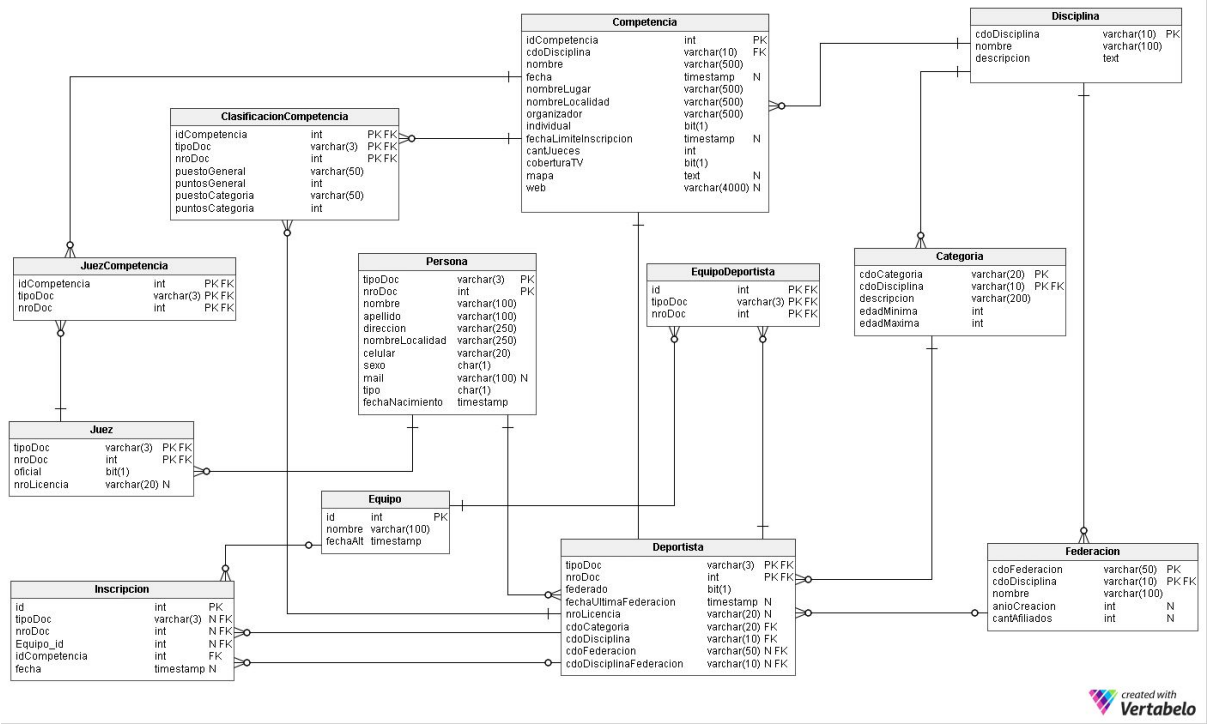
Trabajo práctico espacial
Bases de datos

Nro De Grupo: 18
Intengrantes:
Prado Marcelo
Bravo Eduardo

Introducción:

La cátedra de la materia bases de datos nos solicitó la implementación de controles y servicios a partir de un esquema. Dicho esquema tiene la siguiente forma:



Para ajustar el esquema elaboramos un script en sql llamado G18_creacion.sql a partir del sistema Vertabelo, En Vertabelo importamos el esquema desde un XML proporcionado por la cátedra.
En dicho script creamos todas las tablas con sus restricciones de nulidad, primary key y foreign key de todas las tablas del diagrama. También agregamos los datos para poder realizar pruebas en la estructura.

