DISEÑO DE UNA BASE DE DATOS

FINCA

**Valerie Michel Lasso Lizcano**

**Camilo Machuca Vega**

**T2**

**PEDRO FELIPE GÓMEZ BONILLA**

**CAMPUSLANDS  
PROYECTO FILTRO**

**TIBÚ  
2024**

Contenido

[**Introducción** 3](#_Toc183678442)

[**Caso de estudio** 4](#_Toc183678443)

[**Planificación** 5](#_Toc183678444)

[**Entidades** 5](#_Toc183678445)

[**Atributos** 5](#_Toc183678446)

[**Modelo conceptual** 7](#_Toc183678447)

[**Definición de entidades, atributos y relaciones:** 8](#_Toc183678448)

[*Entidades:* 8](#_Toc183678449)

[*Atributos:* 8](#_Toc183678450)

[*Relaciones*: 8](#_Toc183678451)

[**Propósito de las entidades:** 8](#_Toc183678452)

[**Modelo lógico** 10](#_Toc183678453)

[**Grafica:** 11](#_Toc183678454)

[Link del diagrama lógico y conceptual 11](#_Toc183678455)

[**Relaciones** 11](#_Toc183678456)

[**Cardinalidad** 15](#_Toc183678457)

[**Primera forma Normal(1FN)** 16](#_Toc183678458)

[**Segunda forma Normal (2FN)** 17](#_Toc183678459)

[**Tercera forma Normal (3FN)** 19](#_Toc183678460)

[**Construcción del modelo físico** 20](#_Toc183678461)

[1. Tabla proveedores 20](#_Toc183678462)

[2. Tabla finca 21](#_Toc183678463)

[3. Tabla compras 22](#_Toc183678464)

[4. Tabla areas de la finca 23](#_Toc183678465)

[5. Tabla agroquímicos 24](#_Toc183678466)

[6. Tabla herramientas 24](#_Toc183678467)

[7. Tabla lotes 25](#_Toc183678468)

[8. Tabla cuidado de los cultivos 25](#_Toc183678469)

[9. Tabla de cargos 26](#_Toc183678470)

[10. Tabla de los empleados 27](#_Toc183678471)

[11. Tabla de horarios de los empleados 27](#_Toc183678472)

[12. Tabla de vacaciones 28](#_Toc183678473)

[13. Tabla de bonificación y descuentos 28](#_Toc183678474)

[14. Tabla de pagos empleados 29](#_Toc183678475)

[15. Tabla de los clientes 29](#_Toc183678476)

[16. Tabla maquinarias 30](#_Toc183678477)

[17. Tabla método de pago 30](#_Toc183678478)

[18. Tabla procesos 31](#_Toc183678479)

[19. Tabla productos 32](#_Toc183678480)

[20. Tabla ventas 33](#_Toc183678481)

[21. Tabla monitoreos 34](#_Toc183678482)

[22. Tablas plantas enfermas 34](#_Toc183678483)

[23. Tabla cosechas 35](#_Toc183678484)

[24. Tabla daños 35](#_Toc183678485)

[**Diagrama UML** 36](#_Toc183678486)

[**Diagrama** 37](#_Toc183678487)

[**Usuarios** 37](#_Toc183678488)

[**Consultas** 40](#_Toc183678489)

[**Funciones** 59](#_Toc183678490)

[**Procedimientos** 73](#_Toc183678491)

[**Triggers** 85](#_Toc183678492)

[**Eventos** 98](#_Toc183678493)

# **Introducción**

Este documento contiene el proceso de diseño de una base de datos perteneciente a una finca de cacao, que busca implementar la eficiencia de gestión de la información más relevante, como los procesos de desarrollo y comercialización de cacao. Nuestro grupo implementó un caso de estudio, donde se analizó los requerimientos dados por el cliente. Gracias a este estudio se pudo profundizar el tema, investigando sobre los procesos que lleva a cabo una finca que produce y distribuye sus productos. Nos dimos cuenta que la base de datos no solo lleva un registro de los procesos y comercialización, es importante, claro, pero debemos tener un seguimiento de las herramientas que se usan para el proceso de desarrollo de la finca, el registro de empleados, clientes y un seguimiento al estado del cultivo.

Después de tener un conocimiento más amplio de lo que se necesita para llevar el registro de una finca pasamos a crear entidades y atributos coherentes que ayuden con el orden y el proceso de información de una manera entendible para los usuarios.

Luego de tener la estructura básica de la base de datos procedemos a plasmarla en un diagrama conceptual, siendo las entidades el personaje principal y los atributos los datos que caracterizan cada entidad. A partir del diagrama conceptual se busca las relaciones y cardinalidades de las tablas concluyendo así con el diagrama lógico. A continuación, damos inicio a la normalización, quien se encarga de organizar los datos de manera que se minimicen las redundancias y se maximice la integridad de los datos. Este proceso se realiza a partir de la 1FN, 2FN Y 3FN.

Al final se hará la conversión del modelo lógico al físico, en donde obtendremos su implementación real de todo el proceso antes realizado, añadiendo el tipo de dato y la cantidad de caracteres de cada elemento. También implementaremos un diagrama UML para visualizar la estructura de la base de datos y acto seguido unas consultas, funciones, procedimientos, tiggers y eventos, creados a partir del planteamiento de un problema de información que el usuario pueda necesitar.

# **Caso de estudio**

Mi equipo de trabajo implementó una investigación sobre las fincas de cacao con el objetivo de comprender los procesos y elementos importantes que intervienen en la producción y gestión de la finca. A través de nuestra investigación, identificamos muchos factores importantes que influyen en la finca como:

1. Los daños de las máquinas, herramientas e infraestructura de la finca.
2. Los daños del cultivo.
3. Bonificación para los empleados que cumplen con un buen desempeño.
4. Los datos importantes del empleado en el trabajo como el cargo, horario, las vacaciones y el pago.
5. El registro de compras realizadas a los proveedores.
6. El registro del cuidado a cada cultivo.
7. Las áreas donde se realiza el proceso del producto.
8. Los registros de envío.
9. Descuentos para clientes que compran al por mayor.
10. Los fertilizantes e insecticidas aplicados al cultivo.
11. Un registro de stock del producto a la venta.
12. Información de propiedad como los dueños y el estado (arrendado o propio).

# **Planificación**

En esta parte mi equipo planteó una idea de las entidades y atributos influyentes en la gestión de información de la base de datos teniendo en cuenta los puntos importantes que destacaron en la investigación del caso de estudio. Estos datos pueden cambiar a medida que avanza el proyecto.

### **Entidades**

1. Finca
2. Productos a vender
3. Proveedores
4. Área de la finca
5. Cargo
6. Empleados
7. Compras
8. Bonificaciones/descuentos
9. Pago de los empleados
10. Herramientas
11. Horario de empleados
12. Vacaciones de empleados
13. Fertilizantes/insecticidas
14. Cultivo
15. Cuidado del cultivo
16. Monitoreo
17. Cosecha
18. Procesos
19. Cultivo enfermo
20. Clientes
21. Ventas
22. Registro de envío
23. Precio de envío
24. Daños
25. Lotes

### **Atributos**

1. Finca: id, nombre, ancho, largo, unidad de medida, id dueño, país, ciudad, pueblo, teléfono, correo electrónico, dirección, tipo de propiedad, descripción (propósito de la finca).
2. Productos: id, id de la finca, nombre, peso, unidad de medida, stock, precio de venta, fecha de producción, fecha de vencimiento.
3. Proveedores: id, nombre1, nombre2, apellido1, apellido2, cedula, país, ciudad/pueblo, dirección, teléfono, correo electrónico, fecha de registro.
4. Área de la finca: id, nombre, descripción.
5. Cargo: id, id área finca, nombre, categoría, frecuencia de pago (mensual, quincenal), fecha de pago, salario.
6. Empleados: id, id del Cargo, nombre1, nombre2, apellido1, apellido2, cedula, celular, dirección, estado, fecha de nacimiento, fecha de ingreso, fecha de salida.
7. Compras: id, id del proveedor, nombre del producto comprado, cantidad comprada, costo, fecha de compra, estado (pendiente, pagado), debe (falta por pagar).
8. Bonificaciones/descuentos: id, porcentaje (10%), descripción (recolectó veinte kilos de cacao), estado (activo, inactivo).
9. Pagos de los empleados: id, id del empleado, id de bonificaciones, pago realizado, fecha de pago.
10. Herramientas: id, id área de la finca, id de compras, categoría, tipo, stock, estado (dañado, en reparación, bueno), id herramienta de aplicación.
11. Maquinaria: id, tipo, categoría, tipo de combustible, descripción, estado.
12. Horario de empelados: id, id del empleado, hora de entrada, hora de salida, horas de almuerzo.
13. Vacaciones: id, id del empelado, fecha de salida, fecha de llegada, motivo (enfermedad, receso, vacaciones).
14. Fertilizantes e insecticidas: id, id de la finca, id de compras, nombre de la marca, tipo, categoría, stock, descripción, empaque (recipientes, bolsas, envase), peso, unidad de medida, costo, fecha de vencimiento, id del proveedor, método de aplicación (foliar, terrestre).
15. Lotes: id, ancho, largo, unidad de medida, cantidad de plantas por lote, cantidad de surcos.
16. Cultivo: id, id de la finca, nombre, ancho, largo, unidad de medida, cantidad de árboles, nombre del árbol, fecha de siembra, producción anual, edad del cultivo.
17. Cuidado del cultivo: id, id del lote, id del fertilizante o insecticida, descripción de aplicación, fecha de cuidado, id herramienta de dosis, medida de dosis, unidad de medida, tipo de cuidado.
18. Monitoreo: id, id de cuidado de cultivo, id del empleado, fecha de monitoreo, descripción del resultado, observaciones.
19. Cosecha: id, id del empleado, id del cultivo, cantidad, unidad de medida, fecha de cosecha, id herramienta, problemas de cosecha.
20. Procesos: id, id área finca, id empleado, id de maquinaria, tipo de proceso, producto esperado, fecha de inicio, fecha final, problemas de proceso, descripción del problema.
21. Plantas enfermas: id, id capeo/monitoreo, tipo de enfermedad, nombre de enfermedad, estado (activo, inactivo), descripción, fecha de entrada, fecha de salida.
22. Clientes: id, nombre1, nombre2, apellido1, apellido2, cedula, dirección, teléfono, país, ciudad, pueblo, dirección, fecha de registro.
23. Ventas: id, id empleado, id cliente, id producto, cantidad del producto, total, dirección, id de descuento, precio a pagar, id de envío.
24. Registro envío: id, id transporte (herramienta/transporte), id empleado, país, ciudad, pueblo, id precio de envío, fecha de salida, fecha de llegada, problemas de envío, descripción del problema.
25. Daños: id, id de la finca, tipo de daño, descripción del daño, costo de reparación, fecha del daño, fecha reparación, fecha de reparación terminada.

# **Modelo conceptual**

El modelo conceptual es una representación gráfica del planteamiento de la estructura de una base de datos, que se enfoca en las entidades, atributos y relaciones. Estas tienen u

1. Entidades: Representadas por rectángulos
2. Atributos: Representados por óvalos.
3. Relaciones: Representados por líneas.

**Ejemplo:**

**Entidades Atributos Relaciones**

## **Definición de entidades, atributos y relaciones:**

### *Entidades:*

Representan los objetos que se almacenan en la base de datos, son las cosas e ideas que se quieren describir o almacenar en la base de datos. Una entidad podría ser la planta y esta entidad almacena los datos de la planta.

