

Segunda entrega del proyecto final – SQL Coderhouse

1) Listado de vistas:

- DATOS_ARBITRO

Esta vista fue creada para obtener los datos del árbitro, cuenta con tres columnas (id_arbitro, nombre_arbitro, país_arbitro). Las primeras dos son obtenidas de la tabla 'arbitro' mientras que la ultima es obtenida de la tabla 'país'.

Script de creación de esta vista:

```
76 ----- VISTA DATOS DEL ARBITRO -----
77
78 create or replace view datos_arbitro as (
79 select arbitro.id_arbitro as id_arbitro,
80        arbitro.nombre as nombre_arbitro,
81        pais.nombre as pais_arbitro
82 from arbitro as arbitro
83 left join pais as pais
84 on pais.id_pais = arbitro.id_pais
85 );
```

Ejemplo de utilización de vista:

	id_arbitro	nombre_arbitro	pais_arbitro
▶	1	Anthony Taylor	Inglaterra
	2	Daniel Siebert	Alemania
	3	Clément Turpin	Francia
	4	Antonio Mateu Lahoz	España
	5	Pierluigi Collina	Italia

- DATOS_EQUIPO

Esta vista fue creada para obtener los datos del equipo, cuenta con seis columnas (id_equipo, nombre_equipo, copas_equipo, estadio_equipo, liga_equipo, pais_liga). Las primeras tres son obtenidas de la tabla 'equipo', la cuarta columna (estadio_equipo) es obtenida de la tabla 'estadio' y las últimas dos son obtenidas de la tabla 'liga'.

Script de creación de esta vista:

```
----- VISTA DATOS DEL EQUIPO -----

create or replace view datos_equipo as (
select equipo.id_equipo as id_equipo,
       equipo.nombre as nombre_equipo,
       equipo.cantidad_copas as copas_equipo,
       estadio.nombre as estadio_equipo,
       liga.nombre as liga_equipo,
       pais.nombre as pais_liga
from equipo as equipo
left join estadio as estadio
on estadio.id_estadio = equipo.id_estadio
left join liga as liga
on liga.id_liga = equipo.id_liga
left join pais as pais
on pais.id_pais = liga.id_pais
);
```

Ejemplo de utilización de vista:

	id_equipo	nombre_equipo	copas_equipo	estadio_equipo	liga_equipo	pais_liga
▶	1	Screamer	37	San Javier Stadium	Premier League	Inglaterra
	2	Cat	22	Canarana Stadium	Premier League	Inglaterra
	3	Pelican	20	Paraparaumu Stadium	Premier League	Inglaterra
	4	Red sheep	9	Embessa Stadium	Premier League	Inglaterra
	5	Water moccasin	24	Rae Lakes Stadium	Premier League	Inglaterra
	6	Black vulture	29	Lemhi County Stadium	Premier League	Inglaterra
	7	Deer	43	Robinson River Stadium	Premier League	Inglaterra
	8	Gonolek's	3	Kikwit Stadium	Premier League	Inglaterra
	9	Crab-eating raccoon	32	Biaru Stadium	Premier League	Inglaterra
	10	Lory	8	Tambolaka Stadium	Premier League	Inglaterra
	11	Tree porcupine	49	Pohnpei International ...	Premier League	Inglaterra
	12	Seal	2	Toronto Executive Sta...	Premier League	Inglaterra
	13	Southern lapwing	32	Portsmouth Stadium	Premier League	Inglaterra
	14	Lava gull	37	Tambor Stadium	Premier League	Inglaterra
	15	Springhare	28	El Daein Stadium	Premier League	Inglaterra

- DATOS_ESTADIO

Esta vista fue creada para obtener los datos del estadio, cuenta con cinco columnas (id_estadio, nombre_estadio, nombre_equipo, capacidad_estadio, nombre_ciudad). Las primeras dos y la cuarta son obtenidas de la tabla 'estadio', la tercera columna (nombre_equipo) es obtenida de la tabla 'equipo' y la ultima es obtenida de la tabla 'ciudad'.