ELABORACIÓN DE RESTRICCIONES:

a. Debe haber al menos un juez para cada competencia.
Para esta solución se crea un CHECK a nivel de tupla.

```
ALTER TABLE GR18_Competencia
ADD CONSTRAINT gr18_ck_1 CHECK ( cantJueces > 0 );
```

b. Un deportista no puede formar parte de más de tres equipos en un mismo año.

```
CREATE ASSERSTION gr18_ck_2
CHECK NOT EXISTS (
        SELECT E.nrodoc, count(E.id), EXTRACT (YEAR from I.fechaalta) as anio
        FROM GR18_Equipo I
        JOIN GR18_EquipoDeportista E
        ON (I.id = E.id)
        WHERE I.id IS NOT NULL
        GROUP BY E.nrodoc, anio
        HAVING count(E.id) > 3
);
```

c. Las inscripciones no se pueden realizar luego de la fecha límite de inscripción

```
CREATE ASSERSTION gr18_ck_3
CHECK NOT EXISTS (
    SELECT 1
    FROM GR18_Inscripcion I
    JOIN GR18_Competencia C
    ON (C.idCompetencia = I.idCompetencia)
    WHERE I.fecha > C.fechaLimiteInscripcion
);
```

d. Para cada categoría la edad mínima debe ser por lo menos 10 años menos que la edad máxima.

```
 ALTER TABLE GR18_Categoria
    ADD CONSTRAINT gr18_ck_1 CHECK( edadMinima <= (edadMaxima-10));
```

e. Sólo es posible realizar inscripciones de equipos o de deportistas; no de ambos.

```
ALTER TABLE GR18_Inscripcion
    ADD CONSTRAINT gr18_ck_2 CHECK( (
        Equipo_id IS NULL AND
        tipoDoc IS NOT NULL AND
        nroDoc IS NOT NULL) OR (
        Equipo_id IS NOT NULL AND
        tipoDoc IS NULL AND
        nroDoc IS NULL) );
```

f. Un juez que también es deportista, no puede participar en una competencia en la cual se desempeña como juez.

```
CREATE OR REPLACE FUNCTION TRFN_GR18_Multiplestablas_Juezdeportista()
        RETURNS trigger AS $$
        DECLARE flag1 integer;
        DECLARE flag2 integer;
        BEGIN
            SELECT INTO flag1  1
                FROM GR18_Inscripcion I
                JOIN GR18_JuezCompetencia J
                ON (I.nroDoc = J.nroDoc AND I.tipoDoc = J.tipoDoc AND
I.idCompetencia = J.idCompetencia)
            UNION
            SELECT 1
                FROM GR18_Inscripcion I
                JOIN GR18_EquipoDeportista E
                ON (I.Equipo_id = E.Id)
                JOIN GR18_JuezCompetencia J
                ON (E.nroDoc = J.nroDoc AND E.tipoDoc = J.tipoDoc AND
I.idCompetencia = J.idCompetencia);
            IF (flag1 = 1) THEN
                RAISE EXCEPTION 'Un juez que también es deportista, no puede
participar en una competencia en la cual se desempeña como juez';
        END IF;
        RETURN NEW;
    END; $$
    LANGUAGE 'plpgsql';
```

g. Si la competencia es grupal no se deben permitir inscripciones individuales

```
CREATE ASSERSTION gr18_ck_3
CHECK NOT EXISTS (
    SELECT 1
    FROM GR18_Inscripcion I
    JOIN GR18_Competencia C
    ON (I.idCompetencia = C.idCompetencia)
    WHERE I.individual = B'1'
```

