JUGADORES (a través de su PK equipo_id que se presenta como FK en la tabla JUGADORES) y con la tabla PARTIDO (gracias a su PK equipo_id que se presenta como FK en las columnas equipo_local_id y equipo_visitante_id de la tabla PARTIDO).

Campo	Tipo de Dato	Tipo de Clave
equipo_id	INT NOT NULL AUTO_INCREMENT	PK
nombre	VARCHAR(255) NOT NULL	
ciudad_id	INT NOT NULL	FK

4- **Tabla competencia:** Contiene información sobre las distintas competencias/ligas. Se relaciona con la tabla PAÍS (por intermedio de la FK pais_id), con la tabla EQUIPO_COMPETENCIA (por intermedio de su PK competencia_id) y con la tabla PARTIDOS (ya que su PK competencia_id se presenta como FK en la tabla PARTIDO).

Campo	Tipo de Dato	Tipo de Clave
competencia_id	INT NOT NULL AUTO_INCREMENT	PK
nombre	VARCHAR(255) NOT NULL	
pais_id	INT NOT NULL	FK

5- **Tabla partido:** Posee información sobre los partidos de fútbol disputados, fecha del encuentro, a que competencia pertenece y resultado del mismo. Se relaciona con la tabla EQUIPO (por intermedio de las FKs equipo_local_id y equipo_visitante_id), con la tabla COMPETENCIA (por intermedio de la FK competencia_id) y con la tabla ESTADÍSTICAS_JUGADOR (por intermedio de su PK partido_id que se presenta como FK en la tabla ESTADÍSTICAS_JUGADOR).

Campo	Tipo de Dato	Tipo de Clave
partido_id	INT NOT NULL AUTO_INCREMENT	PK
fecha	DATE NOT NULL	
equipo_local_id	INT NOT NULL	FK
equipo_visitante_id	INT NOT NULL	FK
resultado	VARCHAR(50) NOT NULL	
competencia_id	INT NOT NULL	FK

6- **Tabla jugadores:** Guarda información sobre los jugadores de fútbol, así como a qué equipo y país pertenecen. Se relaciona con la tabla EQUIPO (por intermedio de la FK equipo_id) y con la tabla PAÍS (por intermedio de la FK pais_id) y con la tabla ESTADÍSTICAS_JUGADOR (por intermedio de su PK jugador_id que se presenta como FK en la tabla ESTADÍSTICAS_JUGADOR).

Campo	Tipo de Dato	Tipo de Clave
jugador_id	INT NOT NULL AUTO_INCREMENT	PK
nombre	VARCHAR(255) NOT NULL	
apellido	VARCHAR(255) NOT NULL	
equipo_id	INT NOT NULL	FK
pais_id	INT NOT NULL	FK
posicion	VARCHAR(50) NOT NULL	

fecha_nac	DATE NOT NULL	
altura	DECIMAL (4,2) NOT NULL	
peso	DECIMAL (5,2) NOT NULL	
pierna_habil	VARCHAR(10) NOT NULL	

7- **Tabla estadísticas_jugador:** Contiene estadísticas de los jugadores en los partidos (minutos jugados, asistencias, goles y tarjetas recibidas). Se relaciona con la tabla JUGADORES (por intermedio de la FK jugador_id) y con la tabla PARTIDO (por intermedio de la FK partido_id).

Campo	Tipo de Dato	Tipo de Clave
estadistica_id	INT NOT NULL AUTO_INCREMENT	PK
partido_id	INT NOT NULL	FK
jugador_id	INT NOT NULL	FK
goles	INT NOT NULL	
asistencias	INT NOT NULL	
minutos_jugados	DECIMAL(5,2) NOT NULL	
tarj_roja	BOOLEAN NOT NULL	
tarj_amarilla	BOOLEAN NOT NULL	

8- **Tabla equipo_competencia:** Tabla intermedia que contiene la relación entre equipos y competencias creada para eliminar la relación muchos a muchos. Se relaciona con la tabla EQUIPO (por intermedio de la FK equipo_id), con la tabla COMPETENCIA (por intermedio de la FK competencia_id) y con la tabla POSICIONES (por intermedio de su PK equipo_competencia_id que se presenta como FK en la tabla POSICIONES).

Campo	Tipo de Dato	Tipo de Clave
equipo_competencia_id	INT NOT NULL AUTO_INCREMENT	PK
equipo_id	INT NOT NULL	FK
competencia_id	INT NOT NULL	FK

9- **Tabla posiciones:** Contiene la posición y puntos obtenidos de cada equipo en cada competencia. Se relaciona con la tabla Equipo_Competencia mediante la FK equipo_competencia_id.

