Resolución de ejercicios

PRIMERA PARTE

1. A qué se denomina **JOIN** en una base datos?

A la acción de juntar datos de tablas (bajo un criterio definido con el ON) para generar un nuevo conjunto de datos en base a las tablas involucradas.

2. Nombre y explique 2 tipos de **JOIN**.

INNER JOIN (que es lo mismo que JOIN) y LEFT JOIN.

El INNER JOIN tomará los datos de las tablas que coinciden bajo el criterio definido en el ON.

El LEFT JOIN tomará los datos de la tabla de la izquierda (la primera) y los datos coincidentes bajo el criterio definido en el ON.

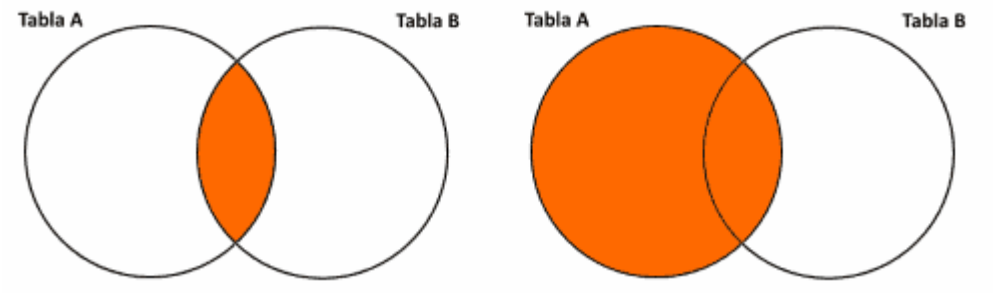
3. Para qué se utiliza el **GROUP BY**?

Para agrupar datos según ciertos campos. Esto permite realizar otras consultas.

4. Para qué se utiliza el **HAVING**?

Se usa con el mismo sentido que el WHERE, sólo que este se utiliza con funciones de agregación (agrupadas con GROUP BY), cosa que con el WHERE no se puede.

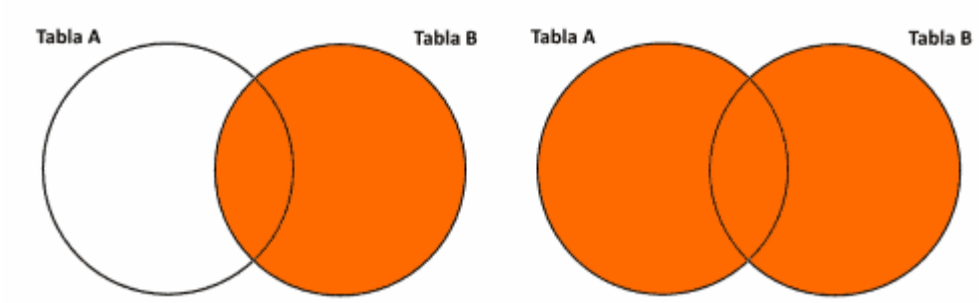
5. Dado lo siguientes diagramas indique a qué tipo de **JOIN** corresponde cada uno:



INNER JOIN y LEFT JOIN.

6. Escriba una consulta genérica por cada uno de los diagramas a

continuación:




```
SELECT *
FROM Tabla1
RIGHT JOIN Tabla2 ON Tabla1.Dato1=Tabl2.Dato2

SELECT *
FROM Tabla1
FULL JOIN Tabla2 ON Tabla1.Dato1=Tabl2.Dato2
```

SEGUNDA PARTE

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

```
SELECT series.title AS 'Título', genres.`name` AS 'Género'
FROM movies_db.series AS series
JOIN movies_db.genres AS genres ON series.genre_id=genres.id;
```

