EVALUACIÓN TÉCNICA DE PROGRAMACIÓN DEL GRUPO CASTORES

NOMBRE: RICARDO GARCÍA JIMÉNEZ
CORREO: ricardogj08@riseup.net
TÉLEFONO: 461 235 4175

1. **CONOCIMIENTOS SQL**

1.1) Permite relacionar tablas de bases de datos para generar nuevas vistas utilizando campos que por lo general contienen una llave foránea para unirlas. Una llave foránea hace referencia aun identificador o campo de la tabla que se quiere relacionar.

1.2)

- **CROSS JOIN**: Es el producto cartesiano de las relaciones que intervienen en la combinación (combinar todos con todos).
- **NATURAL JOIN**: Combina dos relaciones de manera natural entre la clave primaria de una y la clave foránea de otra relación.
- **INNER JOIN**: Combina dos relaciones cuando se necesita especificar el (los) atributo(s) con los que se debe combinar.
- **LEFT JOIN**: Combina dos relaciones realizando primero un INNER JOIN y luego añade todas aquellas filas (registros) de la relación nombrada (tabla) en la parte izquierda de la sentencia que no satisfacen la combinación con la relación de la parte derecha.
- **RIGHT JOIN**: Combina dos relaciones realizando primero un INNER JOIN y luego añade todas aquellas filas (registros) de la relación nombrada (tabla) en la parte derecha de la sentencia que no satisfacen la combinación con la relación de la parte izquierda.
- **FULL JOIN**: Combina dos relaciones realizando primero un INNER JOIN y luego realiza un LEFT JOIN y por último un RIGHT JOIN.

1.3) Son acciones automáticas que se realizan cuando sucede un evento en particular para automatizar tareas repetitivas durante la inserción, modificación, eliminación o consulta de datos.

1.4) Son sentencias SQL reutilizables que se almacenan en la base de datos para realizar un tarea en particular.

1.5)

```
SELECT
   productos.idProducto,
   productos.nombre,
   COUNT(ventas.idVenta) AS numeroVentas
FROM productos
INNER JOIN ventas
   ON productos.idProducto = ventas.idProducto
GROUP BY productos.idProducto
HAVING COUNT(ventas.idVenta) = 1
ORDER BY productos.idProducto ASC;
```

1.6)

```
SELECT
   productos.idProducto,
   productos.nombre,
   SUM(ventas.cantidad) AS cantidadTotalVentas
FROM productos productos
```

```
INNER JOIN ventas
   ON productos.idProducto = ventas.idProducto
GROUP BY productos.idProducto
ORDER BY productos.idProducto ASC;
```

1.7)

```
SELECT
   productos.idProducto,
   productos.nombre,
   productos.precio,
   IFNULL(SUM(ventas.cantidad), 0) AS cantidadTotalVentas,
   (IFNULL(SUM(ventas.cantidad), 0) * productos.precio) AS sumaTotalVentas
FROM productos productos
LEFT JOIN ventas
   ON productos.idProducto = ventas.idProducto
GROUP BY productos.idProducto
ORDER BY productos.idProducto ASC;
```

3. **EJERCICIO PRÁCTICO: DESARROLLO**

3.1) https://github.com/ricardogj08/inventario-castores

3.1.1. https://notabug.org/ricardogj08/inventario-castores

2. **EJERCICIO PRÁCTICO: BD**

2.1) https://github.com/ricardogj08/inventario-castores/blob/master/docs/database.png

2.2) https://github.com/ricardogj08/inventario-castores/tree/master/SCRIPTS

4.**DOCUMENTACIÓN:**

4.1) https://github.com/ricardogj08/inventario-castores

4.2) https://github.com/ricardogj08/inventario-castores/blob/master/README.md

4.3) https://drive.google.com/file/d/14eLwQro6MeM78pyrg7BOhsOjh3UopNS9/view?usp=sharing