Campo	Tipo de Dato	Tipo de Clave
posicion_id	INT NOT NULL AUTO_INCREMENT	PK
equipo_competencia_id	INT NOT NULL	FK
posicion	INT NOT NULL	
puntos	INT NOT NULL	

10 **Tabla log_nuevos_registros:** Esta tabla fué creada con el propósito de registrar de manera automática cada vez que se inserta un nuevo jugador en la tabla jugadores.Este registro automático es gestionado por un trigger llamado after_insert_trigger, que se activa inmediatamente después de que se inserta un nuevo jugador en la tabla jugadores. El trigger inserta en log_nuevos_registros información relevante como la fecha de inserción, los datos del jugador (nombre, apellido, equipo) y el usuario que realizó la acción. Esto es útil para auditorías, mantener la integridad de los datos y verificar qué usuarios han realizado cambios en la base de datos.

Campo	Tipo de Dato	Tipo de Clave
id	INT NOT NULL AUTO_INCREMENT	PK
fecha	TIMESTAMP DEFAULT CURRENT_TIMESTAMP	
id_jugador	INT NOT NULL	
nombre	VARCHAR(255)	
apellido	VARCHAR(255)	
equipo_id	INT	
usuario	VARCHAR(255)	

Estructura e ingesta de datos

- Se realiza por medio del archivo population.sql

Objetos de la Base de Datos

1. Documentación de VISTAS

Vista: view_jugadores_Lazio

Descripción: Esta vista nos mostrará los jugadores que integran el plantel de Lazio, su posición y a qué país pertenecen.

Columnas:

- Nombre: Nombre del jugador
- Apellido: Apellido del jugador
- Posición: Posición dentro del plantel.
- Nacionalidad: Nacionalidad del jugador.

Ejemplo de consulta:

```
SELECT *

FROM futboldb.view_jugadores_lazio;
```

Vista: view_cantidad_jugadores_equipo

Descripción: Esta vista nos dará información acerca de la cantidad de jugadores que componen cada plantel y país al que pertenece dicho equipo presente en la base de datos.

Columnas:

- Equipo: Nombre del equipo
- País: País al que pertenece el equipo
- Cantidad_jugadores: Cantidad de jugadores que integran el plantel.

Ejempo de consulta:

```
SELECT *

FROM view_cantidad_jugadores_equipo;
```

Vista: view_promedio_edad_altura_peso_posicion

Descripción: La creación de esta vista nos resulta útil para conocer los promedios de edad, peso y altura por posición con los que cuenta cada plantel

Nota: Para el cálculo de la edad se toma en cuenta el día 23-06-01 para hacer el cálculo ya que la base de datos cuenta con partidos de este año.

Columnas:

- Equipo: Nombre del equipo.
- Posición: Portero/Defensor/Mediocampista/Delantero.
- Promedio_edad: Promedio de edad por equipo y posición.
- Promedio_altura: Promedio de altura por equipo y posición.
- Promedio_peso: Promedio de peso por equipo y posición.

Ejemplo de consulta:

```
SELECT *

FROM view_promedio_edad_altura_peso_posicion

WHERE equipo = 'River Plate';
```

Vista: view_goleadores_liga_argentina

Descripción: Esta vista mostrará los 5 máximos goleadores de la Liga Profesional Argentina.

Columnas:

- Nombre: Nombre del jugador.
- Apellido: Apellido del jugador.
- Equipo: Equipo al que pertenece.
- Total_goles: Suma de los goles que tiene en el campeonato.

Ejemplo de consulta:

```
SELECT *

FROM view_goleadores_liga_argentina;
```

Vista: view_punteros_distintas_ligas

Descripción: Esta vista tiene el objetivo de visualizar quienes son los punteros de cada liga presente incluyendo los puntos respectivos con los que cuenta en la competencia.

Columnas:

- Liga: Nombre de la liga en la cual participa el equipo.
- Equipo: Nombre del equipo puntero.
- Puntos: Cantidad de puntos cosechados en la respectiva competición.

Ejemplo de consulta:

```
SELECT *

FROM view_punteros_distintas_ligas

ORDER BY puntos DESC;
```

2. Documentación de FUNCIONES:

Función: minutos_disputados_jugador

Descripción: Función para conocer la cantidad de minutos disputados de un jugador X entre determinadas fechas.