### *Atributos:*

Son las características de las entidades, donde se especifica los datos que se desea almacenar. Para la entidad planta los datos que se desean almacenar se definen por atributos, por ejemplo, una planta tiene de atributo el nombre, el lugar donde está ubicada, la especie, edad, estado, y la fecha en que fue sembrada.

### *Relaciones*:

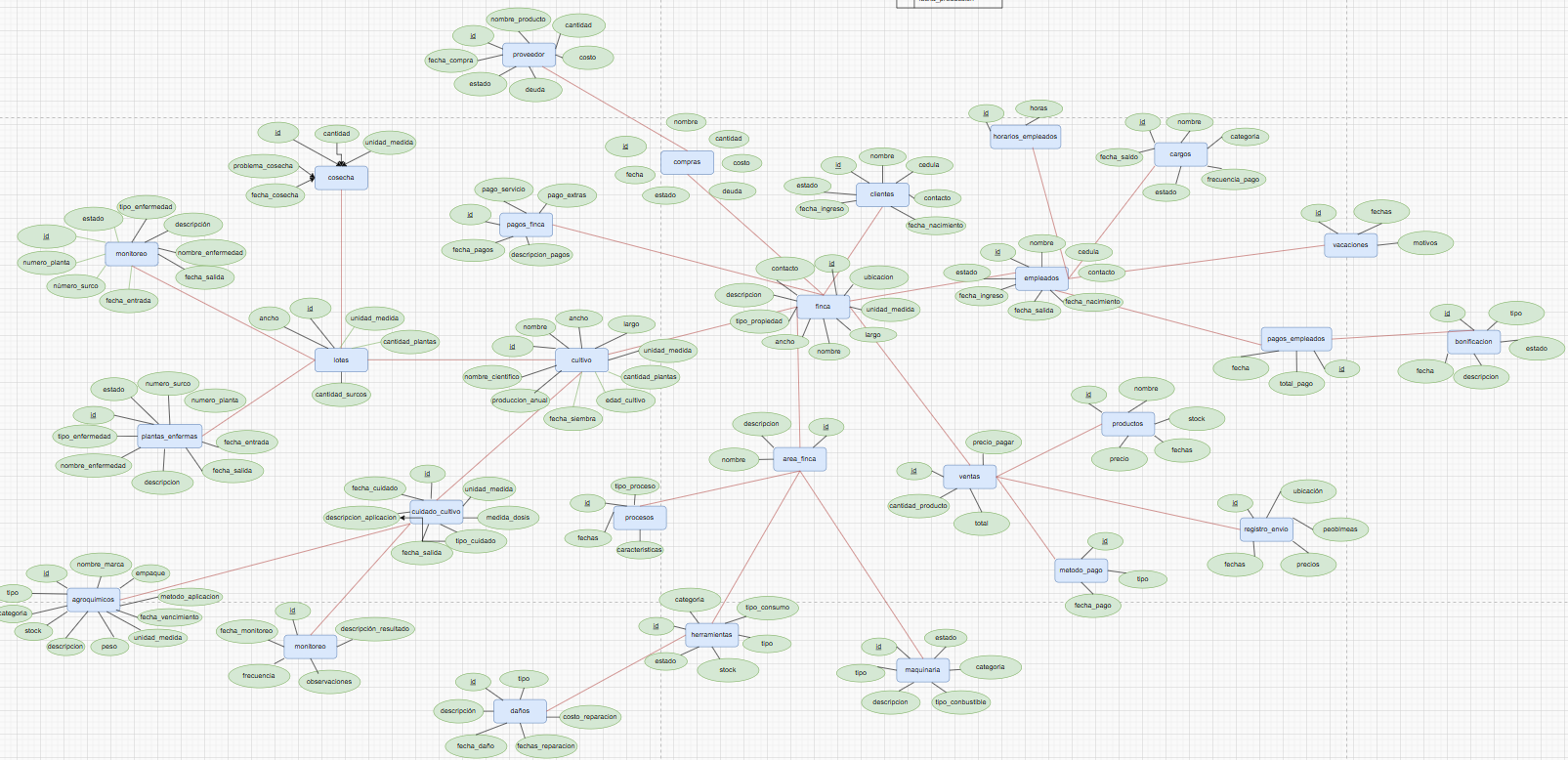
Son las conexiones entre las entidades. Una entidad planta puede estar relacionada a la entidad lote para saber su ubicación, o también puede estar relacionada a una entidad que describa su cuidado.

### **Propósito de las entidades:**

Las entidades son la base importante donde se construye la base de datos, por eso es importante conocer a profundidad el propósito y función de cada una de ellas para la aplicación y ejecución de la base de datos. A continuación, te presentaré las entidades que hemos incluido en nuestra base de datos de una finca de cacao, explicando el propósito, función e importancia.

1. Finca: la entidad finca fue creada para representar la entidad principal donde se desarrolla todo el proceso de producción en el sistema.
2. Productos a vender: la entidad productos fue creada para representar los artículos disponibles que se producen y están disponibles para vender.
3. Proveedores: la entidad proveedores fue creada para representar a las empresas o individuos que proveen los artículos necesarios para llevar a cabo los procesos de la finca, almacenando los datos del proveedor.
4. Área de la finca: la entidad área fue creada para representar a los sectores o zonas donde se desarrollan los procesos de producción.
5. Cargo: la entidad cargo fue creada para representar los puestos o roles que cumple cada empleado en la finca.
6. Empleados: la entidad empleados fue creada para representar a las personas que trabajan en la finca.
7. Compras: la entidad compras fue creada para llevar un registro de los productos comprados que se necesitaban en la finca.
8. Bonificaciones/descuentos: esta entidad fue creada para representar los bonos o descuentos que se dan a los empleados por su desempeño o a los clientes por la cantidad de compra.
9. Pago de los empleados: esta entidad fue creada para representar los pagos realizados a los empleados de la finca.
10. Herramientas: esta entidad fue creada para representar a los instrumentos utilizados en la finca.
11. Horario de empleados: esta entidad fue creada para representar los horarios de trabajo de cada empleado.
12. Vacaciones de empleados: esta entidad fue creada para representar las vacaciones tomadas por los empleados.
13. Fertilizantes/insecticidas: esta entidad fue creada para representar los agroquímicos aplicados a las plantas.
14. Cultivo: esta entidad fue creada para representar el cultivo de la finca.
15. Cuidado del cultivo: esta entidad fue creada para representar las diferentes actividades de cuidado y mantenimiento que se realiza en los cultivos de la finca.
16. Monitoreo: esta entidad fue creada para llevar un seguimiento y control de los cultivos.
17. Cosecha: esta entidad fue creada para llevar un seguimiento de las cosechas de la finca.
18. Procesos: esta entidad fue creada para representar las diferentes actividades y etapas que se realizan en la finca.
19. Cultivo enfermo: esta entidad fue creada para llevar un seguimiento de aquellas plantas que han sido afectadas por enfermedades o plagas.
20. Clientes: esta entidad fue creada para representar a los clientes de la finca.
21. Ventas: esta entidad fue creada para representar las transacciones de venta de productos.
22. Registro de envío: esta entidad fue creada para representar el registro de los envíos de las ventas realizadas.
23. Métodos de pago: fue creada para representar los métodos de pago disponibles en la finca.
24. Daños: esta entidad fue creada para llevar un registro de los instrumentos que sufrieron daños.
25. Lotes: esta entidad fue creada para llevar un registro de los lotes en los que se divide el cultivo.
26. Maquinaria: fue creada para representar los equipos utilizados en la finca.

**Grafica**



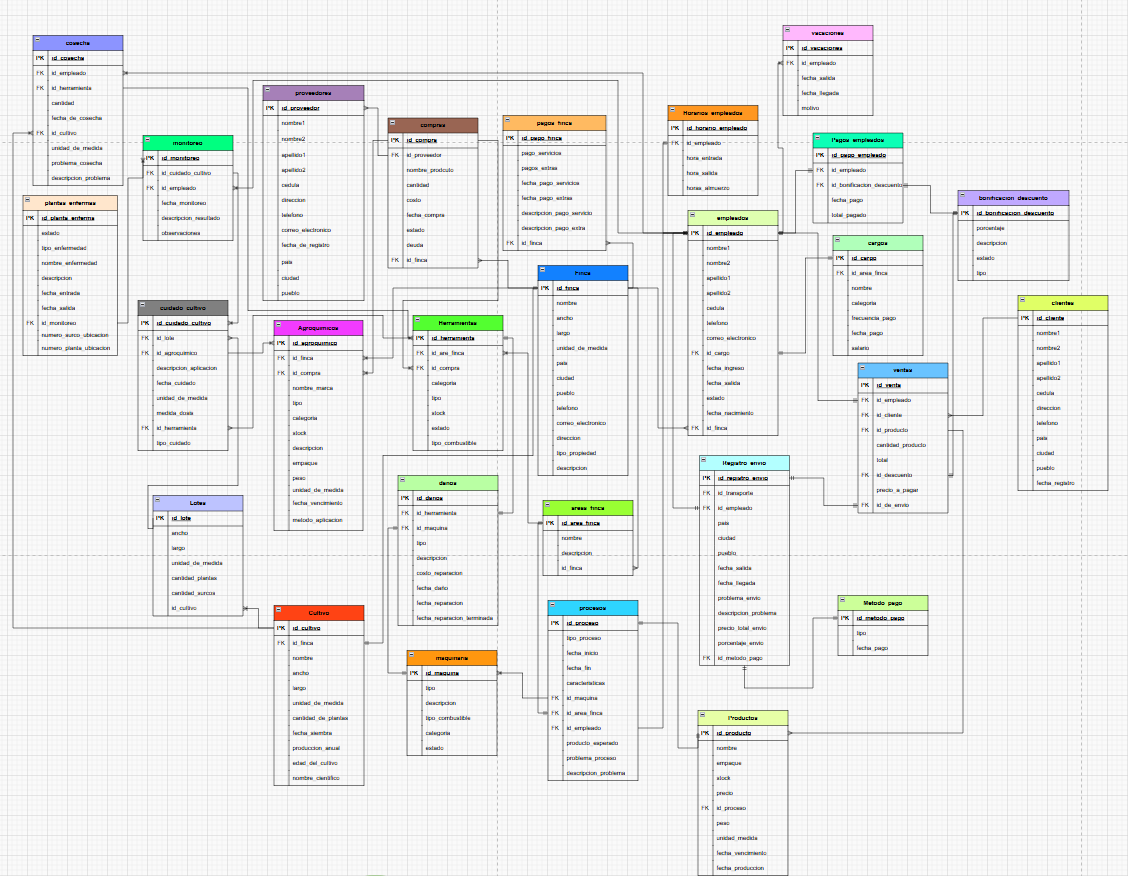
# 

# **Modelo lógico**

El modelo lógico es una representación gráfica con más detalles de la base de datos, que busca implementar un orden y la coherencia entre las tablas, describiendo las entidades, atributos y relaciones, considerando la implementación física. En este paso se puede visualizar las llaves primarias y foráneas mediante la relación de tablas.

1. Llave primaria (PK): identificador único del registro de datos de cada tabla.
2. Llave foránea (FK): es el identificador único traído de otra tabla para relacionar los datos entre ellas.

## **Grafica:**



### [Link del diagrama lógico y conceptual](https://drive.google.com/file/d/1ralxaUidNLrWJp7vJnQqi8x0oB_8G1mb/view?usp=sharing)

### **Relaciones**

1. Finca – Empleados: una finca puede contratar y tener trabajando muchos empleados y un empleado trabaja en una finca.



1. Finca - Areas de la finca: una finca esta dividida en muchas areas de trabajo.

Un area de trabajo esta ubicada en una finca.



