Bases de Datos Relacionales

 Consultas SQL Avanzadas  
// Práctica Individual

****Primera Parte

Responder las siguientes preguntas:

1. ¿A qué se denomina **JOIN** en una base de datos y para qué se utiliza?

Se le denomina JOIN en una base de datos relacional corresponde a un cláusula que permite combinar registros de dos o más tablas y producir un conjunto de datos resultante de dicha fusión. Se utiliza para traer información relevante de tablas relacionadas entre sí mediante una única consulta.

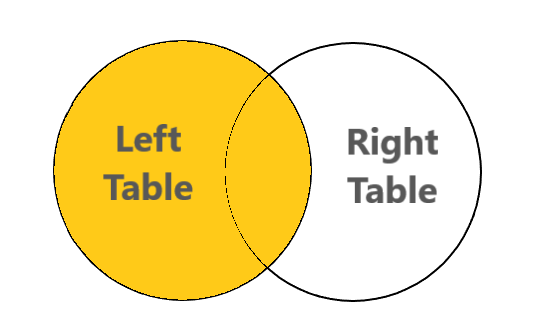
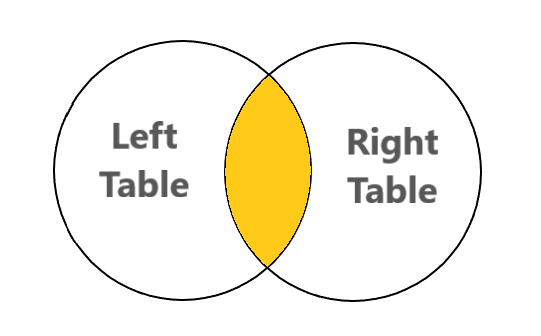
1. Explicar dos tipos de **JOIN**.
   1. INNER JOIN: corresponde a una sentencia SQL que permite combinar registros de dos o más tablas relacionadas entre sí, generando un conjunto de resultados para las filas que tienen coincidencias en ambas tablas. Es un símil a la intersección de conjuntos.
   2. LEFT JOIN: es una sentencia SQL que devuelve todas las filas de la tabla derecha y las coincidencias de la tabla izquierda. Si no hay coincidencias en la tabla izquierda, se devuelve NULL para las columnas correspondientes.
2. ¿Para qué se utiliza el **GROUP BY**?

Esta sentencia SQL se utiliza para reducir la cantidad de registros de una consulta, agrupándolos por valores comunes y aplicando funciones de agregación a cada grupo de filas (promedio, suma, conteo, etc.) de manera tal que se genera un resultado único por grupo.

1. ¿Para qué se utiliza el **HAVING**?

La sentencia GROUP BY agrupa las filas en función de los valores comunes en una o más columnas, mientras que HAVING se utiliza para filtrar los grupos resultantes en función de una condición que debe cumplirse.

1. Escribir una consulta genérica para cada uno de los siguientes diagramas:



SELECT \* FROM left\_table

INNER JOIN right\_table

ON left\_table.id = right\_table.id

SELECT \* FROM left\_table

LEFT JOIN right\_table

ON left\_table.id = right\_table.id



****Segunda Parte

Se propone realizar las siguientes consultas a la base de datos **movies\_db.sql** trabajada en la primera clase.

Importar el archivo **movies\_db.sql** desde ***PHPMyAdmin*** o ***MySQL Workbench*** y resolver las siguientes consultas:

1. Mostrar el título y el nombre del género de todas las series.

SELECT movies.title, genres.name

FROM movies INNER JOIN genres

ON movies.genre\_id = genres.id;



1. Mostrar el título de los episodios, el nombre y apellido de los actores que trabajan en cada uno de ellos.

SELECT series.title AS serie, episodes.title AS episode, actors.first\_name, actors.last\_name

FROM actor\_episode

INNER JOIN actors ON actor\_episode.actor\_id = actors.id

INNER JOIN episodes ON actor\_episode.episode\_id = episodes.id

INNER JOIN seasons ON seasons.id = episodes.season\_id

INNER JOIN series ON series.id = seasons.serie\_id;



1. Mostrar el título de todas las series y el total de temporadas que tiene cada una de ellas.

SELECT series.title, COUNT(\*) AS number\_of\_seasons

FROM seasons

INNER JOIN series ON seasons.serie\_id = series.id

GROUP BY series.id;



1. Mostrar el nombre de todos los géneros y la cantidad total de películas por cada uno, siempre que sea mayor o igual a 3.

SELECT genres.name, count(\*) AS total\_movies

FROM movies

INNER JOIN genres ON movies.genre\_id = genres.id

GROUP BY genres.name

HAVING total\_movies >= 3;



1. Mostrar sólo el nombre y apellido de los actores que trabajan en todas las películas de la guerra de las galaxias y que estos no se repitan.

SELECT DISTINCT actors.first\_name, actors.last\_name

FROM actors

INNER JOIN actor\_movie ON actors.id = actor\_movie.actor\_id

INNER JOIN movies ON movies.id = actor\_movie.movie\_id

WHERE movies.title LIKE 'La Guerra de las %';



A modo de guía para el planteo de consultas, recordar el DER del escenario:

