**PROCEDURE DE ELIMINACIÓN**

1)  **procedure EliminarEmpleado** (in idEmpleado char(10)): Se encarga de llamar a algunos de los procedure de eliminación de los distintos tipos de empleado, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

2)  **procedure EliminarRolDePago** (in id\_Rol\_de\_pago int): Se encarga de eliminar un rol de pago según el id que se introduzca del rol de pago que se quiere eliminar, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

3)  **procedure EliminarOperario** (in idOperario char(10)): Se encarga de eliminar un Operario, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

4)  **procedure EliminarAsistente** (in idAsistente char(10)): Se encarga de eliminar un asistente de operario, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

5)  **procedure EliminarSecretaria** (in idSecretaria char(10)): Se encarga de eliminar una secretaria, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

6) **procedure EliminarMantenimiento** (in idMantenimiento int): Se encarga de eliminar un mantenimiento según el id que se introduzca del mantenimiento que se quiere eliminar, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

7) **procedure EliminarMaquinaria** (in idMaquinaria int): Se encarga de eliminar una maquinaria según el id que se introduzca de la maquinaria y el mantenimiento donde su código es null, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

8) **procedure EliminarRegistro** (in idAsistente char(10),in idSecretaria char(10),in idLimpieza int): Se encarga de eliminar un registro según que coincida con el id del asistente, el id de la secretaría, y el id de la limpieza a la vez, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

9) **procedure EliminarLimpieza** (in idLimpieza int): Se encarga de eliminar una limpieza según el id que se introduzca de la limpieza y los registros que contengan el id 999999, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

10) **procedure EliminarReclamacion** (in idReclamacion int): Se encarga de eliminar una reclamación según el id que se introduzca de la reclamación, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

11) **procedure EliminarCliente** (in idCliente varchar(13)): Se encarga de eliminar un cliente según el id del cliente que se introduzca, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

12) **procedure EliminarFactura** (in idFactura int): Se encarga de eliminar una factura según el id que se introduzca de la factura y los detalles que contengan el id 999999, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

13) **procedure EliminarDetalle** (in idFactura int,in idProducto varchar(6)): Se encarga de eliminar un detalle según que coincida con el id de la factura y el id del producto a la vez, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

14) **procedure EliminarProducto** (in idProducto varchar(6)): Se encarga de eliminar un producto según el id que se introduzca del producto, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

15) **procedure Eliminarlote** (in idLote int): Se encarga de eliminar un lote según el id que se introduzca del lote y las especificaciones que contengan el id 999999, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

16) **procedure EliminarProveedor** (in idProveedor char(10)): Se encarga de eliminar un proveedor según el id que se introduzca del proveedor, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

17) **procedure EliminarEvaluacion** (in idEvaluacion int): Se encarga de eliminar una evaluación a un proveedor según el id que se introduzca de la evaluación, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

18)  **procedure EliminarEspecificacion** (in idLote int,in idMadera varchar(6)): Se encarga de eliminar una especificación según el id del lote y el id del tipo de madera que se ingrese, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

19) **procedure EliminarMadera** (in idMadera varchar(6)): Se encarga de eliminar un tipo de madera según el id que se introduzca del tipo de madera, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

**PROCEDURE DE ACTUALIZACIÓN**

1)  **procedure actualizarEmpleado**(in idEmpleado char(10), in nomb varchar(40),in horaI time,in horaF time,in fechaCapa date,in tipoCapa varchar(20)): Se ingresan todos los datos del empleado para actualizarlos en la tabla empleado y en la tabla respectiva individual del empleado según su rol utilizando el id del empleado como filtro para las tuplas y dependiendo de este llama a alguno de los procedure de actualización de los distintos tipos de empleado, y en caso de que el id ingresado no exista lanzará un mensaje que el ID es inexistente

2) **procedure ActualizarDetalle**(in idFactura int,in idProducto varchar(6),in unidades\_nuevas int,in nuevo\_detalle\_adic varchar(100)): Se ingresan todos los datos del detalle para actualizarlos en la tabla detalle utilizando el id de la factura y el id del producto como filtro para las tuplas, en caso de que alguno de los dos no exista se lanzará un mensaje de que no existe y en caso de que las unidades no sean positivas también se lanzará un mensaje que las unidades no son válidas