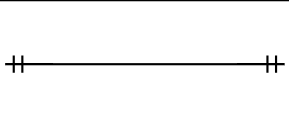
1. Finca – Compras: una finca puede realizar multiples compras. Una compra pertenece a una finca.



1. Finca – Agroquimicos: una finca puede utilizer muchos agroquímicos. Un agroquimico pertenece a una finca.



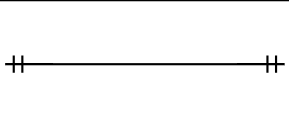
1. Finca – Cultivo: una finca tiene un cultivo. Un cultivo se encuentra en una finca.



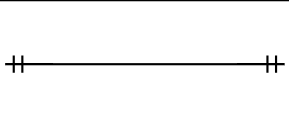
1. Provedores – Compras: una proveedor puede hacer multiples ventas a la finca. Y una le pertenece a una finca.



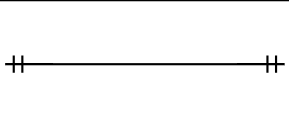
1. Empleados – Pagos de los empleados: un empleado puede recibir un pago. Y un pago le pertenece a un empleado.



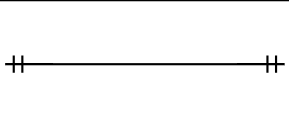
1. Pagos de los empleados – Bonificación y descuentos: un pago tiene una bonificación. Una bonificacio le pertenece a un pago.



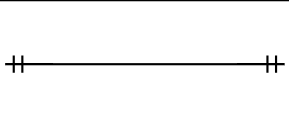
1. Empleados – cargos: un empleado tiene un cargo. Un cargo le pertenece a un empleado.



1. Empleados – Ventas: un empleado puede realizar una venta. Una venta es tendida por un empleado.



1. Empleados – Horarios de los empleados: un empleado tiene un horario. Un horario le pertenece a un empleado.



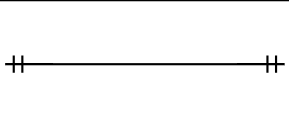
1. Empleados – Vacaciones: un empleado puede tener muchas vacaciones. Un registro en la table de vacaciones le pertenece a un empleado.



1. Empleados – Cosechas: un empleado puede tener varias cosechas. Una cosecha puede tener un empleado.



1. Empleados – Procesos: un empleado tiene un proceso. Un proceso tiene un empleado.



1. Empleados – Monitoreos: un empleado puede hacer varios monitoreos. Un monitoreo es hecho por un empleado.



1. Cuidado del cultivo - Monitoreos: un monitoreo puede registrar muchos cuidados del cultivo. Un cuidado del cultivo tiene un monitoreo.



1. Monitoreos – plantas enfermas: un monitoreo puede registrar muchas plantas enfermas. Una planta enferma tiene un monitoreo.



1. Cosecha – Herramientas: una cosecha puede hacer uso de muchas herramientas. Una herramienta tiene un registro en las cosechas.



1. Cosecha – Cultivo: un cultivo puede tener muchas cosechas. La cosecha se hace en un cultivo.



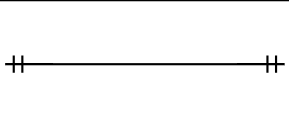
1. Cuidado del cultivo – Lote: un lote puede tener muchos registros del cuidado de las plantas. Un cuidado tiene un cultivo.



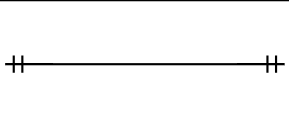
1. Lotes – cultivo: un cultivo tiene muchos lotes. Un lote esta ubicado en un cultivo



1. Daños – herramientas: un daño tiene el registro de un herramienta. Una herramienta tiene un daño.



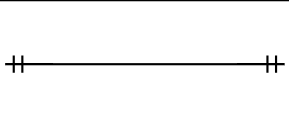
1. Daños – Maquinaria: un daño tiene el registro de una maquinaria. Una maquinaria tiene un daño.



1. Procesos – Maquinaria: un proceso puede hacer uso de muchas maquinarias. Una maquina ayuda a realizar un proceso.



1. Procesos – Areas de la finca: un proceso tiene un area especifica. Un area tiene un proceso.



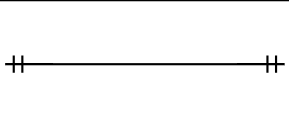
1. Agroquimicos – Compra: una compra puede incluir muchos agroquimicos. Un agroquimico tiene una compra.



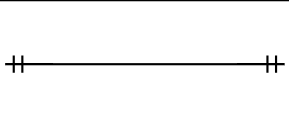
1. Empleados – Registro de envio: un empleado puede hacer muchos envios. Un envoi es hecho por un empleado.



1. Registro – Metodos de pago: un registro tiene un metodo de pago. Un metodo de pago tiene un registro de envoi.



1. Ventas – registro de envio: una venta tiene un registro de envoi. Un registro de envio tiene una venta.



1. Ventas – clientes: un cliente puede hacer muchas compras. Una compra tiene un cliente.



1. Ventas – Productos: una venta tiene muchos productos. Un product tiene una venta.



1. Ventas – descuento: un descuento tiene muchas ventas. Una venta tiene un descuento.



### **Cardinalidad**

1. Finca – Empleados: 1 – n
2. Finca - Areas de la finca: 1 – n
3. Finca – Compras: 1 – n
4. Finca – Agroquimicos: 1 – n
5. Finca – Cultivo: 1 – 1
6. Provedores – Compras: 1 – n
7. Empleados – Pagos de los empleados: 1 – 1
8. Pagos de los empleados – Bonificación y descuentos: 1 – 1
9. Empleados – cargos: 1 – 1
10. Empleados – Ventas: 1 – 1
11. Empleados – Horarios de los empleados: 1 – 1
12. Empleados – Vacaciones: 1 – n
13. Empleados – Cosechas: 1 – n
14. Empleados – Procesos: 1 – 1
15. Empleados – Monitoreos: 1- n
16. Ventas – descuentos: 1 – n
17. Ventas – Productos: 1 – n
18. Ventas – Clientes: 1 – n
19. Ventas – registro de envio: 1 – 1
20. Registro – Metodos de pago: 1 – 1
21. Empleados – Registro de envio: 1 – 1
22. Agroquimicos – Compra: 1 – 1
23. Procesos – Areas de la finca: 1 – 1
24. Procesos – Maquinaria: 1 – n
25. Daños – Maquinaria: 1 – 1
26. Daños – herramientas: 1 – 1
27. Lotes – cultivo: 1 – n
28. Cuidado del cultivo – Lote: 1 – n
29. Cosecha – Cultivo: 1 – n
30. Cosecha – Herramientas: 1 – n
31. Monitoreos – plantas enfermas: 1 – n
32. Cuidado del cultivo - Monitoreos: 1 – n

**Normalización**

La normalización es una parte importante en el diseño de una base de datos, con el fin de asegurar que los datos estén organizados de manera eficiente y sin redundancias. El proceso de normalización se realiza utilizando las siguientes formas normales:

## **Primera forma Normal(1FN)**

1. Cada casilla debe tener un dato.
2. Cada atributo debe contener valores atómicos.
3. Cada fila de la misma tabla debe contener datos diferentes.

**Descripción**: La tabla se encuentra en la primera forma normal después de cumplir con los requerimientos antes mencionados. Esta formal nos ayuda con el orden y la coherencia, proporcionando identificadores únicos, valores únicos y no repetidos.

1. Finca: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
2. Productos a vender: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
3. Proveedores: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
4. Área de la finca: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
5. Cargo: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
6. Empleados: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
7. Compras: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
8. Bonificaciones/descuentos: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
9. Pago de los empleados: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
10. Herramientas: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
11. Horario de empleados: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
12. Vacaciones de empleados: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
13. Agroquímicos: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
14. Cultivo: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
15. Cuidado del cultivo: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
16. Monitoreo: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
17. Cosecha: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
18. Procesos: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
19. Cultivo enfermo: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
20. Clientes: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
21. Ventas: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
22. Registro de envío: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
23. Precio de envío: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
24. Daños: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.
25. Lotes: se encuentra en 1FN, ya que cuenta con un identificador único y cada columna cuenta con valores únicos y no son repetitivos.

## **Segunda forma Normal (2FN)**

1. Debe estar en F1
2. La relación debe tener una clave principal de preferencia simple
3. Cada atributo de la tabla debe depender del atributo clave.

**Descripción**: Para estar en la segunda forma normal debemos haber cumplido con los requerimientos antes mencionados. En este paso verificamos que los datos dependan de la llave primaria.

1. Finca: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
2. Productos a vender: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
3. Proveedores: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
4. Área de la finca: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
5. Cargo: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
6. Empleados: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
7. Compras: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
8. Agroquímicos: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
9. Pago de los empleados: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
10. Herramientas: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
11. Horario de empleados: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
12. Vacaciones de empleados: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
13. Fertilizantes/insecticidas: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
14. Cultivo: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
15. Cuidado del cultivo: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
16. Monitoreo: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
17. Cosecha: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
18. Procesos: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
19. Cultivo enfermo: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
20. Clientes: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
21. Ventas: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
22. Registro de envío: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
23. Precio de envío: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
24. Daños: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.
25. Lotes: se encuentra en 2FN, ya que cuenta con un identificador único (llave primaria) y cada columna depende de él.

## **Tercera forma Normal (3FN)**

1. Debe estar en F2
2. se eliminan los datos repetidos

**Descripción**: En esta tercera forma normal tendremos en cuenta todos los requerimientos antes mencionados y eliminaremos datos repetidos que puedan dañar o generar una sobrecarga de datos en la base de datos, para esto podremos crear una nueva columna.

1. Finca: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
2. Productos a vender: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
3. Proveedores: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
4. Área de la finca: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
5. Cargo: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
6. Empleados: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
7. Compras: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
8. Bonificaciones/descuentos: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
9. Pago de los empleados: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
10. Herramientas: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
11. Horario de empleados: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
12. Vacaciones de empleados: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
13. Agroquímico: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
14. Cultivo: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
15. Cuidado del cultivo: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
16. Monitoreo: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
17. Cosecha: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
18. Procesos: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
19. Cultivo enfermo: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
20. Clientes: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
21. Ventas: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
22. Registro de envío: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
23. Precio de envío: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
24. Daños: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.
25. Lotes: se encuentra en 3FN, ya que pasó por la segunda forma y no cuenta con datos repetidos.

# **Construcción del modelo físico**

Este modelo se implementa después de haber creado el lógico. En esta parte de la estructura de datos pasaremos todo lo antes realizado a un lenguaje de sistema de gestión de datos MYSQL, donde podremos crear tablas a partir de las entidades con sus atributos e implementar la inserción de datos.