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

```
SELECT episodes.title AS 'Título', actors_list.first_name AS 'Nombre actor', actors_list.last_name AS 'Apellido actor'
FROM movies_db.episodes AS episodes
JOIN (SELECT actors.first_name, actors.last_name, actore.episode_id AS id
        FROM movies_db.actor_episode AS actore
    JOIN movies_db.actors AS actors
    ON actors.id=actore.actor_id
```

```
        ) AS actors_list
ON episodes.id=actors_list.id;
```

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

```
SELECT series.title AS 'Título', COUNT(seasons.serie_id) AS 'Temporadas'
FROM movies_db.series AS series
JOIN movies_db.seasons AS seasons
ON seasons.serie_id=series.id
GROUP BY seasons.serie_id;
```

4. 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 AS 'Género', COUNT(movies.genre_id) AS 'Películas'
FROM movies_db.genres AS genres
JOIN movies_db.movies AS movies
ON genres.id=movies.genre_id
GROUP BY genres.`name`, movies.genre_id
HAVING COUNT(movies.genre_id) > 3;
```

5. 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_list.first_name AS 'Nombre actor', actors_list.last_name AS 'Apellido actor'
FROM movies_db.movies AS movies
JOIN (SELECT actors.first_name, actors.last_name, actore.movie_id AS id
        FROM movies_db.actor_movie AS actore
    JOIN movies_db.actors AS actors
    ON actors.id=actore.actor_id
        ) AS actors_list
```

```
ON movies.id=actors_list.id
WHERE movies.title LIKE '%guerra de las galaxias%';
```

6. Ordenar actores por cantidad de géneros que actuó. Mostrar nombre y apellido del actor, y cantidad de géneros.

```
SELECT actors.first_name,actors.last_name, COUNT(DISTINCT genres.id) AS genero
FROM movies_db.actors AS actors
LEFT OUTER JOIN movies_db.movies AS movies
ON movies.id IN (SELECT actor_movie.movie_id
                                    FROM movies_db.actor_movie AS actor_movie
            WHERE actor_movie.actor_id=actors.id)
LEFT OUTER JOIN movies_db.series AS series
ON series.id IN ( SELECT seasons.serie_id
                                FROM movies_db.seasons AS seasons
          WHERE seasons.id IN (SELECT episodes.season_id
                                                    FROM
movies_db.episodes AS episodes
                            WHERE episodes.id IN
                                                        (SELECT
actor_ep.episode_id
                                                        FROM
movies_db.actor_episode AS actor_ep
                                                        WHERE
actor_ep.actor_id=actors.id)
                                                    )
          )
JOIN movies_db.genres AS genres
```

```
ON genres.id=series.genre_id OR genres.id=movies.genre_id
GROUP BY actors.first_name,actors.last_name
ORDER BY genero DESC;

Versión optimizado
SELECT first_name, last_name, COUNT(DISTINCT genre_id) AS genero
FROM movies_db.actors AS actors
LEFT OUTER JOIN
(SELECT genre_id,actor_id
FROM(
        SELECT movies.genre_id, actor_id
        FROM movies_db.movies AS movies
        JOIN (SELECT actor_movie.movie_id, actor_movie.actor_id
        FROM movies_db.actor_movie ) AS actor_movie ON
actor_movie.movie_id=movies.id
UNION
        SELECT series.genre_id, actor_id
        FROM movies_db.series AS series
        JOIN ( SELECT seasons.serie_id, actor_id
                                FROM movies_db.seasons AS seasons
                                JOIN (SELECT episodes.season_id,
actor_ep.actor_id
                                        FROM movies_db.episodes AS episodes
                                        JOIN (SELECT episode_id,actor_id
                                                FROM
movies_db.actor_episode ) AS actor_ep  ON actor_ep.episode_id=episodes.id
                                ) AS episodes ON
```

```
seasons.id=episodes.season_id
                    ) AS seasons ON seasons.serie_id=series.id
) AS RESULT) AS TEMP ON TEMP.actor_id=actors.id
GROUP BY first_name, last_name
ORDER BY genero DESC;
```