educación IT
CODE YOUR FUTURE

Introducción a Bases de Datos y SQL

Módulo 3 - Resolución del desafío

Resolución del ejercicio 1

1. Generar una lista en la que se muestren **todos los artistas y las canciones** de la tabla ***TOP SPOTIFY***. El nombre de cada artista y el título de la canción deben mostrarse **separados con un guión**, en una columna llamada ***CANCIÓN***.

Utilizar la función ***CONCAT*** para realizar este ejercicio. Ordenar **alfabéticamente** la lista resultante por la columna ***CANCIÓN***. Esta consulta, además, debe mostrar el campo ***GENERO***.

```
SELECT CONCAT(ARTISTA, ' - ', TÍTULO) CANCIÓN, GENERO
FROM TOP_SPOTIFY
ORDER BY CANCION;
```

2. Modificar la consulta anterior para obtener el **mismo resultado** con la función ***CONCAT_WS***.

```
SELECT CONCAT_WS(' - ', ARTISTA, TITULO) CANCION, GENERO
FROM TOP_SPOTIFY
ORDER BY CANCION;
```

3. Modificar la consulta anterior para mostrar los **géneros en mayúsculas**.

```
SELECT CONCAT_WS(' - ', ARTISTA, TITULO) CANCION,
UPPER(GENERO) GENERO
FROM TOP_SPOTIFY
ORDER BY CANCION;
```

4. Agregar a la consulta anterior una **columna** con el nombre ***AÑOS*** en la que se calcule la cantidad de años transcurridos **desde que se lanzó cada una de las canciones al año actual**.

```
SELECT CONCAT_WS(' - ', ARTISTA, TITULO) CANCION,
UPPER(GENERO) GENERO, YEAR(CURDATE()) - ANO AÑOS
FROM TOP_SPOTIFY
ORDER BY CANCION;
```

5. Generar una consulta que calcule la **cantidad de registros** que figuran en la tabla ***TOP SPOTIFY***. El resultado debe mostrarse en una columna con el nombre ***CANCIONES***.

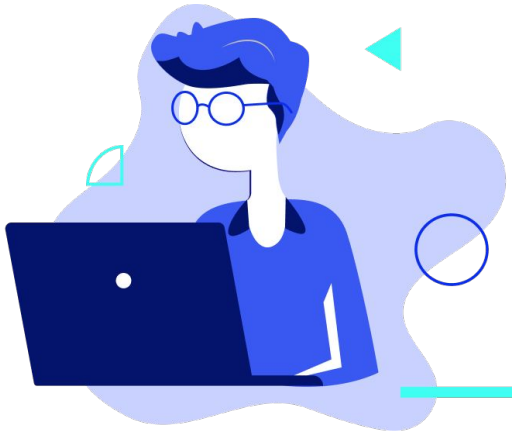
```
SELECT COUNT(*) CANCIONES FROM TOP_SPOTIFY;
```

6. Generar una consulta que muestre la **cantidad de canciones** lanzadas al mercado **por año**.

```
SELECT ANO, COUNT(*) CANCIONES FROM TOP_SPOTIFY
GROUP BY ANO;
```

7. Modificar la consulta anterior para **no mostrar aquellos años** en los que se hayan lanzado **menos de 50** canciones.

```
SELECT ANO, COUNT(*) CANCIONES FROM TOP_SPOTIFY
GROUP BY ANO
HAVING CANCIONES >= 50;
```



Resolución del ejercicio 2

1. Utilizar la base de datos ***LIBRERIA*** y generar una lista en la que se muestren **todos los autores y la provincia en la que nacieron**. El apellido y el nombre de cada autor deben mostrarse en una columna con el nombre ***AUTOR*, separados con una coma y un espacio**. Utilizar la función ***CONCAT*** para llevar a cabo este ejercicio. Ordenar **alfabéticamente** los apellidos resultantes.

```
SELECT CONCAT(APELLIDO, ', ', NOMBRE) AUTOR, PROVINCIA
FROM AUTORES
ORDER BY AUTOR;
```

2. Modificar la consulta anterior para obtener el **mismo resultado** con la función ***CONCAT_WS***.

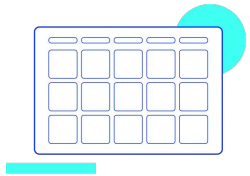
```
SELECT CONCAT_WS(', ', APELLIDO, NOMBRE) AUTOR, PROVINCIA
FROM AUTORES
ORDER BY AUTOR;
```