Descripción: En el sistema de gestión de datos creamos tablas, estas tablas representan las entidades, y las entidades contienen sus respectivos atributos organizados en columnas, a quienes se le ingresa el tipo de dato con la cantidad de caracteres si es necesario.

create database fincaCacao; -- crear la base de datos

use fincaCacao; -- usar la base de datos

### Tabla proveedores

create table proveedores(

id\_proveedor int auto\_increment primary key,

nombre1 varchar(50),

nombre2 varchar(50),

apellido1 varchar(50),

apellido2 varchar(50),

cedula varchar(50),

direccion varchar(50),

telefono varchar(50),

correo\_electronico varchar(50),

fecha\_de\_registro date,

pais varchar(50),

ciudad varchar(50),

pueblo varchar(50)

);

### Tabla finca

create table finca(

id\_finca int auto\_increment primary key,

nombre varchar(50),

ancho decimal(10,2),

largo decimal(10,2),

unidad\_de\_medida varchar(50),

pais varchar(50),

ciudad varchar(50),

pueblo varchar(50),

telefono varchar(50),

correo\_electronico varchar(50),

direccion varchar(50),

tipo\_propiedad varchar(50),

descripcion varchar(200)

);

### Tabla compras

create table compras(

id\_compra int auto\_increment primary key,

nombre\_producto varchar(50),

cantidad int,

costo int,

fecha\_compra date,

estado varchar(50),

deuda int,

id\_finca int,

foreign key(id\_finca)references finca(id\_finca),

id\_proveedor int,

foreign key(id\_proveedor) references proveedores(id\_proveedor)

);

create table cultivo(

id\_cultivo int auto\_increment primary key,

id\_finca int,

foreign key(id\_finca)references finca(id\_finca),

nombre varchar(50),

ancho decimal(10,2),

largo decimal(10,2),

unidad\_de\_medida varchar(50),

cantidad\_de\_plantas decimal(10,2),

fecha\_siembra date,

produccion\_anual decimal(10,2),

edad\_del\_cultivo int,

nombre\_cientifico varchar(50)

);

### Tabla areas de la finca

create table areas\_finca(

id\_area\_finca int auto\_increment primary key,

nombre varchar(50),

descripcion varchar(200),

id\_finca int,

foreign key(id\_finca) references finca(id\_finca)

);

### Tabla agroquímicos

create table agroquimicos(

id\_agroquimico int auto\_increment primary key,

nombre\_marca varchar(50),

tipo varchar(50),

categoria varchar(50),

stock int,

descripcion varchar(200),

empaque varchar(50),

peso decimal(10,2),

unidad\_de\_medida varchar(50),

costo decimal(10,2),

fecha\_verncimiento date,

metodo\_aplicacion varchar(50),

id\_compra int,

foreign key(id\_compra) references compras(id\_compra)

);

### Tabla herramientas

create table herramientas(

id\_herramienta int auto\_increment primary key,

categoria varchar(50),

tipo varchar(50),

stock int,

estado varchar(50),

tipo\_combustible varchar(50),

id\_compra int,

foreign key (id\_compra) references compras(id\_compra),

id\_area\_finca int,

foreign key(id\_area\_finca)references areas\_finca(id\_area\_finca)

);

### Tabla lotes

create table lotes(

id\_lote int auto\_increment primary key,

ancho decimal(10,2),

largo decimal(10,2),

unidad\_de\_medida varchar(50),

cantidad\_de\_plantas int,

id\_cultivo int,

foreign key(id\_cultivo) references cultivo(id\_cultivo)

);

### Tabla cuidado de los cultivos

create table cuidado\_cultivo(

id\_cuidado\_cultivo int auto\_increment primary key,

descripcion\_aplicacion varchar(150),

fecha\_cuidado date,

unidad\_de\_medida varchar(50),

medida\_dosis decimal(10,2),

tipo\_cuidado varchar(50),

id\_lote int,

foreign key(id\_lote)references lotes(id\_lote),

id\_agroquimico int,

foreign key(id\_agroquimico)references agroquimicos(id\_agroquimico),

id\_herramienta int,

foreign key(id\_herramienta) references herramientas(id\_herramienta)

);

### Tabla de cargos

create table cargos(

id\_cargo int auto\_increment primary key,

nombre varchar(50),

categoria varchar(50),

frecuencia\_pago varchar(50),

fecha\_pago date,

salario decimal(10,2),

id\_area\_finca int,

foreign key(id\_area\_finca) references areas\_finca(id\_area\_finca)

);

### Tabla de los empleados

create table empleados(

id\_empleado int auto\_increment primary key,

nombre1 varchar(50),

nombre2 varchar(50),

apellido1 varchar(50),

apellido2 varchar(50),

cedula varchar(50),

telefono varchar(50),

correo\_electronico varchar(50),

fecha\_ingreso date,

fecha\_salida date,

estado varchar(50),

fecha\_nacimiento date,

id\_finca int,

foreign key(id\_finca) references finca(id\_finca),

id\_cargo int,

foreign key(id\_cargo) references cargos(id\_cargo)

);

### Tabla de horarios de los empleados

create table Horarios\_empleados(

id\_horario\_empleado int auto\_increment primary key,

hora\_entrada datetime,

hora\_salida datetime,

horas\_almuerzo datetime,

id\_empleado int,

foreign key(id\_empleado) references empleados(id\_empleado)

);

### Tabla de vacaciones

create table vacaciones(

id\_vacaciones int auto\_increment primary key,

fecha\_salida date,

fecha\_llegada date,

motivo varchar(50),

id\_empleado int,

foreign key(id\_empleado)references empleados(id\_empleado)

);

### Tabla de bonificación y descuentos

create table bonificacion\_descuento(

id\_bonificacion\_descuento int auto\_increment primary key,

porcentaje decimal(10,2),

descripcion varchar(150),

estado varchar(50),

tipo varchar(50)

);

### Tabla de pagos empleados

create table pagos\_empleados(

id\_pago\_empleado int auto\_increment primary key,

fecha\_pago date,

total\_pagado decimal(10,2),

id\_empleado int,

foreign key(id\_empleado) references empleados(id\_empleado),

id\_bonificacion\_descuento int,

foreign key(id\_bonificacion\_descuento)references bonificacion\_descuento(id\_bonificacion\_descuento)

);

### Tabla de los clientes

create table clientes(

id\_cliente int auto\_increment primary key,

nombre1 varchar(50),

nombre2 varchar(50),

apellido1 varchar(50),

apellido2 varchar(50),

cedula varchar(50),

direccion varchar(50),

telefono varchar(50),

pais varchar(50),

ciudad varchar(50),

pueblo varchar(50),

fecha\_registro date

);

### Tabla maquinarias

create table maquinaria(

id\_maquinaria int auto\_increment primary key,

tipo varchar(50),

descripcion varchar(50),

tipo\_combustible varchar(50),

categoria varchar(50),

estado varchar(50)

);

### Tabla método de pago

create table metodo\_pago(

id\_metodo\_pago int auto\_increment primary key,

tipo varchar(50),

fecha\_pago date

);

create table registro\_envio(

id\_registro\_envio int auto\_increment primary key,

id\_transporte int,

foreign key(id\_transporte)references maquinaria(id\_maquinaria),

id\_empleado int,

foreign key(id\_empleado) references empleados(id\_empleado),

pais varchar(50),

ciudad varchar(50),

pueblo varchar(50),

fecha\_salida date,

fecha\_llegada date,

problema\_envio tinyint,

descripcion\_problema varchar(50),

precio\_total\_envio decimal(10,2),

porcentaje\_envio decimal(10,2),

id\_metodo\_pago int,

foreign key(id\_metodo\_pago)references metodo\_pago(id\_metodo\_pago)

);

### Tabla procesos

create table procesos(

id\_proceso int auto\_increment primary key,

fecha\_inicio date,

fecha\_fin date,

caracteristicas varchar(200),

producto\_esperado varchar(150),

problema\_proceso tinyint,

descripcion\_problema varchar(200),

id\_maquinaria int,

foreign key(id\_maquinaria)references maquinaria(id\_maquinaria),

id\_area\_finca int,

foreign key(id\_area\_finca)references areas\_finca(id\_area\_finca),

id\_empleado int,

foreign key(id\_empleado)references empleados(id\_empleado)

);

### Tabla productos

create table productos(

id\_producto int auto\_increment primary key,

nombre varchar(50),

empaque varchar(50),

stock int,

precio decimal(10,2),

peso decimal(10,2),

unidad\_de\_medida varchar(50),

fecha\_vencimiento date,

fecha\_produccion date,

id\_proceso int,

foreign key(id\_proceso)references procesos(id\_proceso)

);

### Tabla ventas

create table ventas(

id\_venta int auto\_increment primary key,

cantidad\_producto int,

total decimal(10,2),

precio\_a\_pagar decimal(10,2),

id\_empleado int,

foreign key(id\_empleado)references empleados(id\_empleado),

id\_cliente int,

foreign key(id\_cliente)references clientes(id\_cliente),

id\_producto int,

foreign key(id\_producto)references productos(id\_producto),

id\_descuento int,

foreign key(id\_descuento)references bonificacion\_descuento(id\_bonificacion\_descuento),

id\_envio int,

foreign key(id\_envio)references registro\_envio(id\_registro\_envio)

);

### Tabla monitoreos

create table monitoreos(

id\_monitoreo int auto\_increment primary key,

fecha\_monitoreo date,

descripcion\_resultado varchar(50),

observaciones varchar(150),

frecuencia varchar(50),

id\_cuidado\_cultivo int,

foreign key(id\_cuidado\_cultivo) references cuidado\_cultivo(id\_cuidado\_cultivo),

id\_empleado int,

foreign key(id\_empleado) references empleados(id\_empleado)

);

### Tablas plantas enfermas

create table plantas\_enfermas(

id\_planta\_enferma int auto\_increment primary key,

estado varchar(50),

tipo\_enfermedad varchar(50),

nombre\_enfermedad varchar(90),

descripcion varchar(150),

fecha\_entrada varchar(50),

fecha\_salida varchar(50),

numero\_surco\_ubicacion int,

numero\_planta\_ubicacion int,

id\_monitoreo int,

foreign key(id\_monitoreo)references monitoreos(id\_monitoreo)

);

### Tabla cosechas

create table cosecha(

id\_cosecha int auto\_increment primary key,

cantidad int,

unidad\_de\_medida varchar(50),

fecha\_cosecha date,

problema\_cosecha tinyint,

descripcion\_problema varchar(150),

id\_empleado int,

foreign key(id\_empleado)references empleados(id\_empleado),

id\_herramienta int,

foreign key(id\_herramienta)references herramientas(id\_herramienta),

id\_cultivo int,

foreign key(id\_cultivo)references cultivo(id\_cultivo)

);

### Tabla daños

create table daños(

id\_daño int auto\_increment primary key,

tipo varchar(50),

descripcion varchar(150),

costo\_reparacion decimal(10,2),

fecha\_daño date,

fecha\_reparacion date,

fecha\_reparacion\_terminada date,

id\_herramienta int,

foreign key(id\_herramienta) references herramientas(id\_herramienta),

id\_maquina int,

foreign key(id\_maquina) references maquinaria(id\_maquinaria)

