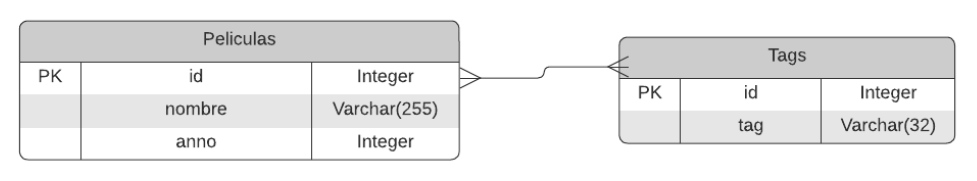
Prueba FINAL- SQL



1. Revisa el tipo de relación y crea el modelo correspondiente. Respeta las claves

primarias, foráneas y tipos de datos.

CREATE TABLE Tags\_Peliculas (ID SERIAL PRIMARY KEY,

ID\_Pelicula int,ID\_Tag int,

FOREIGN KEY (ID\_pelicula) REFERENCES Peliculas(ID),

FOREIGN KEY (ID\_TAG) REFERENCES Tags(ID));

Agregamos tercera columna para realizar de forma correcta la relación.

2. Inserta 5 películas y 5 tags; la primera película debe tener 3 tags asociados, la

segunda película debe tener 2 tags asociados.

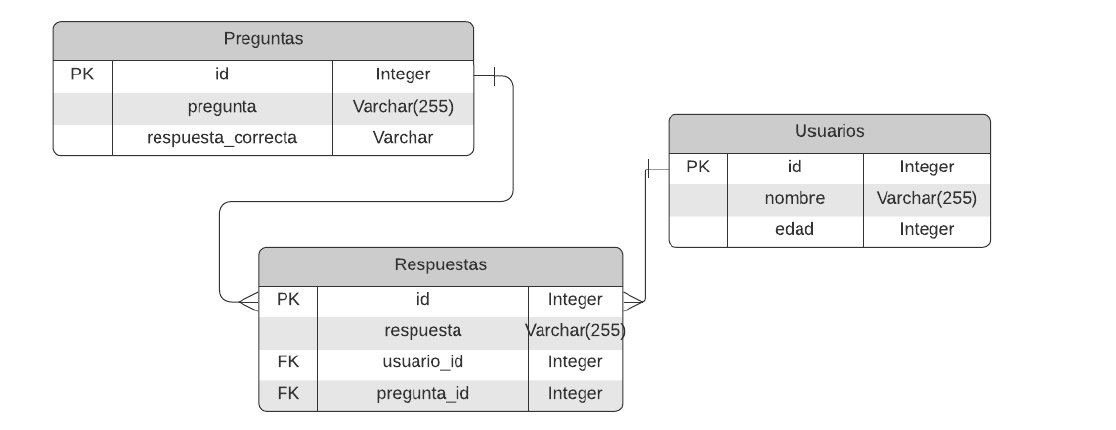
INSERT INTO Pelicula\_Tags (ID\_Pelicula, ID\_Tag) VALUES (1, 1), -- Acción (1, 2), -- Aventura (1, 3); -- Comedia

INSERT INTO Pelicula\_Tags (ID\_Pelicula, ID\_Tag) VALUES (2, 2), -- Aventura (2, 4); -- Drama

3. Cuenta la cantidad de tags que tiene cada película. Si una película no tiene tags debe

mostrar 0.

SELECT p.nombre, COUNT(tp.id\_tag) as Cantidad\_Tags FROM peliculas p LEFT JOIN tags\_peliculas tp ON p.id=tp.id\_pelicula GROUP BY p.nombre,p.id ORDER BY CANTIDAD\_TAGS DESC; el group by es para que las peliculas sean unicas y correspondan al ID



4. Crea las tablas correspondientes respetando los nombres, tipos, claves primarias y

foráneas y tipos de datos.

