Práctica Grupal Base de Datos

(Eduard, Alex, Ernesto)

1. **Recupera la descripción de la aplicación de ED y la base de datos diseñada para la aplicación**

https://aulavirtual33.educa.madrid.org/ies.calderon.pinto/mod/folder/view.php?id=86728

Pensando en la funcionalidad de la aplicación:

2. **Crea 2 consultas monotabla que se ejecuten regularmente en la aplicación. Por ejemplo, si al conectarnos la app nos muestra las ofertas de los productos “SELECT ofertas FROM productos;”**

Eduard:

- Consulta para obtener el precio promedio de los productos por fabricante:
  SELECT fabricante, AVG(precio) AS precio_promedio
  FROM tiendas
  GROUP BY fabricante;

- Consulta para encontrar los productos con el menor stock:
  SELECT producto, stock
  FROM tiendas
  WHERE stock = (SELECT MIN(stock) FROM tiendas);

Alex:

- Consulta para llamar al procedimiento insertar_productos del ejercicio 9:

  CALL insertar_productos (nombre, X€ , X , fabricante);

  - Consulta mostrar usuarios de la tabla grupo

    SELECT usuarios
    FROM grupo

Ernesto:

- Consulta para obtener los usuarios que tengan una cuenta premium

  SELECT contraseña FROM usuarios WHERE premium = true;

3. **Igual que en el apartado anterior, crea 2 consultas multitablas que se ejecutarán de manera frecuente en la aplicación (JOIN)**

Eduard:

- Consulta para obtener información detallada de los productos en stock por fabricante:
  SELECT t.producto, t.precio, t.stock, t.fabricante, u.nombre AS fabricante_nombre
  FROM tiendas t
  JOIN usuarios u ON t.fabricante = u.nombre;

- Consulta para obtener el nombre de usuario y su historial de compras:
  SELECT u.nombre AS usuario, t.producto, t.precio, t.fecha
  FROM tiendas t
  JOIN usuarios u ON t.usuario = u.nombre;

Alex:

- Consulta para relacionar los mensajes realizados con el correo electrónico del usuario

SELECT mensajes
FROM usuarios u
JOIN conversaciones c ON u.correo = c.mensajes;

- Consulta para relacionar las empresas y los productos que han subido para vender en nuestra web

SELECT codigo_empresa
FROM empresa e
JOIN tiendas t ON e.codigo_empresa=t.empresa

Ernesto:
1- Consulta para obtener los mensajes de usuarios premium:

SELECT g.mensajes FROM grupo g
JOIN usuarios u ON u.nombre = g.nombre
WHERE u.premium = true;

2.- Recuperar el correo de un usuario que escribió un mensaje este año:
SELECT c.mensaje FROM conversaciones c
JOIN usuarios u ON u.nombre = c.usuario
WHERE YEAR(c.fecha) = YEAR(curdate());

4. **Crea 2 inserciones, 2 modificaciones y 2 eliminaciones frecuentes que se llevarán a cabo en la aplicación.**

Eduard:

- Inserción de un nuevo producto en la tabla tiendas:
  INSERT INTO tiendas (producto, precio, stock, fabricante)
  VALUES ('Camiseta deportiva', 25.99, 100, 'Nike');

- Inserción de un nuevo usuario en la tabla usuarios:
  INSERT INTO usuarios (nombre, contraseña, premium, correo)
  VALUES ('Eduard', 'contraseña123', 0, 'eduard@gmail.com');

- Actualización del precio de un producto existente en la tabla tiendas:
  UPDATE tiendas
  SET precio = 29.99
  WHERE producto = 'Camiseta deportiva';

- Cambio del estado de premium de un usuario en la tabla usuarios:
  UPDATE usuarios
  SET premium = 1
  WHERE nombre = 'eduard';
- Eliminación de un producto agotado de la tabla tiendas:
  DELETE FROM tiendas
  WHERE stock = 0;

- Eliminación de un usuario que ya no está activo en la tabla usuarios:
  DELETE FROM usuarios
  WHERE nombre = 'eduard';

Alex:

- INSERCIONES:
- Insertar una nueva empresa en la tabla empresas

  INSERT INTO empresas
  VALUES(“codigo_empresa”,”nombre”,”Ubicación”,”Teléfono” , “Correo”);