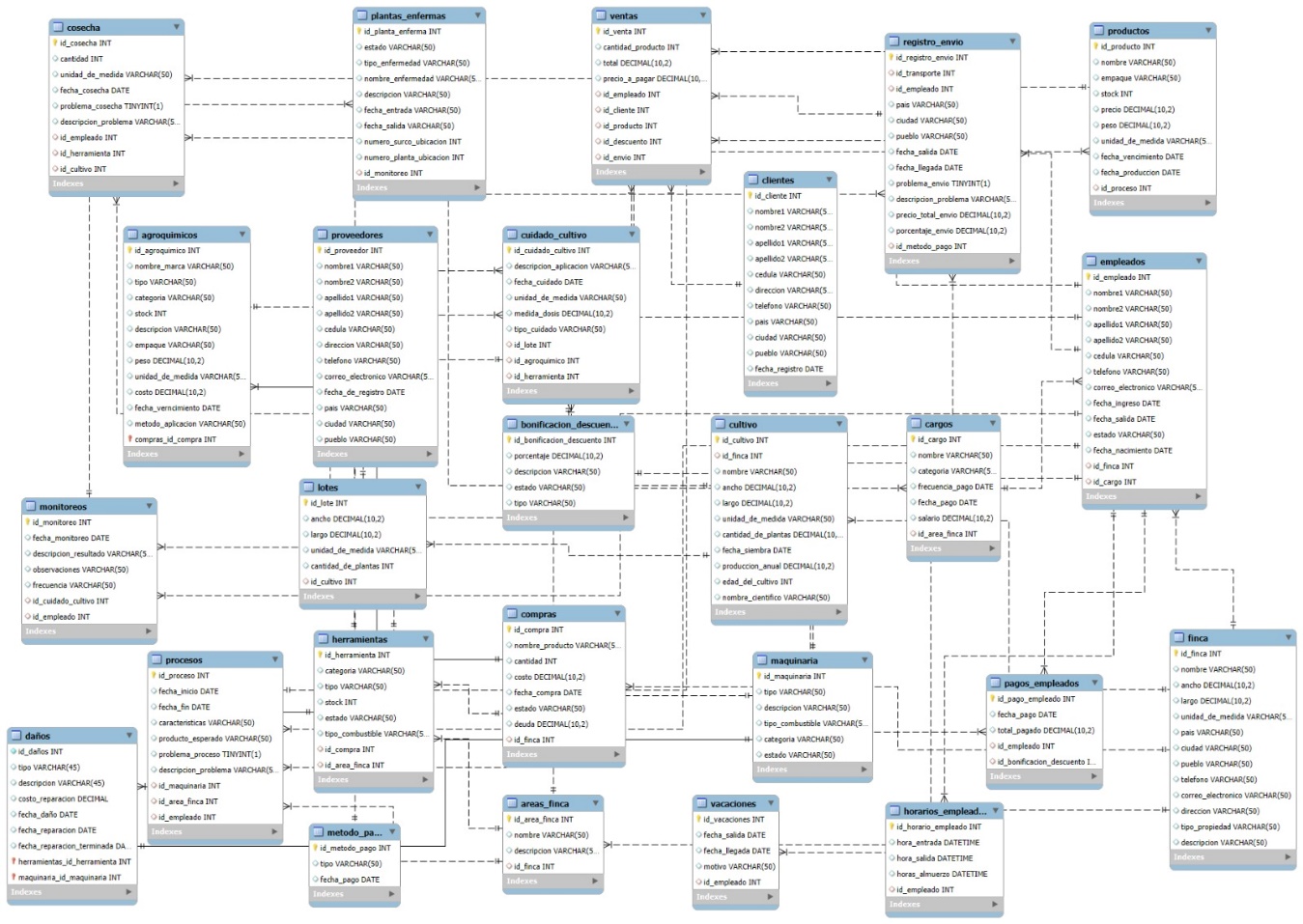
);

# **Diagrama UML**

En la construcción del diagrama UML toma como referencia la normalización para entender mejor la estructura de la base de datos. Este diagrama nos ayudará a visualizar y entender de una mejor manera su construcción e implementación de los requerimientos dados por la finca de cacao

**Descripción**: El diagrama UML ha sido diseñado con el objetivo de representar detalladamente la estructura de cada tabla y sus relaciones. En este diagrama se puede visualizar el nombre de la tabla, sus atributos con su tipo de dato y el limitante de caracteres. Los tipos de datos más usados en este proceso son: primary key(llave primaria), foreing key(llave foránea), varchar(cadena) y int(números enteros).

## **Diagrama**



# **Usuarios**

-- usuarios

**1. usuario administrador general**

create user 'Administrador\_Finca'@'localhost' identified by '2532';

grant all privileges on fincacacao.\* to 'administrador\_finca'@'localhost';

**2. usuario gestor de operaciones**

create user 'Gestor\_Operaciones'@'localhost' identified by '1234';

grant select, insert, update, delete on fincacacao.cultivo to 'Gestor\_Operaciones'@'localhost';

grant select, insert, update, delete on fincacacao.lotes to 'Gestor\_Operaciones'@'localhost';

grant select, insert, update, delete on fincacacao.cosecha to 'Gestor\_Operaciones'@'localhost';

grant select , insert, update, delete on fincacacao.procesos to 'Gestor\_Operaciones'@'localhost';

grant select on fincacacao.areas\_finca to 'Gestor\_Operaciones'@'localhost';

grant select on fincacacao.maquinaria to 'Gestor\_Operaciones'@'localhost';

grant select on fincacacao.herramientas to 'Gestor\_Operaciones'@'localhost';

grant select on fincacacao.empleados to 'Gestor\_Operaciones'@'localhost';

grant select on fincacacao.agroquimicos to 'Gestor\_Operaciones'@'localhost';

**3. usuario contador**

create user 'Contador'@'localhost' identified by 'contador112';

grant select on fincacacao.ventas to 'Contador'@'local@host';

grant select on fincacacao.registro\_envio to 'Contador'@'local@host';

grant select on fincacacao.empleados to 'Contador'@'local@host';

grant select on fincacacao.clientes to 'Contador'@'local@host';

grant select on fincacacao.productos to 'Contador'@'local@host';

grant select on fincacacao.daños to 'Contador'@'local@host';

**4. usuario gestor de inventario**

create user 'Gestor\_inventario'@'localhost' identified by 'inventa345';

grant select , update on fincacacao.agroquimicos to 'Gestor\_inventario'@'localhost';

grant select, update on fincacacao.herramientas to 'Gestor\_inventario'@'localhost';

grant select, update on fincacacao.productos to 'Gestor\_inventario'@'localhost';

grant select, update on fincacacao.maquinaria to 'Gestor\_inventario'@'localhost';

**5 usuario ágronomo**

create user 'Agrónomo'@'localost' identified by '12000';

grant select , update, delete, insert on fincacacao.plantas\_enfermas to 'Agrónomo'@'localost';

grant select on fincacacao.procesos to 'Agrónomo'@'localost';

grant select on fincacacao.agroquimicos to 'Agrónomo'@'localost';

grant select on fincacacao.herramientas to 'Agrónomo'@'localost';

grant select on fincacacao.areas\_finca to 'Agrónomo'@'localost';

grant select on fincacacao.empleados to 'Agrónomo'@'localost';

grant select , update, delete, insert on fincacacao.monitoreos to 'Agrónomo'@'localost';

grant select on fincacacao.cultivo to 'Agrónomo'@'localost';

grant select on fincacacao.cuidado\_cultivo to 'Agrónomo'@'localost';

grant select on fincacacao.lotes to 'Agrónomo'@'localost';

# **Consultas**

**consultas**

use fincaCacao;

**1. Cantidad de cacao producido en un año especifico.**

select sum(cantidad) from cosecha where fecha\_cosecha between'2024-01-01' and '2024-12-31';

**2. Promedio de producción de cacao del cultivo.**

select avg(cantidad) from cosecha

inner join cultivo on cosecha.id\_cultivo=cultivo.id\_cultivo;

**3. Productos disponibles en el inventario.**

select nombre from productos where not stock is null or stock=0;

**4. Listar los empleados que se encuentran trabajando en la finca.**

select \* from empleados;

select nombre1, apellido1 from empleados where estado='activo';

**5. Listar los empleados que están de vacaciones.**

select nombre1, nombre2 from empleados

inner join vacaciones on empleados.id\_empleado= vacaciones.id\_empleado where fecha\_llegada= now();

**6. Total de horas trabajadas por los empleados sin contar el almuerzo.**

select nombre1,apellido1, sum(timestampdiff(hour,hora\_entrada,hora\_salida)) from horarios\_empleados

inner join empleados on horarios\_empleados.id\_empleado=empleados.id\_empleado

group by 1,2;

**7. Promedio de salario de los empleados en la finca.**

select nombre1, apellido1, avg(total\_pagado)promedio\_salario from pagos\_empleados

inner join empleados on pagos\_empleados.id\_empleado=empleados.id\_empleado

group by 1,2;

**8. Listar los empleados que trabajan y trabajaron en la finca.**

select \* from empleados;

**9. Listar las herramientas que tengan algún daño.**

select categoria, tipo from herramientas where estado='dañado';

**10. Listar las herramientas que se encuentren en buen estado.**

select categoria, tipo from herramientas where estado='bien';

**11. Listar las maquinas que se encuentran en mal estado.**

select categoria, tipo from maquinaria where estado ='en mantenimiento';

**12. Listar las maquinas que están disponibles.**

select categoria, tipo from maquinaria where estado ='operativa';

**13. Listar los procesos por los que pasa el cacao para crear un producto.**

select distinct caracteristicas,productos.nombre from procesos

inner join productos where producto\_esperado=productos.nombre;

**14. Listar los tipos de agroquímicos de la finca.**

select distinct tipo from agroquimicos;

**15. Listar los tipos de agroquímicos disponibles.**

select distinct tipo from agroquimicos where not stock=0 or null;

**16. Promedio de cosecha.**

select avg(cantidad) as promedioCosecha from cosecha;

**17. Listar todas las herramientas utilizadas en las cosechas.**

select distinct tipo from cosecha

inner join herramientas on cosecha.id\_herramienta= herramientas.id\_herramienta;

**18. Listar las cosechas que tuvieron algún problema y el motivo.**

select descripcion\_problema from cosecha where problema\_cosecha= 1;

**19. Calcular cuantas cosechas tuvieron problemas.**

select count(\*) from cosecha where problema\_cosecha=1;

**20. Cuál es el promedio de la cantidad de fertilizante o insecticida aplicada al cultivo.**

select a.id\_agroquimico, a.nombre\_marca, a.tipo, medida\_dosis from agroquimicos a inner join cuidado\_cultivo c on c.id\_agroquimico=a.id\_agroquimico where tipo='fertilizante' or tipo='insecticida';

**21. Cuantas plantas están enfermas.**

select \* from plantas\_enfermas;

**22. Listar las plantas que se encuentran enfermas, con el tipo de enfermedad, el nombre de la enfermedad, y la ubicación de la planta.**

select id\_planta\_enferma, tipo\_enfermedad, nombre\_enfermedad, numero\_surco\_ubicacion, numero\_planta\_ubicacion, id\_monitoreo from plantas\_enfermas;

**23. Cuantas plantas murieron**

select count(\*) as plantas\_muertas from plantas\_enfermas where estado='muerta';

**24. Cuál es la enfermedad más común.**

select tipo\_enfermedad, count(\*) as cantidad from plantas\_enfermas group by tipo\_enfermedad order by cantidad desc limit 1;

**25. Cuál es el lote donde más se han enfermado las plantas.**

select l.id\_lote, count(p.id\_planta\_enferma) as 'cantidad\_enfermedades' from plantas\_enfermas p

inner join monitoreos m on p.id\_monitoreo=m.id\_monitoreo

inner join cuidado\_cultivo c on m.id\_cuidado\_cultivo=c.id\_cuidado\_cultivo

inner join lotes l on c.id\_lote=l.id\_lote group by l.id\_lote order by cantidad\_enfermedades desc limit 1;

select \* from plantas\_enfermas;

**26. Listar todos los monitoreos de una planta que murió.**

select m.id\_monitoreo, count(m.id\_monitoreo) as cantidad\_monitoreo from monitoreos m inner join plantas\_enfermas p on p.id\_monitoreo=m.id\_monitoreo where p.estado= 'muerta' group by 1;

**27. Listar las plantas enfermas desde la más reciente a la más antigua, con el nombre de la enfermedad, fechas, y ubicación de la planta.**

select \* from plantas\_enfermas where estado='Enferma' order by fecha\_entrada desc;

**28. Listar el empleado con mayor número de monitoreos.**

select e.id\_empleado, e.nombre1, e.nombre2, e.apellido1, e.apellido2, count(m.id\_monitoreo) as 'cantidad\_monitoreos'

from empleados e inner join monitoreos m on e.id\_empleado=m.id\_empleado

group by e.id\_empleado order by cantidad\_monitoreos desc limit 1;