3)  **procedure ActualizarProducto**(in idProducto varchar(6),in nuevoNombre varchar(30),in nuevoPrecioUnitario float,in nueva\_calidad varchar(10),in nueva\_cond\_amb varchar(20), in nuevo\_tiempo\_almac int,in nueva\_dimension varchar(20),in nueva\_desc varchar(100)): Se ingresan todos los datos del producto para actualizarlos en la tabla producto utilizando el id del producto como filtro para las tuplas,y en caso de que el id no exista se lanzará un mensaje que el id del producto no existe

4) **procedure ActualizarFactura**(in idFactura int,in NuevaIdSecretaria char(10),in NuevoIdCliente char(10),in NuevaFecha date,in NuevaHora time,in NuevaDireccion varchar(30),in metodoPago enum('Efectivo','Transferencia'),in NuevoSubtotalSinImpuestos float,in Nuevosubtotal\_0PorCiento float): Se ingresan todos los datos de la factura para actualizarlos en la tabla factura utilizando el id de la factura como filtro para las tuplas, en caso de que el método de pago no sea efectivo o transferencia se lanzará un mensaje de que el método de pago es inválido, y si el id de la factura o el de la secretaría o el id del cliente no existe se lanzará un mensaje de que el id no existe

5) **procedure ActualizarCliente**(in cedulaCliente varchar(13),in NuevoNombre varchar(30),in direccionNueva varchar(50),in NuevoNumContacto int,in NuevoCorreo varchar(20)): Se ingresan todos los datos del cliente para actualizarlos en la tabla Cliente utilizando el ID del cliente como filtro para las tuplas, en caso de que el ID del cliente no exista se lanzará un mensaje de que la cédula no se encuentra en la base de datos

6) **procedure ActualizarReclamacion** (in idEspecifico int,in idSecretaria char(10),in cedulaCliente varchar(13),in descripcion\_nueva varchar(100)): Se ingresan todos los datos de la reclamación para actualizarlos en la tabla Reclamacion utilizando el ID de la reclamación como filtro para las tuplas, en caso de que el ID de la reclamación, secretaría o la cédula del cliente no exista, se lanzará un mensaje de que el ID no se encuentra existente

7)  **procedure actualizarMantenimiento**(in idNuevo int, in idOperario char(10),in codMaquinaria int, in idSecretaria char(10),in NuevosDetalles varchar(100),fechaNueva date): Se ingresan todos los datos del mantenimiento para actualizarlos en la tabla Mantenimiento utilizando el ID del mantenimiento como filtro para las tuplas, en caso de que el ID del mantenimiento, operario,código de la maquinaria o ID no exista, se lanzará un mensaje de que el ID no se encuentra existente

8) **procedure actualizarOperario**(in idOperario char(10), in nomb varchar(40),in horaI time,in horaF time,in fechaCapa date,in tipoCapa varchar(20)):Se ingresan todos los datos del operario para actualizarlos en la tabla Operario utilizando el ID del operario como filtro para las tuplas, en caso de que el ID del mantenimiento, operario,código de la maquinaria o ID no exista, se lanzará un mensaje de que el ID no se encuentra existente

9) **procedure actualizarMaquinaria**(in codig int,in nomb varchar(20),in mar varchar(20),in fechaAdqui date): Se ingresan todos los datos de la maquinaria para actualizarlos en la tabla Maquinaria utilizando el código de la maquinaria como filtro para las tuplas, en caso de que el código de la maquinaria no exista se lanzará un mensaje de que el código no se encuentra existente

10) **procedure actualizarRegistro**(in idAsist char(10),idLimpieza int,idSecretaria char(10),in fech date): Se ingresan todos los datos del registro para actualizarlos en la tabla Registro utilizando el ID del asistente, el ID de la limpieza y el ID de la secretaría como filtro para las tuplas, en caso de que alguno de estos IDs no exista se lanzará un mensaje de que el ID no se encuentra existente

11) **procedure actualizarAsistente**(in idAsistente char(10), in nomb varchar(40),in horaI time,in horaF time,in fechaCapa date,in tipoCapa varchar(20)): Se ingresan todos los datos del asistente para actualizarlos en la tabla Asistente\_operario utilizando el ID del asistente como filtro para las tuplas, en caso de que el ID del asistente no exista se lanzará un mensaje de que el ID no se encuentra existente