h. Se debe controlar que en la clasificación, por cada competencia grupal, todos los integrantes de cada equipo tengan asignados los mismos puntos y puestos

```
CREATE OR REPLACE FUNCTION TRFN_GR18_ClasificacionCompetencia_mismos() RETURNS
trigger AS $$
        DECLARE flag integer;
        BEGIN
    SELECT INTO flag COUNT(*) as cantidad, CC.puestoGeneral, CC.puntosGeneral,
CC.idCompetencia, I.Equipo_id
    FROM GR18_ClasificacionCompetencia CC
    JOIN GR18_Competencia C
    ON ( CC.idCompetencia = C.idCompetencia )
    JOIN GR18_Inscripcion I
    ON ( I.idCompetencia = C.idCompetencia )
    JOIN GR18_EquipoDeportista E
    ON ( E.id = I.Equipo_id )
    JOIN GR18_Deportista D
    ON ( E.nroDoc = D.nroDoc AND E.tipoDoc = D.tipoDoc )
    WHERE C.individual = B'0'
    GROUP BY ( CC.puestoGeneral, CC.puntosGeneral, CC.idCompetencia, I.Equipo_id
)
    HAVING COUNT(*) <> (SELECT COUNT(*) FROM GR18_EquipoDeportista WHERE id =
I.Equipo_id );

               IF (flag = 1) THEN
                    RAISE EXCEPTION 'La competencia es grupal por lo tanto no se
deben permitir inscripciones individuales';
            END IF;
            RETURN NEW;
        END; $$
    LANGUAGE 'plpgsql';
```

SERVICIOS:

1 Dado una competencia, se debe devolver un listado, lo más completo posible, de los deportistas inscriptos.

```
CREATE OR REPLACE FUNCTION GR18_lista_deportista(INTEGER) RETURNS TABLE(
  a VARCHAR(3),
  b INTEGER,
  c VARCHAR(50),
  d VARCHAR(50),
  e VARCHAR(100),
  f timestamp,
  g VARCHAR(100)

) AS $B$
BEGIN
      RETURN QUERY
  SELECT P.tipodoc, P.nrodoc, P.apellido, P.nombre, C.nombre, C.fecha,
C.nombrelugar
  FROM  GR18_Competencia C
  JOIN GR18_Inscripcion I
  ON ( I.idCompetencia = C.idCompetencia )
  LEFT JOIN GR18_Equipo E
  ON ( E.id = I.Equipo_id )
  LEFT JOIN GR18_EquipoDeportista ED
  ON ( E.id = ED.id )
  LEFT JOIN GR18_Deportista D
  ON ( ED.tipoDoc = D.tipoDoc AND ED.nroDoc = D.nroDoc )
  LEFT JOIN GR18_Deportista DD
  ON ( I.tipoDoc = DD.tipoDoc AND I.nroDoc = DD.nroDoc )
  JOIN GR18_Persona P
  ON ( P.tipoDoc = D.tipoDoc AND P.nroDoc = D.nroDoc )
  WHERE C.idCompetencia = $1;
END;
$B$
LANGUAGE 'plpgsql';
SELECT GR18_lista_deportista(12);
```

2. Se debe realizar un reporte que devuelva los equipos en los que participa o participó un deportista.

```
CREATE OR REPLACE FUNCTION GR18_equipos(INTEGER) RETURNS TABLE(c VARCHAR(100), d
timestamp) AS $A$
BEGIN
    RETURN QUERY
    SELECT E.nombre, E.fechaalta
    FROM GR18_equipo E
    JOIN GR18_equipoDeportista ED
```

```
    ON ( E.id = ED.id )
    LEFT JOIN GR18_deportista D
    ON ( ED.tipoDoc = D.tipoDoc AND ED.nroDoc = D.nroDoc )
    WHERE D.nroDoc = $1;
END;
$A$
LANGUAGE 'plpgsql';
SELECT GR18_equipos(20);
```