Script de creación de esta vista:

```
----- VISTA DATOS DEL ESTADIO -----  
  
create or replace view datos_estadio as (  
  select  estadio.id_estadio as id_estadio,  
          estadio.nombre as nombre_estadio,  
          equipo.nombre as nombre_equipo,  
          estadio.capacidad as capacidad_estadio,  
          ciudad.nombre as nombre_ciudad  
  from estadio as estadio  
  left join equipo as equipo  
  on equipo.id_estadio = estadio.id_estadio  
  left join ciudad as ciudad  
  on ciudad.id_ciudad = estadio.id_ciudad  
);
```

Ejemplo de utilización de vista:

	id_estadio	nombre_estadio	nombre_equipo	capacidad_estadio	nombre_ciudad
▶	1	San Javier Stadium	Screamer	57373	Londres
	2	Canarana Stadium	Cat	24687	Londres
	3	Paraparaumu Stadium	Pelican	64388	Londres
	4	Embessa Stadium	Red sheep	49603	Londres
	5	Rae Lakes Stadium	Water moccasin	60481	Londres
	6	Lemhi County Stadium	Black vulture	59581	Londres
	7	Robinson River Stadium	Deer	35249	Londres
	8	Kikwit Stadium	Gonolek's	33759	Londres
	9	Biaru Stadium	Crab-eating raccoon	20941	Londres
	10	Tambolaka Stadium	Lory	66554	Londres
	11	Pohnpei International ...	Tree porcupine	55117	Londres
	12	Toronto Executive Sta...	Seal	51674	Londres
	13	Portsmouth Stadium	Southern lapwing	74610	Londres
	14	Tambor Stadium	Lava gull	20652	Londres
	15	El Daein Stadium	Springhare	42329	Londres

- DATOS_JUGADOR

Esta vista fue creada para obtener los datos del jugador, cuenta con seis columnas (id_jugador, nombre_jugador, edad_jugador, posicion_jugador, pierna_habil_jugador, pais_jugador). Las primeras tres son obtenidas de la tabla 'jugador', la cuarta columna (posicion_jugador) es obtenida de la tabla 'posicion', la quinta columna (pierna_habil_jugador) es obtenida de la tabla 'pierna_habil' y la ultima es obtenida de la tabla 'pais'.

Script de creación de esta vista:

```
----- VISTA DATOS DEL JUGADOR -----  
  
create or replace view datos_jugador as (  
select jugador.id_jugador as id_jugador,  
       jugador.nombre as nombre_jugador,  
       jugador.edad as edad_jugador,  
       posicion.nombre as posicion_jugador,  
       pierna_habil.descripcion as pierna_habil_jugador,  
       pais.nombre as pais_jugador  
from jugador as jugador  
left join posicion as posicion  
on posicion.id_posicion = jugador.id_posicion  
left join pierna_habil as pierna_habil  
on pierna_habil.id_pierna_habil = jugador.id_pierna_habil  
left join pais as pais  
on pais.id_pais = jugador.id_pais  
);
```

Ejemplo de utilización de vista:

	id_jugador	nombre_jugador	edad_jugador	posicion_jugador	pierna_habil_jugador	pais_jugador
▶	1	Mohandis	35	Delantero	Izquierda	España
	2	Hardy	31	Mediocampista	Derecha	Inglaterra
	3	Jammal	33	Mediocampista	Derecha	Francia
	4	Ephrayim	38	Defensor	Derecha	Inglaterra
	5	Yale	18	Mediocampista	Derecha	España
	6	Irv	40	Mediocampista	Izquierda	Alemania
	7	My	20	Arquero	Izquierda	España
	8	Nevil	33	Defensor	Izquierda	España
	9	Kelby	32	Defensor	Derecha	España
	10	Claudian	26	Mediocampista	Izquierda	Italia
	11	Jase	25	Mediocampista	Derecha	España
	12	Eldredge	22	Delantero	Derecha	Italia
	13	Eamon	35	Delantero	Izquierda	Italia
	14	Dwight	26	Defensor	Derecha	Francia
	15	Morly	36	Arquero	Derecha	España






- DATOS_PARTIDO

Esta vista fue creada para obtener los datos del partido, cuenta con ocho columnas (id_partido, fecha_partido, estadio_partido, equipo1_nombre, equipo2_nombre, arbitro_nombre, liga_partido). Las primeras tres son obtenidas de la tabla 'partido', la cuarta columna (equipo1_nombre) es obtenida de la tabla 'equipo' al igual que la quinta columna (equipo2_nombre), la sexta columna (arbitro_partido) es obtenida de la tabla 'arbitro' y la septima es obtenida de la tabla 'liga' y la ultima columna es una concatenación que funciona como resultado de partido, concatenando goles del equipo1, un guion medio y los goles del equipo2.