12) **procedure actualizarLimpieza**(in idLimpieza int, in place varchar(30)): Se ingresan todos los datos de la limpieza para actualizarlos en la tabla Limpieza utilizando el ID de la Limpieza como filtro para las tuplas, en caso de que el ID no exista se lanzará un mensaje de que el ID no se encuentra existente

13) **procedure actualizarSecretaria**(in idSecretaria char(10), in nomb varchar(40),in horaI time,in horaF time,in fechaCapa date,in tipoCapa varchar(20)): Se ingresan todos los datos de la secretaría para actualizarlos en la tabla Secretaria utilizando el ID de la Secretaria como filtro para las tuplas, en caso de que el ID no exista se lanzará un mensaje de que el ID no se encuentra existente

14)**procedure** **actualizarProveedor:** Este procedimiento actualiza la información del proveedor en la tabla proveedor utilizando la cédula (p\_cedu), el nuevo nombre (p\_nomb), y el nuevo teléfono (p\_telef). Si la cédula proporcionada no existe en la tabla proveedor, se lanza un mensaje de error con 'Proveedor no existente'. Si la cédula existe, se actualizan los detalles del proveedor en la tabla proveedor. En caso de que el proveedor no sea encontrado, se realiza un rollback; de lo contrario, se realiza un commit.

**15)ActualizarLoteMadera**: Este procedimiento actualiza la información de un lote de madera en la tabla lote\_madera utilizando el ID del lote (idLote), el ID del proveedor (idProveedor), el ID de la secretaria (idsecret), el nuevo precio (precioNuevo), y la nueva fecha (fechaNueva). Si el ID del lote, el ID del proveedor, o el ID de la secretaria no existen, se lanzan errores correspondientes. Si todos los IDs son válidos, se actualizan los detalles del lote de madera; si no, se realiza un rollback, y si todo es correcto, un commit.

**16)ActualizarEspecificacion**: Este procedimiento actualiza la especificación de un lote de madera en la tabla especificacion utilizando el ID del lote (idLote), el ID de la madera (idMadera), y la nueva cantidad (cantidadNueva). Verifica la existencia del ID del lote y el ID de la madera, y que la cantidad nueva no sea negativa. Si alguno de estos no cumple con los requisitos, se lanza un error; si todo es correcto, se actualizan los detalles de la especificación.

**17)ActualizarTipoMadera**: Este procedimiento actualiza la información del tipo de madera en la tabla tipo\_de\_madera, utilizando el ID de la madera (idMadera), el nuevo nombre (NuevoNombre), el nuevo precio unitario (precUnit), y las condiciones ambientales (condicAmb). Verifica que el ID de la madera exista en la tabla; si no existe, se lanza un error. Si el ID es válido, se actualizan los detalles del tipo de madera; si no, se realiza un rollback, y si todo es correcto, un commit.

**18)ActualizarRolDePago**: Este procedimiento actualiza la información del rol de pago en la tabla rol\_de\_pagos, utilizando el ID del rol (idNuevo), el ID del empleado (idEmpleado), los días laborados (laborados), el sueldo nuevo (sueldoNuevo), las horas extras (horasExtras), el total de ingresos (totalIngresos), los egresos del IEES (EgresosIEES), los anticipos (NuevosAnticipos), y el total de egresos (totalEgresos). Verifica la existencia del ID del rol y el ID del empleado. Si alguno no existe, se lanza un error. Si ambos IDs son válidos, se actualizan los datos del rol de pago; si no, se realiza un rollback, y si todo es correcto, un commit.

**19)actualizarRegistro**: Este procedimiento actualiza los datos de un registro en la tabla Registro, utilizando el ID del asistente (idAsist), el ID de la limpieza (idLimpieza), el ID de la secretaría (idSecretaria), y la fecha (fech). Se ingresan todos los datos del registro para actualizarlos en la tabla Registro utilizando el ID del asistente, el ID de la limpieza y el ID de la secretaría como filtro para las tuplas. Si alguno de estos IDs no existe, se lanza un mensaje de error indicando que el código no se encuentra existente.