Parámetros:

- p_jugador_id: Id del jugador.
- p_fecha_inicio: Fecha de inicio del intervalo (formato YYYY-MM-DD).
- p_fecha_fin: Fecha de fin del intervalo (formato YYYY-MM-DD).

Retorno:

- Cantidad de minutos disputados por un jugador en el intervalo de tiempo especificado.

Ejemplo de uso:

```
SELECT minutos_disputados_jugador(1, '2023-01-01', '2023-05-30') AS total_minutos;
```

Función: cantidad_jugadores_por_pais

Descripción: Función que calcula la cantidad de jugadores en la base de datos de acuerdo a un país al que pertenecen y devuelve un mensaje en caso de que no haya jugadores de ese país.

Parámetros:

- p_nombre_pais: Nombre del país.

Retorno:

- **Mensaje:** En el caso que el país esté presente en la base de datos retornará la cantidad de jugadores que corresponden a ese país. Caso contrario retornará el mensaje “No hay datos de jugadores del país”.

Ejemplo de uso:

```
SELECT cantidad_jugadores_por_pais('Argentina') AS mensaje;
```

Función: edad_jugador

Descripción: Esta función calculará la edad de un jugador en años pasándole como parámetros el nombre, apellido y club al que pertenece.

Nota: Como fecha actual se utiliza el día 2023-06-01 (dado que es el último registro insertado en la base de datos de la temporada 2023).En caso que no haya coincidencias saldrá el mensaje: No se encontró el jugador o equipo especificado

Parámetros:

- p_nombre: Nombre del jugador
- p_apellido: Apellido del jugador
- p_nombre_equipo: Equipo al que pertenece el jugador.

Retorno: Devolverá la edad en años del jugador.

Ejemplo de uso:

```
SELECT edad_jugador('Ignacio', 'Scocco', 'River Plate') AS edad;
```

3. Documentación de TRIGGERS:

Trigger: after_insert_trigger

Descripción: Registra la inserción de un nuevo jugador en la tabla log_nuevos_registros.

Detalles:

- Tabla afectada: Jugadores
- Acción: Insert
- Información registrada: fecha, id_jugador,nombre, apellido, equipo_id, usuario que realiza la inserción.

Ejemplo: Se inserta un nuevo jugador en la tabla jugadores y el trigger registra la acción en la tabla log_nuevos_registros con los detalles.

Trigger: before_insert_trigger

Descripción: Este trigger está diseñado para ejecutarse antes de la operación de inserción (donde una incorrecta inserción de datos en la tabla jugadores como ser nombre y apellido todo en mayúsculas o todo en minúscula lo convertirá al formato correspondiente).

Detalles:

- Tabla afectada: Jugadores
- Acción: Insert
- Validación: Correcta escritura del Nombre y Apellido del jugador.

  Ejemplo: Se intenta insertar el nombre y apellido de un jugador todo en mayúsculas (ejemplo: ‘LUCAS’, ‘BERMUDEZ’). El trigger se dispara y corrige la inserción al formato correcto ‘Lucas’, ‘Bermudez’).

4. Documentación de Procedimientos Almacenados

Procedimiento: ingreso_nuevo_jugador

Descripción: Inserta un nuevo jugador en la tabla jugadores utilizando los valores proporcionados como parámetros de entrada, validando si existe o no el jugador, el equipo y el país, de lo contrario arrojará error.

Parámetros:

- p_nombre: Nombre del jugador.
- p_apellido: Apellido del jugador.
- p_equipo_id: Id del equipo.
- p_pais_id: Nacionalidad del jugador.
- p_posicion: Posición del jugador.
- p_fecha_nac: Fecha de nacimiento.
- p_altura: Altura del jugador.
- p_peso: Peso del jugador.
- p_pierna_habil: Pierna hábil del jugador.

Retorno: Mensaje de éxito o error (‘El país no existe’, ‘El equipo no existe’ o ‘El jugador ya existe en la base de datos’).

Ejemplo de uso:

```
CALL ingreso_nuevo_jugador('Claudo','Echeverri',1,1,'Delantero','2006-01-02',1.71,62.00,'Izquierda');
```

Procedimiento: actualizar_equipo_jugador

Descripción: El objetivo de la creación de este procedimiento es la actualización del equipo al que pertenece un jugador. Este procedimiento será realmente útil en época de mercado de pases donde los jugadores son transferidos de un equipo a otro.