Script de creación de esta vista:

```
----- VISTA DATOS DEL PARTIDO -----  
  
create or replace view datos_partido as (  
select partido.id_partido as id_partido,  
       partido.fecha as fecha_partido,  
       estadio.nombre as estadio_partido,  
       equipo1.nombre as equipo1_nombre,  
       equipo2.nombre as equipo2_nombre,  
       arbitro.nombre as arbitro_partido,  
       liga.nombre as liga_partido,  
       concat(goles_equipo1, '-', goles_equipo2) as resultado_partido  
from partido as partido  
left join estadio as estadio  
on estadio.id_estadio = partido.id_estadio  
left join equipo as equipo1  
on equipo1.id_equipo = partido.id_equipo1  
left join equipo as equipo2  
on equipo2.id_equipo = partido.id_equipo2  
left join arbitro as arbitro  
on arbitro.id_arbitro = partido.id_arbitro  
left join liga as liga  
on liga.id_liga = equipo1.id_liga  
);
```

Ejemplo de utilización de vista:

Result Grid		 Filter Rows:	<input type="text"/>	Export:	 Wrap Cell Contents:	 Fetch rows:		
	id_partido	fecha_partido	estadio_partido	equipo1_nombre	equipo2_nombre	arbitro_partido	liga_partido	resultado_partido
▶	1	2022-10-10	San Javier Stadium	Screamer	Cat	Pierluigi Collina	Premier League	0-6
	2	2022-03-10	San Javier Stadium	Screamer	Pelican	Antonio Mateu Lahoz	Premier League	5-4
	3	2023-01-13	San Javier Stadium	Screamer	Red sheep	Pierluigi Collina	Premier League	2-6
	4	2022-04-06	San Javier Stadium	Screamer	Water moccasin	Antonio Mateu Lahoz	Premier League	3-3
	5	2023-01-27	San Javier Stadium	Screamer	Black vulture	Pierluigi Collina	Premier League	3-5
	6	2023-02-23	San Javier Stadium	Screamer	Deer	Anthony Taylor	Premier League	0-1
	7	2022-10-20	San Javier Stadium	Screamer	Gonolek's	Pierluigi Collina	Premier League	6-5
	8	2022-06-30	San Javier Stadium	Screamer	Crab-eating raccoon	Clément Turpin	Premier League	3-0
	9	2022-09-05	San Javier Stadium	Screamer	Lory	Antonio Mateu Lahoz	Premier League	0-6
	10	2023-01-14	San Javier Stadium	Screamer	Tree porcupine	Pierluigi Collina	Premier League	1-5
	11	2022-08-28	San Javier Stadium	Screamer	Seal	Pierluigi Collina	Premier League	6-6
	12	2022-08-26	San Javier Stadium	Screamer	Southern lapwing	Daniel Siebert	Premier League	6-4
	13	2022-06-10	San Javier Stadium	Screamer	Lava gull	Antonio Mateu Lahoz	Premier League	5-2
	14	2022-03-29	San Javier Stadium	Screamer	Springhare	Pierluigi Collina	Premier League	2-6
	15	2022-07-04	San Javier Stadium	Screamer	Langur	Daniel Siebert	Premier League	1-3

2) Lista de funciones:

- FN_CANT_PARTIDOS_POR_EQUIPO

Esta función obtiene por parámetro un dato de tipo entero con el id del equipo a buscar la cantidad de partidos. Esta función retorna un dato de tipo entero también con la cantidad de partidos que jugo ese equipo.

Esta función fue creada con el objetivo de averiguar cuantos partidos llevan jugados cada uno de los equipos.

La tabla que implementa esta función es 'partido'.

Script de creación de función:

```
----- FUNCION CANTIDAD PARTIDOS POR EQUIPO -----  
  
delimiter $$  
create function fn_cant_partidos_por_equipo(p_id_equipo int)  
returns int  
deterministic  
begin  
declare v_cant_partidos int;  
set v_cant_partidos =  
    (select count(*)  
from partido  
where (id_equipo1 = p_id_equipo)  
or (id_equipo2 = p_id_equipo));  
return v_cant_partidos;  
end$$  
delimiter ;
```

- FN_CANT_PARTIDOS_POR_FECHA

Esta función obtiene por parámetro un dato de tipo 'date' con la fecha a buscar la cantidad de partidos. Esta función retorna un dato de tipo entero también con la cantidad de partidos que se jugaron en la fecha pasada por parámetro.