**PROCEDURE DE INSERCIÓN**

1) PROCEDURE InsertCliente(IN p\_cedula VARCHAR(13),IN p\_nombre VARCHAR(30),IN p\_direccion VARCHAR(100),IN p\_num\_contacto INT,IN p\_correo\_contacto VARCHAR(100)): Se encarga de insertar un nuevo cliente en la tabla cliente, en caso de que la cédula que se inserte ya exista se lanzará un mensaje que dice que la cédula ya existe

2) PROCEDURE InsertProducto(IN p\_ID VARCHAR(6),IN p\_nombre VARCHAR(30),IN p\_precioUnitario FLOAT,IN p\_calidad ENUM('De primera', 'De segunda'),IN p\_condic\_ambiental VARCHAR(20),IN p\_tiempo\_almacenamiento INT,IN p\_dimension VARCHAR(20),IN p\_descripcion VARCHAR(100)): Se encarga de insertar un nuevo producto en la tabla productos, en caso de que el id que se inserte ya exista se lanzará un mensaje mencionando que ese id ya existe, en caso de que la calidad no sea de primera o de segunda se lanzará un mensaje de que la calidad fue insertada de forma incorrecta

3) PROCEDURE InsertSecretaria(IN p\_ID CHAR(10),IN p\_nombre VARCHAR(40),IN p\_horaInicio TIME,IN p\_horaFine TIME,IN p\_fechaCapacitacion DATE,IN p\_tipoCapacitacion VARCHAR(20)): Se encarga de insertar una nueva secretaría en la tabla secretaria y en la tabla empleado, en caso de que el id de la secretaría ya exista se lanzará un mensaje mencionando que ese id ya existe

4 PROCEDURE InsertFactura(IN p\_ID\_secretaria CHAR(10),IN p\_ID\_cliente VARCHAR(13),IN p\_fecha DATE,IN p\_hora TIME,IN p\_direccion\_local VARCHAR(100),IN p\_metodo\_pago ENUM('Efectivo', 'Transferencia'),IN p\_subtotal\_sin\_impuestos FLOAT,IN p\_subtotal\_0Porcent FLOAT,IN p\_ValorTotal FLOAT): Se encarga de insertar una nueva factura en la tabla factura, en caso de que el id de la secretaría o el id del cliente no existan se lanzará un mensaje mencionando que el id no existe, en caso de que el método de pago ingresado no sea efectivo o transferencia también se lanzará un mensaje haciendo mención de que el método de pago es incorrecto

5) PROCEDURE InsertDetalle(IN p\_ID\_factura INT,IN p\_ID\_producto VARCHAR(6),IN p\_unidades INT,IN p\_totalProdu INT,IN p\_detalle\_adic VARCHAR(100)): Se encarga de insertar un nuevo detalle en la tabla detalle, en caso de que el id de la factura y el id del producto existan en una misma tupla se lanzará un mensaje de que el detalle ya existe, en caso de que el id de la factura o el id del producto no existan se lanzará un mensaje haciendo mención de que el id no existe

6) PROCEDURE InsertReclamacion(IN p\_ID\_secretaria CHAR(10),IN p\_ID\_cliente VARCHAR(13),IN p\_descripcion VARCHAR(100)): Se encarga de insertar una nueva reclamación en la tabla Reclamacion, en caso de que el id de la secretaría o el id del cliente no existan se lanzará un mensaje haciendo mención de que el id no existe

7) PROCEDURE InsertProveedor(IN p\_cedula CHAR(10),IN p\_nombre VARCHAR(30),IN p\_telefono INT): Se encarga de insertar un nuevo proveedor en la tabla Proveedor, en caso de que la cédula del proveedor ya exista se lanzará un mensaje de que la cédula del proveedor ya existe

8) PROCEDURE InsertLoteMadera(IN p\_id\_proveedor CHAR(10),IN p\_id\_secretaria CHAR(10),IN p\_precio FLOAT,IN p\_fecha\_llegada DATE):Se encarga de insertar un nuevo lote de madera en la tabla lote\_madera, en caso de que la cédula del proveedor o el id de la secretaría no exista se lanzará un mensaje de que el id no existe