**29. Listar el lote con mayor número de monitoreos.**

select l.id\_lote, l.ancho, l.largo, l.unidad\_de\_medida, count(m.id\_monitoreo) as 'cantidad\_monitoreos'

from lotes l inner join cuidado\_cultivo c on l.id\_lote=c.id\_lote inner join monitoreos m on c.id\_cuidado\_cultivo=m.id\_cuidado\_cultivo

group by l.id\_lote

order by cantidad\_monitoreos desc limit 1;

**30. Listar el agroquímico mas utilizado en el cuidado del cultivo y cuantos quedan disponibles.**

select a.nombre\_marca, a.stock, count(c.id\_agroquimico) as cantidad\_utilizada

from agroquimicos a inner join cuidado\_cultivo c on a.id\_agroquimico= c.id\_agroquimico

group by a.id\_agroquimico order by cantidad\_utilizada desc limit 1;

**31. Listar los tipos de cuidado.**

select tipo\_cuidado from cuidado\_cultivo;

**32. Calcular el precio total de compra por cada agroquímico.**

select a.nombre\_marca, sum(c.cantidad \* c.costo) as precio\_total\_compra

from agroquimicos a inner join compras c on a.id\_compra=c.id\_compra

group by a.id\_agroquimico;

**33. Calcular el total de compra de los agroquimicos.**

select sum(c.cantidad \* c.costo) as precio\_total\_compra\_agroquimicos

from agroquimicos a inner join compras c on a.id\_compra=c.id\_compra;

**34. Listar los agroquímicos no disponibles con el número del proveedor.**

select id\_agroquimico, a.nombre\_marca, a.tipo, a.categoria, a.descripcion, p.telefono as telefono\_proveedor from agroquimicos a inner join compras c on a.id\_compra= c.id\_compra inner join proveedores p on c.id\_proveedor=p.id\_proveedor;

**35. Listar los métodos de aplicación de cada agroquímico con la unidad de medida.**

select id\_agroquimico, nombre\_marca, metodo\_aplicacion, unidad\_de\_medida from agroquimicos;

**36. calcular la cantidad de plantas por lote.**

select id\_lote,cantidad\_de\_plantas from lotes;

**37. Listar los tipos de maquinaria que hay en la finca que se encuentran operativas.**

select tipo from maquinaria;

**38. Listar las máquinas en mal estado.**

select \* from maquinaria where estado='en mantenimiento';

**39. Listar las máquinas en buen estado.**

select tipo from maquinaria where estado='operativa';

**40.** **Listar el tipo de maquina con su categoría y el tipo de combustible que se usa ordenadas de forma alfabeticamente por el tipo.**

select tipo, categoria, tipo\_combustible from maquinaria order by 1 asc;

**41. Cuál es el combustible más utilizado.**

select tipo\_combustible, count(tipo\_combustible) as 'cantidad\_combustible' from maquinaria group by 1 order by cantidad\_combustible desc limit 1;

**42 Cual es la máquina más utilizada en los procesos.**

select tipo, count(tipo)as 'cantidad\_maquina' from maquinaria group by 1 order by cantidad\_maquina desc limit 1;

**43. Cuál es el promedio de duración de cada proceso.**

select avg(timestampdiff(day,fecha\_fin, fecha\_inicio)) as promedio\_dias\_procesos from procesos ;

**44. Listar el área de la finca donde se realiza cada proceso.**

select p.caracteristicas, af.nombre from procesos p inner join areas\_finca af on p.id\_area\_finca=af.id\_area\_finca;

**45. Listar los procesos con problemas y la descripción del problema.**

select id\_proceso, caracteristicas, problema\_proceso, descripcion\_problema from procesos where problema\_proceso= 1;

**46. Cual fue el proceso con mayor número de problemas.**

select id\_proceso, caracteristicas,count(problema\_proceso) as 'cantidad\_problemas' from procesos where problema\_proceso= 1 group by 1,2 order by cantidad\_problemas desc limit 1 ;

**47. Cuál es el número de problemas de cada proceso.**

select id\_proceso, count(problema\_proceso) as cantidad\_problemas, descripcion\_problema from procesos where problema\_proceso= 1 group by 1;

**48. Listar los agroquímicos que faltan por pagar.**

select a.id\_agroquimico, a.nombre\_marca, a.tipo, a.categoria, c.deuda from agroquimicos a inner join compras c on c.id\_compra= a.id\_compra where c.deuda>0;

**49. Listar las 3 compras que tengan una deuda más alta.**

select id\_compra, nombre\_producto, cantidad, costo, deuda from compras order by deuda desc limit 3;

**50. Calcular el total de las deudas por compras.**

select id\_compra, sum(deuda) as total\_deuda from compras group by 1;

**50. Calcular el total de las deudas por compras.**

select sum(deuda) from compras;

**51. Cual fue la compra mas reciente y el agroquímico que se compró.**

select tipo, fecha\_compra from compras

inner join agroquimicos on compras.id\_agroquimicos= agroquimicos.id\_agroquimicos

order by 1 desc limit 1;

**52. Calcular las horas de trabajo del empleado sin contar el almuerzo.**

select nombre1, apellido1,

(timestampdiff(hour, hora\_entrada, hora\_salida)-1) as horas\_trabajo

from horarios\_empleados

inner join empleados on horarios\_empleados.id\_empleado=empleados.id\_empleado

group by 1,2,3;

**53. Listar el cargo de cada empleado y el sueldo.**

select nombre1, apellido1, nombre, salario from empleados

inner join cargos on empleados.id\_cargo=cargos.id\_cargo;

**54. Listar los empleados que han sido despedidos.**

select nombre1, apellido1, estado from empleados where estado='despedido';

**55. Quien es el empleado más antiguo de la finca, el cargo y el total de pagos que ha recibido.**

select nombre1, apellido1, fecha\_ingreso, sum(total\_pagado)as total\_Pagos from empleados

inner join pagos\_empleados on empleados.id\_empleado=pagos\_empleados.id\_empleado

group by 1,2,3 order by fecha\_ingreso asc limit 1;

**56. Quien fue el empleado que duro menos tiempo trabajando en la finca.**

select nombre1, apellido1, timestampdiff(day, fecha\_ingreso,fecha\_salida)dias\_trabajo from empleados

where not fecha\_salida is null

order by 3 asc limit 1;

**57. Quien es el empleado más viejo y el cargo que ejerce.**

select nombre1, apellido1, timestampdiff(year, fecha\_nacimiento, now())as edad, nombre from empleados

inner join cargos on empleados.id\_cargo = cargos.id\_cargo

order by 3 desc limit 1;

**58. Quien es el empleado más joven y el cargo que ejerce.**

select nombre1, apellido1, timestampdiff(year, fecha\_nacimiento, now())as edad, nombre from empleados

inner join cargos on empleados.id\_cargo = cargos.id\_cargo

order by 3 asc limit 1;

**59. Calcular el total de pagos de cada empleado.**

select nombre1, apellido1, sum(total\_pagado) from empleados

inner join pagos\_empleados on empleados.id\_empleado= pagos\_empleados.id\_empleado

group by 1,2;

**60. Calcular el número de pagos de cada empleado.**

select nombre1, apellido1, count(total\_pagado)as totalPagos from empleados

inner join pagos\_empleados on empleados.id\_empleado= pagos\_empleados.id\_empleado

group by 1,2;

**61. Calcular el promedio del total pagado a los empleados.**

select nombre1, apellido1, avg(total\_pagado)as promedioPagado from empleados

inner join pagos\_empleados on empleados.id\_empleado= pagos\_empleados.id\_empleado

group by 1,2;

**62. Listar los pagos del empleado con su fecha.**

select nombre1, nombre2, fecha\_pago from empleados

inner join pagos\_empleados on empleados.id\_empleado= pagos\_empleados.id\_empleado;

**63. Calcular el número de compras por cada cliente**

select nombre1, apellido1, count(ventas.id\_cliente) from ventas

inner join clientes on ventas.id\_cliente=clientes.id\_cliente

group by 1,2;

**64. Calcular el promedio de compra por cada cliente.**

select clientes.id\_cliente,nombre1, apellido1, avg(precio\_a\_pagar)as promedioTotalCompra from ventas

inner join clientes on ventas.id\_cliente=clientes.id\_cliente

group by 1;

**65. Cual es el producto más vendido y mostrar si está disponible.**

select nombre, count(ventas.id\_producto)as totalVentas, stock from ventas

inner join productos on ventas.id\_producto= productos.id\_producto

group by 1,3 order by 2 desc limit 1;

**66. Quien es el empleado con mayor número de ventas.**

select nombre1, apellido1, count(ventas.id\_empleado)as totalVentas from empleados

inner join ventas on empleados.id\_empleado= ventas.id\_empleado

group by 1,2 order by 3 desc limit 1;

**67. Quien es el cliente con mayor número de compras.**

select nombre1, apellido1, count(ventas.id\_cliente) from clientes

inner join ventas on clientes.id\_cliente=ventas.id\_cliente

group by 1,2 order by 3 desc limit 1;

**68. Quien es el cliente con menor número de compras.**

select nombre1, apellido1, count(ventas.id\_cliente) from clientes

inner join ventas on clientes.id\_cliente=ventas.id\_cliente

group by 1,2 order by 3 asc limit 1;

**69. Listar las ventas que tuvieron problemas en el envío y el porqué.**

select id\_venta, descripcion\_problema from ventas

inner join registro\_envio on ventas.id\_venta= registro\_envio.id\_registro\_envio

where problema\_envio=1;

**70. Cuál es el método de transporte más utilizado en el envío.**

select tipo, count(id\_transporte) from registro\_envio

inner join maquinaria on registro\_envio.id\_transporte= maquinaria.id\_maquinaria

group by 1 order by 2 desc limit 1;

**71. Listar los métodos de transporte que hay en la finca.**

select tipo from maquinaria where categoria='transporte';

**72. Calcular el número de ventas que no tuvieron problemas en el envío.**

select count(ventas.id\_venta) from ventas

inner join registro\_envio on ventas.id\_envio= registro\_envio.id\_registro\_envio

where problema\_envio = 0;

**73. Calcular el número de ventas que tuvieron problemas con el envío.**

select count(ventas.id\_venta) from ventas

inner join registro\_envio on ventas.id\_envio= registro\_envio.id\_registro\_envio

where problema\_envio = 1;

**74. Quien es el empleado con mayores problemas de envío.**

select nombre1, apellido1, count(registro\_envio.id\_empleado) from empleados

inner join registro\_envio on empleados.id\_empleado= registro\_envio.id\_empleado

where problema\_envio=1

group by 1,2 order by 3 desc limit 1;

**75. Listar cuanto tiempo duró en llegar cada pedido.**

select id\_registro\_envio, timestampdiff(day,fecha\_salida,fecha\_llegada)

from registro\_envio;

**76. Cual fue la venta con mayor tiempo en tránsito.**

select id\_venta, timestampdiff(day,fecha\_salida,fecha\_llegada)as dias from ventas

inner join registro\_envio on ventas.id\_envio= registro\_envio.id\_registro\_envio

group by 1 order by 2 desc limit 1;

**77. Cuál es el método de pago más común.**

select tipo, count(registro\_envio.id\_metodo\_pago) from registro\_envio

inner join metodo\_pago on registro\_envio.id\_metodo\_pago= metodo\_pago.id\_metodo\_pago

group by 1 order by 2 desc;

**78. Listar los productos vencidos.**

select nombre from productos

where fecha\_vencimiento < now();

**79. Quien es el cliente más antiguo.**

select nombre1,apellido1, timestampdiff(day, fecha\_registro,now())

from clientes order by 3 desc limit 1;