Esta función fue creada con el objetivo de averiguar cuantos partidos se jugaron en cierta fecha.

La tabla que implementa esta función es 'partido'.

Script de creación de función:

```
----- FUNCION CANTIDAD PARTIDOS POR FECHA -----  
  
delimiter $$  
create function fn_cant_partidos_por_fecha(p_fecha date)  
returns int  
deterministic  
begin  
declare v_cant_partidos int;  
set v_cant_partidos =  
    (select count(*)  
from partido  
where (fecha = p_fecha));  
return v_cant_partidos;  
end$$  
delimiter ;
```

3) Listado de stored procedures:

- SP_INSERT_PAIS

Este stored procedure se encarga de insertar un registro en la tabla 'pais' en la base de datos siempre y cuando se pase por parametro un nombre valido, es decir, un dato de tipo varchar() no vacío.

Script de creación del stored procedure:

```
----- STORED PROCEDURE DE INSERT A TABLA 'PAIS' -----  
  
-- DROP AND CREATE PROCEDIMIENTO "SP_INSERT_PAIS" QUE AGREGA REGISTROS A LA TABLA "PAIS"  
DROP PROCEDURE IF EXISTS SP_INSERT_PAIS;  
DELIMITER $$  
CREATE PROCEDURE SP_INSERT_PAIS(  
    IN NOMBRE VARCHAR(60) -- PARAMETRO DE ENTRADA DE TIPO VARCHAR  
)  
BEGIN  
    IF NOMBRE != "" THEN -- CONDICIONAL QUE PIDE QUE EL VALOR NOMBRE NO ESTE VACIO PARA CARGAR EL NUEVO REGISTRO.  
        INSERT INTO PAIS (NOMBRE)  
        VALUES (NOMBRE);  
        SELECT * FROM PAIS ORDER BY ID_PAIS DESC; -- MUESTRA LA TABLA CON EL VALOR YA INGRESADO.  
    ELSE  
        SET @MESSAGE = "Error al intentar registrar un nuevo pais."; -- EN CASO DE QUE EL NOMBRE SEA VACIO SETEAMOS EL MENSAJE A MOSTRAR  
        SELECT @MESSAGE; -- MUESTRA EL MENSAJE CON EL ERROR.  
    END IF;  
END  
$$
```

- SP_ORDER_BY_PROPERTY

Este stored procedure se encarga de ordenar una tabla por uno de sus campos de forma ascendente o descendente (según se ingrese).

Las tablas que interactúan con este stored procedure son todas, ya que depende del nombre de la tabla que se ingrese por parámetros.

Script de creación del stored procedure:

```
----- STORED PROCEDURE DE ORDENAMIENTO -----  
  
-- DROP AND CREATE PROCEDIMIENTO "SP_ORDER_BY_PROPERTY" QUE ORDENA TABLA POR PROPIEDAD DE FORMA ASCENDENTE/DESCENDENTE SEGUN SE LE PASE"  
DROP PROCEDURE IF EXISTS SP_ORDER_BY_PROPERTY;  
delimiter $$  
CREATE PROCEDURE SP_ORDER_BY_PROPERTY (IN param_tabla varchar(32), -- PARAMETRO DE ENTRADA DE TIPO VARCHAR  
    IN param_order varchar(32), -- PARAMETRO DE ENTRADA DE TIPO VARCHAR  
    IN param_asc_desc varchar(32) -- PARAMETRO DE ENTRADA DE TIPO VARCHAR  
)  
BEGIN  
    SET @CONSULTA = CONCAT('SELECT * FROM ', param_tabla, ' order by', ' ', param_order, ' ', param_asc_desc);  
    PREPARE CONSULTA FROM @CONSULTA;  
    EXECUTE CONSULTA;  
    DEALLOCATE PREPARE CONSULTA;  
END  
$$  
delimiter
```

4) Scripts:

- [Script de primeros inserts](#)
- [Archivos 'csv' para importacion de datos y archivo 'pdf' con el paso a paso](#)
- [Script de creacion de vistas](#)
- [Script de creacion de funciones](#)
- [Script de creacion de stored procedures](#)
- [Script de creacion de triggers](#)

Repositorio en Github: [Repositorio SQL-Coderhouse](#)