- Insertar un producto

  INSERT INTO producto
  VALUES (“Nombre”,”Precio”,”Stock”,”Fabricante”);

- MODIFICACIONES

- Modificación de un precio de producto

  UPDATE tiendas SET precio= ”nuevo” WHERE precio = ”20€”

- ELIMINACIONES
- Eliminar productos que ya no se quieran vender
  DELETE FROM tiendas
  WHERE productos = ’nombre’;

Ernesto:
– INSERCIONES:
- Insertar el mensaje de un usuario en una conversación
  INSERT INTO conversaciones VALUES(“nombreusuario”, “mensaje”,“fecha”);

- Insertar el mensaje de un usuario en una conversación grupal
  INSERT INTO grupo VALUES(“nombreusuario”, “mensaje”,“fecha”);

– MODIFICACIONES:
- Modificación de un mensaje de un usuario
  UPDATE conversaciones SET mensaje = "nuevo" WHERE usuario = "nombre";

- Modificación del mensaje en un grupo
  UPDATE grupo SET mensajes = "nuevos" WHERE usuarios = "nombres";

  – ELIMINACIONES:
- Eliminar conversaciones del año pasado
  DELETE FROM conversaciones WHERE year(fecha) < year(curdate());
- Eliminar grupos del año pasado
  DELETE FROM grupo WHERE year(fecha) < year(curdate());

5. **Imaginaros que vienen dos personas nuevas a vuestra empresa, os ayudarán a mejorar aspectos de la aplicación ¿Qué vistas le proporcionaríais teniendo en cuenta las funciones que van a desarrollar (define también qué función desarrollarán)? crea mínimo 2 vistas.**
   Eduard:

   Desarrollador Frontend:
   - Esta persona se encarga de mejorar la interfaz de usuario y la experiencia del cliente en la aplicación web.
   - Una vista útil sería que muestre estadísticas sobre el uso de la aplicación por parte de los usuarios, que podría incluir información como la cantidad de usuarios activos, la duración promedio de la sesión, las páginas más visitadas…

   Analista de datos:
   - Esta persona se encargará de analizar los datos de la aplicación para identificar tendencias, patrones y oportunidades de mejora.
   - Una vista que podría ser útil para este rol es una vista que muestre un resumen de las ventas y el rendimiento de los productos en la aplicación, incluyendo gráficos, tablas…

   Alex:

   Operador de sistemas informáticos:
   - Esta persona se encargará de administrar servidores , redes locales de nuestra empresa, incidencias en equipos informáticos.
   - Un tipo de rol que puede ser muy ventajoso sobre todo teniendo en cuenta la necesidad de la rapidez en caso de una caída de red o gestión de servidor. Para que la aplicación funcione al 100%

   Manager de contenido digital:
   - Esta persona se encargará de dar visualización a nuestra app, la manera más efectiva de llegar a más gente y el tipo de gente al que llegar.

   Ernesto:
   Gerente de Tienda Asociada:
   - Recibirá información de las ventas realizadas en nuestra aplicación y se encargará de enviar ofertas.
   - Necesitaria una vista que muestre las ventas de su tienda, además de que se muestre cual es el producto con la mayor cantidad de ventas recientes para poder decidir que rebajar.

   Administrador de incidencias:

- Recibirá avisos de usuarios reportados y podrá examinar los mensajes enviados por dichos usuarios que incumplan las normas.
- La vista deberá incluir todos los mensajes del usuario, tanto en conversaciones individuales como grupales.

6. **Define qué usuarios crearías para que puedan acceder al SGBD y qué permisos tendrán.**

Eduard:

Administrador de la base de datos:

- Este usuario tiene acceso completo y control total sobre la base de datos.
- Tiene permisos para crear, modificar y eliminar bases de datos, tablas, índices, usuarios…
- Puede realizar tareas de mantenimiento y configuración del servidor de base de datos.
- Tiene privilegios de administración, como CREATE, ALTER ,DROP…

Desarrollador de aplicaciones:

- Este usuario tiene acceso para desarrollar y probar aplicaciones que utilizan la base de datos.
- Tiene permisos para crear y modificar tablas, índices…
- Puede insertar, actualizar y eliminar datos de las tablas de la aplicación.
- Tiene privilegios de SELECT, INSERT, UPDATE, DELETE, CREATE , ALTER….