**80. Listar los descuentos disponibles.**

select id\_bonificacion\_descuento, descripcion from bonificacion\_descuento

where tipo='descuento' and estado='activo';

**81. Listar las bonificaciones disponibles.**

select id\_bonificacion\_descuento, descripcion from bonificacion\_descuento

where estado='activo' and tipo= 'bonificacion';

**82. Listar los descuentos que no están disponibles.**

select id\_bonificacion\_descuento, descripcion from bonificacion\_descuento

where tipo='descuento' and estado='inactiva';

**83. Listar las bonificaciones que no están disponibles.**

select id\_bonificacion\_descuento, descripcion from bonificacion\_descuento

where estado='inactiva' and tipo= 'bonificacion';

**84. consulta para saber cuál es la bonificación más utilizada.**

select bonificacion\_descuento.id\_bonificacion\_descuento,

count(pagos\_empleados.id\_bonificacion\_descuento)as numeroBonificacionesUsadas

from pagos\_empleados

inner join bonificacion\_descuento on pagos\_empleados.id\_bonificacion\_descuento=

bonificacion\_descuento.id\_bonificacion\_descuento

where tipo='bonificacion' group by 1 order by 2 desc limit 1;

**85.Cuál es la diferencia del pago establecido con el pago dado al empleado.**

select salario,total\_pagado,total\_pagado-salario as diferencia from pagos\_empleados

inner join empleados on pagos\_empleados.id\_empleado= empleados.id\_empleado

inner join cargos on empleados.id\_cargo= cargos.id\_cargo;

**86. Listar los empleados que están y estuvieron en receso por enfermedad y la fecha.**

select nombre1, apellido1, vacaciones.fecha\_salida, fecha\_llegada from empleados

inner join vacaciones on empleados.id\_empleado= vacaciones.id\_empleado

where motivo='descanso medico';

**87. Listar los motivos de vacaciones.**

select distinct motivo from vacaciones;

**88. Calcular los días de vacaciones por empleado.**

select nombre1, apellido1,timestampdiff(day, vacaciones.fecha\_salida,fecha\_llegada)as dias

from empleados inner join vacaciones on empleados.id\_empleado= vacaciones.id\_empleado;

**89. Calcular el número de empleados que hay en la finca.**

select count(\*) from empleados;

**90. Calcular el número de empleados que hay en cada área.**

select areas\_finca.nombre, count(empleados.id\_empleado) as num\_empleados

from empleados inner join cargos on empleados.id\_cargo= cargos.id\_cargo

inner join areas\_finca on cargos.id\_area\_finca = areas\_finca.id\_area\_finca

group by 1;

**91. Calcular el total pagado por daños.**

select sum(costo\_reparacion) from daños;

**92. Calcular en días el tiempo que pasó en cada reparación.**

select timestampdiff(day, fecha\_reparacion, fecha\_reparacion\_terminada) from daños;

**93. Cuantos proveedores hay.**

select count(\*) from proveedores;

**94. Cuál es el proveedor más antiguo.**

select nombre1, apellido1, timestampdiff(day,fecha\_de\_registro, now())dias from proveedores

group by 1,2,3 order by 3 desc limit 1;

**95. Cuantos problemas de cosecha hay en este momento en la finca.**

select count(\*) from cosecha

where problema\_cosecha=1 and fecha\_cosecha>'2024-01-01'

and fecha\_cosecha<now();

**96. Cuáles son los lotes que faltan por monitoreo.**

select lotes.id\_lote from lotes

inner join cuidado\_cultivo on lotes.id\_lote=cuidado\_cultivo.id\_lote

left join monitoreos on cuidado\_cultivo.id\_cuidado\_cultivo= monitoreos.id\_cuidado\_cultivo

where monitoreos.id\_monitoreo is null;

**97. Cuál es el tiempo en días, que tomo cada planta enferma hasta quedar sana.**

select numero\_planta\_ubicacion, timestampdiff(day,fecha\_entrada,fecha\_salida) from plantas\_enfermas

where estado='sana';

**98. Cuanto tiempo duro cada planta que enfermó hasta morir.**

select numero\_planta\_ubicacion, timestampdiff(day,fecha\_entrada,fecha\_salida) from plantas\_enfermas

where estado='murio';

**99. Cuál ha sido el costo total de cada herramienta comprada.**

select tipo, sum(costo) from herramientas

inner join compras on herramientas.id\_compra = compras.id\_compra

group by 1;

**100. cuantas vacaciones han tenido los empleados este año**

select nombre1, apellido1, vacaciones.fecha\_salida, vacaciones.fecha\_llegada, count(vacaciones.id\_empleado) from empleados

inner join vacaciones on empleados.id\_empleado= vacaciones.id\_empleado

where vacaciones.fecha\_salida and fecha\_llegada>'2024-01-01' and vacaciones.fecha\_salida and fecha\_llegada<now()

group by 1,2,3,4;

# **Funciones**

Las funciones de la base de datos están diseñadas para permitir al usuario realizar operaciones como la creación, edición y eliminación de registros e implementación de consultas. Esto hace que el usuario puede tener un mejor manejo de la información almacenada de una manera más fácil.

use fincaCacao

-- ############################################################

**1. obtener el total de compras realizados por un cliente.**

delimiter //

create function total\_ventas\_productos( id int)

returns int

deterministic

begin

declare total int;

select count(v.id\_cliente) into total from ventas v inner join clientes c on v.id\_cliente=c.id\_cliente where v.id\_cliente=id;

return total;

end //

delimiter ;

select total\_ventas\_productos(2);

**2. obtener el cliente con mayor total en compras**

delimiter //

create function mayor\_pedidoo()

returns varchar(50)

deterministic

begin

declare nombre\_cliente varchar(50);

select c.nombre1 into nombre\_cliente from ventas v inner join clientes c on v.id\_cliente=C.id\_cliente group by 1 order by sum(v.total) desc limit 1 ;

return nombre\_cliente;

end //

delimiter ;

select mayor\_pedidoo() as clienteConMayorTotal, v.total from ventas v inner join clientes c on v.id\_cliente=C.id\_cliente order by v.total desc limit 1 ;

1. **Esta función le permite al usuario registrar más herramientas.**

delimiter //

create function añadir\_herramienta(categoria2 varchar(50), tipo2 varchar(50), stock2 int, estado2 varchar(50), tipo\_combustible2 varchar(50), id\_compra2 int, id\_area\_finca2 int)

returns integer deterministic

begin

insert into herramientas(categoria, tipo, stock, estado, tipo\_combustible, id\_compra, id\_area\_finca)

values (categoria2, tipo2, stock2, estado2, tipo\_combustible2, id\_compra2, id\_area\_finca2);

return last\_insert\_id();

end //

delimiter ;

select añadir\_herramienta('corte', 'machete', 10, 'Dañado', null, 1,1);

**4. Esta función le permite al usuario eliminar un registro de las herramientas.**

delimiter //

create function eliminar\_herramienta(id int)

returns int deterministic

begin

delete from herramientas

where id\_herramienta=id;

return 1;

end;

// delimiter ;

-- drop function eliminar\_herramienta;

select eliminar\_herramienta(51);

**5.Esta función le permite al usuario registrar una compra.**

select \* from compras;

delimiter //

create function añadir\_compra(nombre\_producto varchar(50), cantidad int, costo int, fecha\_compra date, estado varchar(50), deuda int, id\_finca int, id\_proveedor int)

returns integer deterministic

begin

insert into compras(nombre\_producto, cantidad, costo, fecha\_compra, estado, deuda, id\_finca, id\_proveedor )

values (nombre\_producto, cantidad, costo, fecha\_compra, estado, deuda, id\_finca, id\_proveedor );

return last\_insert\_id();

end //

delimiter ;

select añadir\_compra('Fungicida Organico', 10, 2000, now(), 'pagado', 0, 1,2);

**6. Esta función le permite al usuario registrar una planta enferma.**

delimiter //

create function añadir\_planta\_enferma(estado varchar(50), tipo\_enfermedad varchar(50), nombre\_enfermedad varchar(90), descripcion varchar(150), fecha\_entrada varchar(50), fecha\_salida varchar(50), numero\_surco\_ubicacion int, numero\_planta\_ubicacion int, id\_monitoreo int )

returns integer deterministic

begin

insert into plantas\_enfermas(estado, tipo\_enfermedad, nombre\_enfermedad, descripcion, fecha\_entrada, fecha\_salida, numero\_surco\_ubicacion, numero\_planta\_ubicacion, id\_monitoreo)

values(estado, tipo\_enfermedad, nombre\_enfermedad, descripcion, fecha\_entrada, fecha\_salida, numero\_surco\_ubicacion, numero\_planta\_ubicacion, id\_monitoreo);

return last\_insert\_id();

end //

delimiter ;

select añadir\_planta\_enferma('Enferma', 'virus', 'mildiu', 'Hongo que afecta a las hojas', now(), null,2,8,1);

**7. Esta función le permite al usuario registrar un nuevo daño.**

delimiter //

create function cambiar\_estado\_herramienta(id int, estado2 varchar(50))

returns integer deterministic

begin

update herramientas set estado= estado2 where id\_herramienta= id;

return last\_insert\_id();

end //

delimiter ;

select cambiar\_estado\_herramienta(2,'Dañado');

**8. Esta función le permite al usuario registrar un nuevo monitoreo.**

delimiter //

create function añadir\_monitoreo(fecha\_monitoreo date, descripcion\_resultado varchar(50), observaciones varchar(150), frecuencia varchar(50), id\_cuidado\_cultivo int, id\_empleado int)

returns integer deterministic

begin

insert into monitoreos(fecha\_monitoreo, descripcion\_resultado , observaciones , frecuencia, id\_cuidado\_cultivo , id\_empleado )

values(fecha\_monitoreo, descripcion\_resultado, observaciones, frecuencia, id\_cuidado\_cultivo, id\_empleado );

return 1;

end //

delimiter ;

select añadir\_monitoreo(now(), 'Observacion de Mildiu en algunas plantas', 'Aplicacion de fungicida en plantas afectadas', 'Mensual', 1,1);

**9. Esta función le permite al usuario añadir un nuevo empleado.**

delimiter //

create function añadir\_empleado(nombre1 varchar(50),nombre2 varchar(50),apellido1 varchar(50),apellido2 varchar(50),cedula varchar(50), telefono varchar(50),correo\_electronico varchar(50),fecha\_ingreso date,fecha\_salida date, estado varchar(50),fecha\_nacimiento date,id\_finca int,id\_cargo int)

returns integer deterministic

begin

insert into empleados(nombre1,nombre2,apellido1,apellido2,cedula, telefono,correo\_electronico,fecha\_ingreso,fecha\_salida, estado,fecha\_nacimiento,id\_finca,id\_cargo)

values(nombre1,nombre2,apellido1,apellido2,cedula, telefono,correo\_electronico,fecha\_ingreso,fecha\_salida, estado,fecha\_nacimiento,id\_finca,id\_cargo);

return 1;

end //

delimiter ;

select añadir\_empleado('Camilo', 'Antonio', 'vega', 'Rodrigues', '109999233', '123454434', 'camilo35@gmail.com',now(), null, 'Activo', '1999-05-12', 1,2);

**10. Esta función le permite al usuario cambiar la frecuencia de pagos en el cargo.**

delimiter //

create function editar\_frecuencia\_pago(id int, frecuencia\_pago2 varchar(50))

returns integer deterministic

begin

update cargos set frecuencia\_pago= frecuencia\_pago2 where id\_cargo=id;

return last\_insert\_id();

end //

delimiter ;

select editar\_frecuencia\_pago(1, 'semanal');