3. Modificar la consulta anterior para mostrar los **nombres de los autores en mayúsculas**.

```
SELECT UPPER(CONCAT_WS(', ', APELLIDO, NOMBRE)) AUTOR, PROVINCIA
FROM AUTORES
ORDER BY AUTOR;
```

4. Modificar la consulta anterior para mostrar **únicamente la inicial del nombre del autor y su apellido**.

```
SELECT UPPER(CONCAT(APELLIDO, ', ', LEFT(NOMBRE, 1), '.')) AUTOR,
PROVINCIA
FROM AUTORES
ORDER BY AUTOR;
```



5. Generar una columna con el nombre ***INGRESO*** en la que se muestren **todos los empleados y el año en el que ingresaron** a trabajar a la empresa. El apellido y el nombre de cada empleado deben mostrarse **separados con una coma y un espacio** en una columna con el nombre ***EMPLEADO***. Utilizar la función ***CONCAT*** para llevar a cabo este ejercicio. Ordenar el resultado de la consulta **de mayor a menor según los años** de ingreso de cada empleado.

```
SELECT CONCAT(APELLIDO, ', ', NOMBRE) EMPLEADO, YEAR(FECHA_INGRESO) AS INGRESO
FROM EMPLEADOS
ORDER BY INGRESO DESC;
```

Resolución del ejercicio 3

1. Modificar la consulta anterior para **agregar una columna** con el nombre ***ANTIGÜEDAD***. Esta deberá calcular la cantidad de **años de antigüedad de cada empleado** dentro de la empresa al día de hoy.

```
SELECT CONCAT(APELLIDO, ', ', NOMBRE) EMPLEADO,
YEAR(FECHA_INGRESO) AS INGRESO,
TIMESTAMPDIFF(YEAR,FECHA_INGRESO, CURDATE()) AS ANTIGÜEDAD
FROM EMPLEADOS
ORDER BY INGRESO DESC;
```

2. Generar una consulta para obtener el **precio más bajo** de la tabla ***LIBROS***. El resultado se debe mostrar en una columna con el nombre ***MENOR PRECIO***.

```
SELECT MIN(PRECIO) 'MENOR PRECIO' FROM LIBROS;
```

3. Modificar la consulta anterior para **agregar una columna** que calcule el **precio más alto** de la tabla ***LIBROS***. Mostrar el resultado en una columna con el nombre ***MAYOR PRECIO***.

```
SELECT MIN(PRECIO) 'MENOR PRECIO', MAX(PRECIO) 'MAYOR PRECIO' FROM LIBROS;
```

4. Modificar la consulta anterior para **agregar una columna** que calcule el **precio promedio** de todos los libros. Mostrar el resultado en una columna con el nombre ***PRECIO PROMEDIO***. El promedio resultante debe mostrar **2 decimales como máximo**.

```
SELECT MIN(PRECIO) 'MENOR PRECIO', MAX(PRECIO) 'MAYOR PRECIO',
ROUND(AVG(PRECIO), 2) AS 'PRECIO PROMEDIO'
FROM LIBROS;
```

5. Generar una consulta que permita obtener el **precio más bajo, el más alto y el promedio** de los libros pertenecientes a c**ada categoría**.

```
SELECT CATEGORIA, MIN(PRECIO) 'MENOR PRECIO', MAX(PRECIO) 'MAYOR PRECIO',
ROUND(AVG(PRECIO), 2) AS 'PRECIO PROMEDIO' FROM LIBROS GROUP BY CATEGORIA;
```

6. Modificar la consulta anterior para **no mostrar la categoría *"SIN ASIGNAR"***.

```
SELECT CATEGORIA, MIN(PRECIO) 'MENOR PRECIO', MAX(PRECIO) 'MAYOR PRECIO',
ROUND(AVG(PRECIO), 2) AS 'PRECIO PROMEDIO' FROM LIBROS GROUP BY CATEGORIA
HAVING CATEGORIA <> 'SIN ASIGNAR';
```

¡Terminaste el módulo!

Todo listo para rendir el examen