9) PROCEDURE InsertEvaluacion(IN p\_id\_proveedor CHAR(10),IN p\_calidad VARCHAR(10),IN p\_puntualidad VARCHAR(15),IN p\_detalle VARCHAR(100)):Se encarga de insertar insertar una nueva evaluación en la tabla Evaluacion, en caso de que la cédula del proveedor ya exista se lanzará un mensaje que diga que el id del proveedor no existe

10) PROCEDURE InsertTipoDeMadera(IN p\_id VARCHAR(6),IN p\_nombre VARCHAR(30),IN p\_precio\_unitario FLOAT,IN p\_condic\_ambiental VARCHAR(20)): Se encarga de insertar un nuevo tipo de madera en la tabla tipo\_de\_madera, en caso de que el id ya exista se lanzará un mensaje mencionando que el id insertado ya existe

11) PROCEDURE InsertEspecificacion(IN p\_id\_lote INT,IN p\_id\_madera VARCHAR(6),IN p\_importe INT,IN p\_cantidad INT): Se encarga de insertar una especificación nueva en la tabla Especificacion, en caso de que el id del lote y el id de la madera coincidan en una misma tupla se lanzará un mensaje haciendo mención de que la especificación ya existe, en caso de que el id del lote o el id de la madera no existan se lanzará un mensaje de que el id ingresado no existe

12) PROCEDURE InsertOperario(IN p\_ID CHAR(10),IN p\_nombre VARCHAR(40),IN p\_horaInicio TIME,IN p\_horaFin TIME,IN p\_fechaCapacitacion DATE,IN p\_tipoCapacitacion VARCHAR(20)): Se encarga de insertar un nuevo operario en la tabla operario y en la tabla empleado, en caso de que el id del operario insertado ya exista en la tabla operario o empleado se lanzará un mensaje que diga que aquel operario ya existe

13) PROCEDURE InsertAsistenteOperario(IN p\_ID CHAR(10),IN p\_nombre VARCHAR(40),IN p\_horaInicio TIME,IN p\_horaFin TIME,IN p\_fechaCapacitacion DATE,IN p\_tipoCapacitacion VARCHAR(20)): Se encarga de insertar un nuevo asistente de operario en la tabla asistente\_operario y en la tabla empleado, en caso de que el id del asistente insertado ya exista en la tabla asistente o empleado se lanzará un mensaje que diga que aquel operario ya existe

14) PROCEDURE InsertMaquinaria(IN p\_codigo INT,IN p\_nombre VARCHAR(50),IN p\_marca VARCHAR(20),IN p\_fecha\_adqui DATE):Se encarga de insertar una maquinaria en la Maquinaria, en caso de que el código insertado ya exista en la tabla se lanzará un mensaje haciendo mención de que dicha maquinaria ya existe

15) PROCEDURE InsertMantenimiento(IN p\_ID\_operario CHAR(10),IN p\_codigo\_maquinaria INT,IN p\_ID\_secretaria CHAR(10),IN p\_detalles VARCHAR(100),IN p\_fecha DATE): Se encarga de insertar un nuevo mantenimiento en la tabla Mantenimiento, en caso de que el id del operario, el código de la maquinaria o el id de la secretaría no exista se lanzará un mensaje haciendo mención de que dicha maquinaria ya existe

16) PROCEDURE InsertLimpieza(IN p\_lugar VARCHAR(30)): Se encarga de insertar una nueva limpieza en la tabla Limpieza

17) PROCEDURE InsertRegistro(IN p\_ID\_asistente CHAR(10),IN p\_ID\_limpieza INT,IN p\_ID\_secretaria CHAR(10),IN p\_fecha DATE): Se encarga de insertar un nuevo registro en la tabla Registro, en caso de que el id del asistente, el id de la limpieza y el id de la secretaría coincidan en una misma tupla se lanzará un mensaje haciendo mención de que el registro ya existe, en caso de que alguno de estos IDs no exista se lanzará un mensaje que el ID es inexistente

18)CREATE PROCEDURE InsertEmpleado(IN p\_ID CHAR(10),IN p\_nombre VARCHAR(100),IN p\_horaInicio TIME,IN p\_horaFin TIME,IN p\_fechaCapacitacion DATE,IN p\_tipoCapacitacion VARCHAR(50)): Se encarga de insertar un nuevo empleado en la tabla Empleado, en caso de que el id insertado ya exista en la tabla se lanzará un mensaje de que el empleado ya existe