Parámetros:

- p_jugador_id: Id del jugador
- p_nuevo_equipo_id: Id del nuevo equipo del jugador.

Retorno: Mensaje de éxito o error (‘El jugador no existe’ o ‘El nuevo equipo no existe’ en caso que no estén presentes en la base).

Ejemplo de uso:

CALL actualizar_equipo_jugador(239, 11);

Procedimiento: registrar_partido

Descripción: Procedimiento almacenado diseñado para ingresar nuevos registros de partidos y que a su vez actualiza las posiciones, es decir que si el equipo ganó le sume 3 puntos en la tabla posiciones, si empató le sume 1 y si perdió no sume puntos.

Nota: Utilizamos las funciones SUBSTRING_INDEX y CAST para extraer y convertir los goles del formato goles_local-goles_visitante.

Parámetros:

- p_fecha: Fecha del nuevo partido a registrar.
- p_equipo_local_id: Id del equipo local.
- p_equipo_visitante_id: Id del equipo visitante.
- p_resultado: Resultado del encuentro.
- p_competencia_id: Id de la competencia a la cual pertenece el partido.

Retorno: Mensaje de éxito o error.

Ejemplo de uso:

CALL registrar_partido('2023-06-01', 1, 2, '5-0', 1);

Roles y Permisos

Primero creamos un súper usuario llamado ‘super_admin’ el cual cuenta con todos los privilegios posibles en todas las bases de datos y tablas dentro del servidor MySQL. Esto incluye privilegios para realizar cualquier operación, como SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, GRANT, REVOKE y puede otorgar estos permisos que posee a otros usuarios.

Además, el usuario está configurado con la política de seguridad FAILED_LOGIN_ATTEMPTS 3, lo que significa que, si el usuario intenta iniciar sesión y falla 3 veces, su cuenta será bloqueada temporalmente (1 día).

Luego creamos un usuario llamado ‘admin_futboldb’ que, a diferencia del primero solo tiene permisos sobre la base de datos futboldb (puede operar con los datos, realizar operaciones de definición de esquemas, crear vistas, etc) pero NO tiene la capacidad de otorgar sus propios privilegios a otros usuarios.

También hemos creados dos ROLES:

1. **role_select_tablas:** Este rol tiene permisos de visualización (SELECT) de todas las tablas y vistas de la base de datos.

2. **role_crud_futbol**: tiene permisos completos de **CRUD** (Create, Read, Update, Delete) en las tablas jugadores, estadisticas_jugador, y partido de la base de datos futboldb. Estos permisos permiten realizar cualquier operación relacionada con la creación, lectura, modificación, y eliminación de datos en esas tres tablas específicas.

Por último, creamos dos usuarios más: ‘cuerpo_tecnico’ al cual se le otorgó el rol: rol_select_tablas e ‘interno_club’ asignándole el rol: rol_crud_futbol respectivamente.

Asignar estos roles con permisos limitados no solo mejora la seguridad, sino que también optimiza el flujo de trabajo, asegurando que cada usuario se enfoque en sus responsabilidades sin riesgos innecesarios. Esto crea un entorno de trabajo más organizado, seguro y eficiente, en el cual las tareas y permisos están claramente definidos.

Back up de la base de datos

Se ha añadido un script del backup de la base de datos desarrollada en este proyecto.

Adicionalmente se puede generar el comando make backup-db en **CodeSpaces**, el cual permite ejecutar un backup de la base de datos de manera manual.

Herramientas y tecnologias usadas

- **Makefile** (Para generar una interfaz sencilla de procesos)
- **Docker** (Para generar un container)
- **MySQL** (Motor de bases de datos version: latest)
- **MySQL Workbench** (Interfaz gráfica)
- **ChatGPT** (Para generar datos ficticios)
- **Draw.io** (Para crear el diagrama entidad-relación)
- **Power BI** (Para generar dashboard de ejemplo)

Como levantar el proyecto en CodeSpaces GitHub

- **env:** Archivo con contraseñas y data secretas.
- **Makefile:** Abstracción de creacción del proyecto.
- **docker-compose.yml:** Permite generar la base de datos en forma de contenedores.

Pasos para arrancar el proyecto

- En la terminal de linux escribir :
    - make *En caso de que genere el error de que no existe conexion al socket, volver al correr el comando make.*
    - make clean-db *Para limpiar la base de datos.*
    - make test-db *Para mirar los datos de cada tabla.*
    - make backup-db *Para realizar un backup de la base de datos.*
    - make access-db *Para acceder a la base de datos.*