3. Se debe realizar un reporte que dada la competencia, devuelva la clasificación de la misma.

```
CREATE OR REPLACE FUNCTION GR18_clasificacion(INTEGER) RETURNS TABLE(a
VARCHAR(3), b INTEGER, c VARCHAR(100), d VARCHAR(100), e VARCHAR(50), f INTEGER)
AS $AA$
BEGIN
       RETURN QUERY
        SELECT P.tipodoc, P.nrodoc, P.apellido, P.nombre, CC.puestoGeneral,
CC.puntosGeneral
        FROM GR18_ClasificacionCompetencia CC
        JOIN GR18_Deportista D
        ON ( CC.tipoDoc = D.tipoDoc AND
        CC.nroDoc = D.nroDoc )
        JOIN GR18_Persona P
        ON ( P.tipoDoc = D.tipoDoc AND P.nroDoc = D.nroDoc )
        WHERE idCompetencia = $1
        ORDER BY CC.puestoGeneral ASC;
END;
$AA$
LANGUAGE 'plpgsql';
SELECT GR18_clasificacion(11);
```

4. Dado el documento de un juez, se debe realizar un listado que devuelva en qué competencias ha participado inscripto.

```
CREATE OR REPLACE FUNCTION GR18_jueces(INTEGER) RETURNS TABLE(a VARCHAR(50),b
VARCHAR(500),c timestamp,d VARCHAR(500),e VARCHAR(500)) AS $A$
BEGIN
    RETURN QUERY
    SELECT C.nombre,C.cdodisciplina,C.fecha,C.nombrelugar,C.nombrelocalidad
    FROM GR18_JuezCompetencia JC
    JOIN GR18_Competencia C
      ON (C.idCompetencia = JC.idCompetencia)
    WHERE JC.nroDoc = $1;
END;
$A$
LANGUAGE 'plpgsql';
SELECT GR18_jueces(34587447);
```

DEFINICIÓN DE VISTAS:

1. Se debe realizar una vista que devuelva las competencias ordenada por disciplina

Definicion:

```
CREATE VIEW GR18_competenciasordenadas as(
  SELECT C.* FROM gr18_competencia as C ORDER BY C.cdodisciplina
);
```

2. Se necesita una vista que de todas las competencias que aún no tienen todos los deportistas con
resultados en la clasificación
Definicion:

```
CREATE VIEW GR18_deportistassinclasificar AS
  (SELECT  DISTINCT C.cdodisciplina, C.nombre, C.fecha, C.nombrelugar,
C.nombrelocalidad
    FROM GR18_Competencia C
    JOIN GR18_Inscripcion I
    ON ( C.idCompetencia = I.idCompetencia)
    JOIN GR18_EquipoDeportista E
    ON (E.id = I.Equipo_id)
    WHERE (I.nroDoc, I.tipoDoc) NOT IN ( SELECT nroDoc, tipoDoc FROM
GR18_ClasificacionCompetencia)
  )
  UNION
  (
    SELECT DISTINCT C.cdodisciplina, C.nombre, C.fecha, C.nombrelugar,
C.nombrelocalidad
    FROM GR18_Competencia C
    JOIN GR18_Inscripcion I
    ON ( C.idCompetencia = I.idCompetencia)
    WHERE (I.nroDoc, I.tipoDoc) NOT IN ( SELECT nroDoc, tipoDoc FROM
GR18_ClasificacionCompetencia)
  );
```

3 Crear una vista que devuelve para cada deportista y para cada competencia en la que participó
durante el corriente año, la sumatoria de puntos adquiridos en la general y en su categoría.
Definicion:

```
CREATE VIEW GR18_puntosDeportistas AS(
SELECT
  P.tipodoc, P.nrodoc, P.nombre, P.apellido, SUM(C.puntosCategoria) as
categoria, SUM(C.puntosGeneral) as general, EXTRACT(YEAR FROM CO.fecha)
FROM
GR18_ClasificacionCompetencia C
```

```
JOIN GR18_Persona P
ON (P.tipoDoc = C.tipoDoc AND P.nroDoc = C.nroDoc)
JOIN GR18_competencia as CO
ON (C.idcompetencia = CO.idcompetencia)
WHERE
EXTRACT(YEAR FROM CO.fecha) = EXTRACT(YEAR from CURRENT_TIMESTAMP)
GROUP BY P.tipodoc, P.nrodoc, CO.fecha
);
```