19)CREATE PROCEDURE InsertRolDePagos(IN p\_ID\_empleado CHAR(10),IN p\_rol ENUM('A', 'S', 'O'),IN p\_dias\_laborados INT,IN p\_sueldo FLOAT,IN p\_horas\_extras FLOAT,IN p\_total\_ingresos FLOAT,IN p\_egreso\_IESS FLOAT,IN p\_anticipos FLOAT,IN p\_total\_egresos FLOAT,IN p\_liquido\_a\_recibir FLOAT): Se encarga de insertar un nuevo rol de pago en la table Rol\_de\_pagos, en caso de que el id del empleado insertado no exista en la tabla empleado se lanzará un mensaje de que el id es inexistente, en caso de que el rol ingresado no pertenezca a ‘O’,’A’ o ‘S’ lanzará un mensaje de que el rol ingresado no es válido

**TRIGGERS**

**trigger trgActualizarProducto:** Trigger encargado de actualizar el ID de la tabla Especificación asociada al ID antiguo del Producto que se modificó los datos

**trigger ActualizarTotal:** Trigger encargado de actualizar el subtotal sin impuestos y el valor total de la factura asociada al Detalle que se modificó los datos

**trigger trgActualizarCliente:** Trigger encargado de actualizar el ID de las tablas Reclamación y Factura asociadas al ID antiguo del Cliente que se modificó los datos

**trigger actualizarOperario:** Trigger encargado de actualizar el ID de las tablas Mantenimiento y Rol\_de\_pagos asociadas al ID antiguo del Operario que se modificó los datos

**trigger actualizarMaquinaria:** Trigger encargado de actualizar el ID de la tabla Mantenimiento asociada al ID antiguo del Operario que se modificó los datos

**trigger actualizarAsistente:** Trigger encargado de actualizar el ID de las tablas Registro y Rol\_de\_pagos asociadas al ID antiguo del Asistente\_operario que se modificó los datos

**trigger actualizarSecretaria:** Trigger encargado de actualizar el ID de las tablas Factura, Mantenimiento, Reclamacion, Registro, Rol\_de\_pagos asociadas al ID antiguo de la Secretaria que se modificó los datos

**trigger actualizarProveedor:** Trigger encargado de actualizar el ID de las tablas Lote\_madera y Evaluacion asociadas al ID antiguo del Proveedor que se modificó los datos

**trigger trgActualizarTipoMadera:** Trigger encargado de actualizar el ID de la tabla Especificacion asociadas al ID antiguo del Tipo\_de\_madera que se modificó los datos

**TRIGGERS INSERCIÓN**

**trigger trgNuevoTotal:** Trigger encargado de modificar el valor total, el subtotal sin impuestos y el subtotal 0% de la tabla Factura que tenga relación con el nuevo Detalle insertado, aumentando o disminuyendo con los datos que provengan de la tabla insertada

**trigger trgNuevoPrecio:** Trigger encargado de modificar el precio de la tabla Lote\_madera que tenga relación con la nueva Especificacion insertada, aumentando o disminuyendo con los datos que provengan de la tabla insertada

**Reportes**

Reporte Mensual de Compras:

**Propósito:** Esta vista genera un reporte de las compras realizadas durante el mes actual. Incluye detalles sobre el proveedor, la fecha de llegada de los lotes de madera, el tipo de madera, el importe (coste) de la compra, y la cantidad comprada.

**Uso:** Ideal para obtener una visión global de las compras realizadas en un mes específico, lo que permite análisis de gastos, control de inventario, y seguimiento de proveedores.

Reporte de ventas diarias

**Propósito:** Esta vista proporciona un resumen diario de las ventas, mostrando cuántas unidades de cada producto se vendieron y el total generado por esas ventas.

**Uso:** Útil para el seguimiento diario de las ventas, análisis de rendimiento de productos en un día específico, y control de inventario.

Reporte mensual de mantenimiento

**Propósito:** Esta vista ofrece un reporte mensual de las actividades de mantenimiento realizadas en la maquinaria, especificando quién realizó el mantenimiento, los detalles del mismo, la fecha, y la maquinaria involucrada.