Alex:
Operador de sistemas informáticos:

- Esta persona debería acceder a el usuario administrador del sistema teniendo en cuenta su rol, el poder crear y eliminar usuarios y dar y quitar permisos.

Manager de contenido digital:

- Esta persona debería poder ver los datos de nuestra base de datos de la tabla ventas , la interacción de los usuarios con las publicaciones y el perfil de los usuarios.

Ernesto:

Gerente de tienda asociada:

- Este usuario deberá poder publicar productos y modificar precios
  Tiene permisos de insertar, modificar y consultar los productos y sus ventas (SELECT, INSERT, UPDATE Y DELETE)

Administrador de incidencias:

- Este usuario necesitará ver conversaciones en sus tablas respectivas, y luego poder eliminar usuarios en caso de que alguno rompa demasiadas normas:
  Tiene permisos de SELECT en las tablas conversaciones y grupos, y
  Permisos de DELETE de la tabla usuarios.

7. **¿Has detectado alguna consulta que se realiza constantemente? ¿echas de menos algún índice? por ejemplo, si se busca siempre información por ciudad sería recomendable crear un índice en el campo ciudad.**

Eduard:

- Consulta para buscar productos por nombre:
  Si los usuarios buscan productos por su nombre con frecuencia, sería recomendable crear un índice en el campo PRODUCTO de la tabla TIENDAS

- Consulta para filtrar productos por fabricante:
  Si los usuarios suelen buscar productos de un fabricante específico, sería útil crear un índice en el campo FABRICANTE de la tabla TIENDAS

Alex:

- Consulta para filtrar el correo electrónico de los usuarios en caso de querer ponernos en contacto con ellos.
- El índice que pondría sería en caso de haber una tabla empresas crear codigo_empresa.

Ernesto:

- -Consulta de los mensajes publicados por usuarios específicos
- Sería recomendable que los usuarios sirvan de índice en las tablas de conversión y grupo.

8. **¿Qué triggers pensáis que necesita la base de datos?, crea 2 trigger por tabla.**

<u>Triggers para la tabla TIENDAS :</u>

- **Registro de cambios en el stock de productos:**
  CREATE TRIGGER tr_stock_cambios
  AFTER UPDATE ON tiendas
  FOR EACH ROW

```
BEGIN
  IF OLD.stock <> NEW.stock THEN
    INSERT INTO registro_stock (producto, stock_anterior,
stock_actual, fecha_cambio)
    VALUES (NEW.producto, OLD.stock, NEW.stock, NOW());
  END IF;
END;
```

- **Validación de precios de productos:**

```
CREATE TRIGGER tr_precio_productos
BEFORE INSERT ON tiendas
FOR EACH ROW
BEGIN
  IF NEW.precio <= 0 THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'El precio tiene que ser mayor que cero';
  END IF;
END;
```

Triggers para la tabla USUARIOS

- **Registro de nuevas cuentas de usuario:**

```
CREATE TRIGGER tr_nuevo_usuario
AFTER INSERT ON usuarios
FOR EACH ROW
BEGIN
  INSERT INTO registro_usuarios (nombre_usuario, fecha_creacion)
  VALUES (NEW.nombre, NOW());
END;
```

- **Registro de cambio de contraseña**

```
CREATE TRIGGER cambio_contraseña
AFTER UPDATE ON usuarios
FOR EACH ROW
INSERT INTO historialContraseñas(usuario, viejaContraseña,
nuevaContraseña)
VALUES(NEW.nombre, OLD.contraseña, NEW.contraseña);
```

Triggers para la tabla GRUPO

-Trigger que almacena un historial de ediciones de los mensajes de los usuarios

```
CREATE TRIGGER cambio_mensaje
AFTER UPDATE ON grupo
 FOR EACH ROW
 INSERT INTO historialMensaje(usuario, mensaje, nuevoMensaje, fecha)
 VALUES(NEW.usuario, OLD.mensaje, NEW.mensaje, NEW.fecha);

CREATE TRIGGER nuevo_evento
AFTER UPDATE ON grupo
 FOR EACH ROW
 INSERT INTO historialContraseñas(evento, fecha)
 VALUES(NEW.evento, DATE(curdate()));
```