CREATE TABLE Preguntas (ID INT, pregunta varchar(255), respuesta\_correcta varchar, CONSTRAINT pk\_preguntas PRIMARY KEY (ID));

CREATE TABLE USUARIOS (ID INT, Nombre varchar(255), edad int, CONSTRAINT pk\_usuarios PRIMARY KEY (ID));

CREATE TABLE RESPUESTAS (ID INT, RESPUESTA VARCHAR(255),USUARIO\_ID INT, PREGUNTA\_ID INT

, CONSTRAINT pk\_respuestas PRIMARY KEY (ID),

CONSTRAINT FK\_respuestas\_preguntas FOREIGN KEY (pregunta\_id) REFERENCES PREGUNTAS(ID),

CONSTRAINT FK\_usuario\_respuestas FOREIGN KEY (usuario\_id) REFERENCES USUARIOS(ID));

5. Agrega 5 usuarios y 5 preguntas.

INSERT INTO Usuarios(id,nombre,edad) VALUES (1,'Sebastian',21);

INSERT INTO Usuarios(id,nombre,edad) VALUES (2,'Matias',14);

INSERT INTO Usuarios(id,nombre,edad) VALUES (3,'Paola',22);

INSERT INTO Usuarios(id,nombre,edad) VALUES (4,'Daniela',18);

INSERT INTO Usuarios(id,nombre,edad) VALUES (5,'Uve',77);

INSERT INTO Preguntas (id,pregunta,respuesta\_correcta) VALUES (1,'Quien es el GOAT?','CR7');

INSERT INTO Preguntas (id,pregunta,respuesta\_correcta) VALUES (2,'Why so serius?','IDK');

INSERT INTO Preguntas (id,pregunta,respuesta\_correcta) VALUES (3,'Tortuga ninja antifaz rojo','Raphael');

INSERT INTO Preguntas (id,pregunta,respuesta\_correcta) VALUES (4,'Copas mundiales Chile','Cero');

INSERT INTO Preguntas (id,pregunta,respuesta\_correcta) VALUES (5,'Copas americas Chile ','Dos');

a. La primera pregunta debe estar respondida correctamente dos veces, por dos

usuarios diferentes.

INSERT INTO Respuestas (id,respuesta,usuario\_id,pregunta\_id) VALUES (1,'CR7',1,1);

INSERT INTO Respuestas (id,respuesta,usuario\_id,pregunta\_id) VALUES (2,'CR7',2,1);

b. La segunda pregunta debe estar contestada correctamente solo por un

usuario.

INSERT INTO Respuestas (id,respuesta,usuario\_id,pregunta\_id) VALUES (3,'IDK',4,2);

c. Las otras dos preguntas deben tener respuestas incorrectas

INSERT INTO Respuestas (id,respuesta,usuario\_id,pregunta\_id) VALUES (4,'Leonardo',5,3);

INSERT INTO Respuestas (id,respuesta,usuario\_id,pregunta\_id) VALUES (5,'Cuatro',4,4);

Contestada correctamente significa que la respuesta indicada en la tabla respuestas

es **exactamente igual** al texto indicado en la tabla de preguntas.

6. Cuenta la cantidad de respuestas correctas totales por usuario (independiente de la

pregunta).

SELECT u.nombre, COUNT(r.id) as respuestas\_correctas FROM usuarios u LEFT JOIN respuestas r ON u.id=r.usuario\_id

LEFT JOIN preguntas p ON p.id=r.pregunta\_id WHERE r.respuesta=p.respuesta\_correcta GROUP BY u.nombre;

7. Por cada pregunta, en la tabla preguntas, cuenta cuántos usuarios respondieron

correctamente.

SELECT p.pregunta,count(r.id) as respuestas\_correctas FROM preguntas p LEFT JOIN respuestas r ON p.id=r.pregunta\_id WHERE r.respuesta=p.respuesta\_correcta GROUP BY p.pregunta;

8. Implementa un borrado en cascada de las respuestas al borrar un usuario. Prueba la

implementación borrando el primer usuario.

ALTER TABLE RESPUESTAS DROP CONSTRAINT FK\_usuario\_respuestas;

ALTER TABLE RESPUESTAS ADD CONSTRAINT FK\_Usuario\_respuestas FOREIGN KEY (usuario\_id) REFERENCES USUARIOS(ID) ON DELETE CASCADE

Prueba

DELETE FROM USUARIOS WHERE ID=1;

9. Crea una restricción que impida insertar usuarios menores de 18 años en la base de

datos.

ALTER TABLE USUARIOS ADD CONSTRAINT restriccion\_edad CHECK (edad>=18);

Si me da error debo cambiar a Matias (ID=2)

UPDATE Usuarios SET edad=19 WHERE id=2;

10. Altera la tabla existente de usuarios agregando el campo email. Debe tener la

restricción de ser único.

ALTER TABLE USUARIOS ADD Email varchar(50) UNIQUE;