**Uso:** Es fundamental para el seguimiento del mantenimiento preventivo y correctivo de la maquinaria, permitiendo asegurar que todas las tareas de mantenimiento se realicen a tiempo y registrar quién las realizó.

Reporte diario de limpieza de Área de Corte

**Propósito:** Esta vista muestra los empleados responsables de la limpieza diaria en el área de corte. Específicamente, indica quiénes han realizado tareas de limpieza en dicha área en la fecha actual.

**Uso:** Es útil para el control diario de limpieza, asegurando que el área de corte se mantenga en condiciones adecuadas, y para la asignación de responsabilidades en cuanto a tareas de limpieza.

Índices

**idx\_lote\_madera\_fecha\_llegada**

Este índice fue debido a la existencia de la vista ReporteCompras y la búsqueda de elementos por fecha en la tabla Lote\_madera. Los datos obtenidos observan una ligera mejoría en el tiempo de ejecución y las filas escaneadas, lo cual con mas datos, es importante.

Antes:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Type | Possible keys | Key | Rows Scanned | Duration (seconds) | Extra info | Rows returned |
| ALL | PRIMARY ,id\_proveedor |  | 10 | 0.0020 | Using where, using filesort | 1 |

Después:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Type | Possible keys | Key | Rows Scanned | Duration (seconds) | Extra info | Rows returned |
| range | PRIMARY ,id\_proveedor ,idx\_lote\_madera, fecha\_llegada | idx\_lote\_madera, fecha\_llegada | 1 | 0.0018 | Using index condition | 1 |

**idx\_factura\_fecha**

Este índice fue debido a la existencia de la vista ReporteVentasDiay la búsqueda de elementos por fecha en la tabla factura.

Antes:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Type | Possible keys | Key | Rows Scanned | Duration (seconds) | Extra info | Rows returned |
| ALL | ID | ID | 9 | 0.0020 | Using where |  |

Después:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Type | Possible keys | Key | Rows Scanned | Duration (seconds) | Extra info | Rows returned |
| range | ID, **idx\_factura\_fecha** | **idx\_factura\_fecha** | 1 | 0.0012 | Using index | 11 |

**idx\_mantenimiento\_fecha**

Este índice fue debido a la existencia de la vista ReporteMantenimiento y la búsqueda de elementos por fecha en la tabla de mantenimiento.

Antes:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Type | Possible keys | Key | Rows Scanned | Duration (seconds) | Extra info | Rows returned |
| range | ID\_operario,codigo\_maquinaria | PRIMARY | 9 | 0.0018 | using where | 1 |
| eq\_ref | PRIMARY | PRIMARY | 1 | 0.0018 | Using index | 1 |

Después:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Type | Possible keys | Key | Rows Scanned | Duration (seconds) | Extra info | Rows returned |
| range | ID\_operario,codigo\_maquinaria,idx\_mantenimiento\_fecha | idx\_mantenimiento\_fecha | 1 | 0.0010 | Using index condition, using where | 1 |
| eq\_ref | PRIMARY | PRIMARY | 1 | 0.0010 | Using index | 1 |

**idx\_registro\_fecha y idx\_limpieza\_lugar**

Estos indices fue debido a la existencia de la vista AreaDeCorteLimpieza y la búsqueda de elementos por fecha en la tabla registro y de elementos del lugar en la tabla Limpieza.

Antes:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Type | Possible keys | Key | Rows Scanned | Duration (seconds) | Extra info | Rows returned |
| ALL | PRIMARY, ID\_limpieza | PRIMARY | 8 | 0.0031 | Using where | 1 |
| eq\_ref | PRIMARY | PRIMARY | 8 | 0.0031 | Using where | 1 |

Después:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Type | Possible keys | Key | Rows Scanned | Duration (seconds) | Extra info | Rows returned |
| ref | PRIMARY,ID\_limpieza,idx\_registro\_fecha | Idx\_registro\_fecha | 1 | 0.0026 | Using index | 1 |
| Eq\_ref | PRIMARY,idx\_limpieza\_lugar | PRYMARY | 1 | 0.0026 | Using index | 1 |

**Link github:**

<https://github.com/Darvier88/AserrioDataPro.git>