Triggers para la tabla CONVERSACIONES

-Trigger que almacena un historial de conversaciones abiertas

```
CREATE TRIGGER nueva_conversacion
AFTER INSERT ON conversaciones
 FOR EACH ROW
 INSERT INTO historialConversaciones(usuario, abiertaOcerrada)
 VALUES(OLD.USUARIO, "ABIERTA");
```

-Trigger que almacena un historial de conversaciones cerradas

```
CREATE TRIGGER fin_conversacion
AFTER DELETE ON conversaciones
 FOR EACH ROW
 INSERT INTO historialConversaciones (usuario, abiertaOcerrada)
 VALUES(OLD.usuario, “Cerrada”);;
```

9. **¿Qué procedimientos almacenados podría necesitar la base de datos?, crea 2 procedimientos almacenados.**

Eduard:

- Procedimiento para actualizar el stock de un producto:
  CREATE PROCEDURE actualizar_stock(

```
    IN producto_nombre VARCHAR(50),
    IN cantidad INT
)
BEGIN
    UPDATE tiendas
    SET stock = stock + cantidad
    WHERE producto = producto_nombre;
END;
```

- Procedimiento para obtener el historial de compras de un usuario:

```
CREATE PROCEDURE historial_compras(
    IN usuario_nombre VARCHAR(50)
)
BEGIN
    SELECT t.producto, t.precio, t.fecha
    FROM tiendas t
    JOIN usuarios u ON t.usuario = u.nombre
    WHERE u.nombre = usuario_nombre;
END;
```

Alex:

- Procedimiento para insertar nuevos productos a la tabla tiendas:

Procedimiento :

```
DELIMITER //
CREATE PROCEDURE insertar_productos (
        p_producto      VARCHAR(30),
    p_precio float,
    p_stock int ,
    p_fabricante VARCHAR(30)

    )
    BEGIN
                INSERT INTO tiendas (producto,precio,stock,fabricante)
       VALUES (p_producto, p_precio,p_stock,p_fabricante);
        END //
    DELIMITER ;
```

- Procedimiento para insertar nuevas empresas en la tabla empresas:

```
DELIMITER //
CREATE PROCEDURE insertar_empresas (
   e_empresa  VARCHAR(50),
   e_ubicacion VARCHAR(50),
   e_telefono int ,
   e_correo VARCHAR(50)
   )
   BEGIN
        INSERT INTO empresas (empresa,ubicacion,telefono,correo)
     VALUES (e_empresa, e_ubicacion,e_telefono,e_correo);
        END //
   DELIMITER ;
```

Ernesto:

Procedimiento para crear un nuevo usuario:

```
CREATE PROCEDURE crear_usuario(
        u_nombre VARCHAR(30),
   u_contraseña VARCHAR(30),
   u_premium BOOLEAN,
   u_correo VARCHAR(30)
)
BEGIN
        INSERT INTO usuarios(nombre,contraseña,premium,correo)
VALUES(u_nombre, u_contraseña, u_premium, u_correo);
   END //
   DELIMITER;
```

Procedimiento para publicar un mensaje:

```
CREATE PROCEDURE crear_mensaje(
        c_usuario VARCHAR(30),
   c_mensaje VARCHAR(100),
   c_fecha DATE

)
BEGIN
        INSERT INTO conversaciones(usuario,mensaje,fecha) VALUES(c_usuario,
c_mensaje, c_fecha);
   END //
   DELIMITER;
```

10. **Realiza una presentación del trabajo para compartir con los compañeros (tenéis que presentarla entre todos los componentes del grupo)** o **crea un repositorio en Github con las siguientes secciones completadas**
    a. Descripción de la aplicación y diseño de la BBDD
    b. Consultas monotablas (Si sois cuatro 8 consultas, 2 cada uno de vosotros)
    c. Consultas multitabla
    d. Inserciones, modificaciones y eliminación de datos
    e. Vistas
    f. Usuarios y permisos
    g. Índices
    h. Triggers
    i. Procedimientos almacenados