**11. Esta función le permite al usuario añadir una nueva bonificación.**

delimiter //

create function crearBonificacion(porcentajeI decimal(10,2), descripcionI varchar (150)

, estadoI varchar(50), tipoI varchar(50))

returns integer deterministic

begin

insert into bonificacion\_descuento (porcentaje,descripcion,estado,tipo)

values (porcentajeI,descripcionI,estadoI,tipoI);

return last\_insert\_id();

end;

// delimiter ;

select crearBonificacion(12.00,'Bonificación por cumplimiento de metas','Inactiva','Bonificacion');

**12. Esta función le permite al usuario añadir un nuevo descuento.**

delimiter //

create function crearDescuento(porcentaje decimal(10,2), descripcion varchar(150),

estado varchar(50), tipo varchar(50))

returns integer deterministic

begin

insert into bonificacion\_descuento (porcentaje,descripcion,estado,tipo)

values (porcentaje,descripcion,estado,tipo);

return last\_insert\_id();

end;

// delimiter ;

select crearDescuento(15, 'Descuento por mejor comprador', 'Inactiva', 'Descuento');

**13. Esta función le permite al usuario añadir un nuevo registro de pagos.**

delimiter //

create function registroPagos(fecha date, total decimal (10,2), id\_empleado int,

id\_bono\_o\_descuento int)

returns integer deterministic

begin

insert into pagos\_empleados(fecha\_pago,total\_pagado,id\_empleado,id\_bonificacion\_descuento)

values (fecha, total, id\_empleado ,id\_bono\_o\_descuento);

return last\_insert\_id();

end;

// delimiter ;

select registroPagos('2024-11-26',840.00,1,11);

**14. Esta función le permite al usuario eliminar un método de pago.**

delimiter //

create function eliminar\_metodoPago(id int)

returns int deterministic

begin

delete from metodo\_pago

where id\_metodo\_pago=id;

return 1;

end;

// delimiter ;

select eliminar\_metodoPago(100);

**15. Esta función le permite al usuario añadir un nuevo método de pago.**

delimiter //

create function crearMetodoPago(tipo varchar(50), fecha\_pago date)

returns int deterministic

begin

insert into metodo\_pago (tipo, fecha\_pago)

values (tipo, fecha\_pago);

return last\_insert\_id();

end;

// delimiter ;

select crearMetodoPago('Bancolombia','2024-11-26');

**16. Esta función le permite al usuario cambiar la descripción del problema en la entidad proceso.**

delimiter //

create function cambiarDescripcionProceso(id int,descripcion varchar(200))

returns int deterministic

begin

update procesos

set descripcion\_problema= descripcion where id=id\_proceso;

return last\_insert\_id();

end;

// delimiter ;

select cambiarDescripcionProceso(17,'Problema con el motor de molienda');

**17. Esta función le permite al usuario cambiar el registro de la fecha de vencimiento del agroquímico.**

delimiter //

create function cambiarFechaVencimientoAgroquimico(id int, fecha date)

returns int deterministic

begin

update agroquimicos

set fecha\_verncimiento= fecha where id=id\_agroquimico;

return last\_insert\_id();

end;

// delimiter ;

select cambiarFechaVencimientoAgroquimico(1,'2024-11-26');

**18. Esta función le permite al usuario cambiar el stock del agroquímico.**

delimiter //

create function cambiarStockAgroquimico(id int, stock int)

returns int deterministic

begin

update agroquimicos

set stock= stock where id=id\_agroquimico;

return last\_insert\_id();

end;

// delimiter ;

select cambiarStockAgroquimico(4,0);

**19. Esta función le permite al usuario cambiar el stock de las herramientas.**

delimiter //

create function cambiarStockHerramientas (id int, stock int)

returns int deterministic

begin

update herramientas

set stock= stock where id= id\_herramienta;

return last\_insert\_id();

end;

// delimiter ;

select cambiarStockHerramientas(25,10);

**20. Esta función le permite al usuario cambiar el precio del producto**

delimiter //

create function cambiarPrecioProducto(id int, precio decimal(10,2))

returns int deterministic

begin

update productos

set precio= precio where id= id\_producto;

return last\_insert\_id();

end;

// delimiter ;

select cambiarPrecioProducto(1,'25.00');

# **Procedimientos**

Estos procedimientos son diseñados para facilitar la gestión y el análisis de la información.

use fincaCacao

**1. Procedimiento para obtener los productos vendidos por cliente y el total de compra y el empleado que realizo la venta.**

delimiter //

create procedure productos\_Vendidos\_empleado()

begin

select e.nombre1 as nombre\_empleado,c.nombre1, c.apellido1, p.nombre, sum(precio\_a\_pagar) from ventas v

inner join clientes c on v.id\_cliente=c.id\_cliente

inner join productos p on v.id\_producto=p.id\_producto

inner join empleados e on e.id\_empleado=v.id\_empleado

group by 1,2,3,4;

end;

// delimiter ;

call productos\_Vendidos\_empleado();

**2. Procedimiento para calcular el total de la deuda de las compras**

delimiter //

create procedure calcular\_deuda\_compras()

begin

declare total\_deuda decimal(10,2);

select sum(deuda) into total\_deuda from compras;

select concat('el total es:', total\_deuda) as total\_deuda;

end //

delimiter ;

call calcular\_deuda\_compras();

**3. Procedimiento para cambiar el estado de compra cuando el producto**

-- sea pagado y registrarlo en la tabla correspondiente.

delimiter //

create procedure cambiar\_estado\_compra(in id int)

begin

if (select deuda from compras where id\_compra=id)= 0 then

update compras set estado = 'Pagado' where id\_compra=id;

end if;

end //

delimiter ;

call cambiar\_estado\_compra(1);

**4. Procedimiento para calcular el tiempo que duró en producción cada producto.**

delimiter //

create procedure tiempo\_produccion()

begin

declare tiempo\_produccion int;

select p.id\_producto, p.nombre as nombre\_producto, datediff(pr.fecha\_fin, pr.fecha\_inicio)as dias\_produccion

from productos p inner join procesos pr on p.id\_proceso=pr.id\_proceso;

end //

delimiter ;

call tiempo\_produccion();

**5. Procedimiento para cambiar el estado de la maquinaria cuando sea reparada y registrarlo en la tabla correspondiente.**

delimiter //

create procedure cambiar\_estado\_maquinaria(in id int)

begin

if exists( select \* from daños where id\_maquina= id and fecha\_reparacion\_terminada is not null) then

update maquinaria set estado='Reparada' where id\_maquinaria=id;

end if;

end //

delimiter ;

call cambiar\_estado\_maquinaria(1);

**6. Procedimiento para generar reportes de empleados con salarios y cargos**

delimiter //

create procedure reporte\_empleados()

begin

select e.id\_empleado,e.nombre1,e.apellido1, c.nombre as cargo, c.salario, a.nombre as area

from empleados e inner join cargos c on e.id\_cargo=c.id\_cargo

inner join areas\_finca a on c.id\_area\_finca=a.id\_area\_finca order by c.salario desc;

end //

delimiter ;

call reporte\_empleados();

1. **Procedimiento para asignar herramientas a un área.**

delimiter //

create procedure añadir\_herramienta\_area(in id\_herramienta int, in id\_area int)

begin

update herramientas set id\_area\_finca= id\_area where id\_herramienta= id\_herramienta;

end //

delimiter ;

call añadir\_herramienta\_area(1,2);

**8. procedimiento para sacar el informe de la cosecha realizada por un empleado.**

delimiter //

create procedure informe\_cosecha\_empleado(in id int)

begin

select c.id\_cosecha, c.fecha\_cosecha, c.cantidad, c.unidad\_de\_medida, c.descripcion\_problema as problema\_cosecha,

h.tipo as herramienta\_utilizada from cosecha c inner join herramientas h

on c.id\_herramienta=h.id\_herramienta

where c.id\_empleado= id order by c.fecha\_cosecha desc;

end //

delimiter ;

call informe\_cosecha\_empleado(4);

**9 procedimiento para cambiar el stock de una herramienta segun la cantidad que tenga dañadas**

delimiter //

create procedure actualizar\_stock\_herramienta(in id int, in cantidad\_dañada int)

begin

update herramientas set stock= stock-cantidad\_dañada where id\_herramienta=id;

end //

delimiter ;

call actualizar\_stock\_herramienta(2,2);

**10. Procedimientos para obtener los empleados con los cargos que más pagos han recibido y si se encuentra activo.**

delimiter //

create procedure cargos\_Empleados\_activos()

begin

select nombre1, apellido1, nombre, estado, count(total\_pagado) from empleados

inner join cargos on empleados.id\_cargo=cargos.id\_cargo

inner join pagos\_empleados on empleados.id\_empleado= pagos\_empleados.id\_empleado

group by 1,2,3,4 order by 5 desc;

end;

// delimiter ;

call cargos\_Empleados\_activos();

**11. Procedimiento para obtener el total de compras realizadas en la finca con detalles del producto comprado.**

delimiter //

create procedure registroCompras()

begin

select nombre\_producto, sum(costo) from compras

group by 1;

end;

// delimiter ;

call registroCompras();

**12. Procedimiento para obtener un reporte de cosechas con empleado, por cultivo y total de cosechas.**

delimiter //

create procedure reporteCosechas()

begin

select nombre as nombreCultivo, cantidad\_de\_plantas, nombre1,

apellido1, count(cantidad) from cosecha

inner join empleados on cosecha.id\_empleado= empleados.id\_empleado

inner join cultivo on cosecha.id\_cultivo=cultivo.id\_cultivo

group by 1,2,3,4;

end;

// delimiter ;

call reporteCosechas();

**13. Procedimientos para obtener los empleados con los cargos que más pagos han recibido.**

delimiter //

create procedure cargosEmpleados()

begin

select nombre1, apellido1, nombre, count(total\_pagado) from empleados

inner join cargos on empleados.id\_cargo=cargos.id\_cargo

inner join pagos\_empleados on empleados.id\_empleado= pagos\_empleados.id\_empleado

group by 1,2,3 order by 4 desc;

end;

// delimiter ;

call cargosEmpleados();

**14. Procedimiento para obtener los productos vendidos por cliente y el total de compra.**

delimiter //

create procedure productosVendidos()

begin

select nombre1, apellido1, nombre, sum(precio\_a\_pagar) from ventas

inner join clientes on ventas.id\_cliente=clientes.id\_cliente

inner join productos on ventas.id\_producto=productos.id\_producto

group by 1,2,3;

end;

// delimiter ;

call productosVendidos();

**15. Procedimiento para obtener la ubicación de las herramientas y el proceso que realiza.**

delimiter //

create procedure ubicacionHerramientas()

begin

select tipo, nombre as ubicacion, descripcion from herramientas

inner join areas\_finca on herramientas.id\_area\_finca= areas\_finca.id\_area\_finca;

end;

// delimiter ;

call ubicacionHerramientas();

**16. Procedimiento para obtener los empleados que no han tenido vacaciones en este año.**

delimiter //

create procedure vacacionesEmpleados()

begin

select nombre1, apellido1,vacaciones.fecha\_salida ,fecha\_llegada from empleados

inner join vacaciones on empleados.id\_empleado=vacaciones.id\_empleado

where not fecha\_llegada >= '2024-01-01' and fecha\_llegada <= now();

end;

// delimiter ;

call vacacionesEmpleados();

**17. Procedimiento para calcular el total de pedidos vendidos por cada cliente.**

delimiter //

create procedure totalClientes()

begin

select nombre1, apellido1, count(ventas.id\_cliente) from ventas

inner join clientes on ventas.id\_cliente= clientes.id\_cliente

group by 1,2;

end;

// delimiter ;

call totalClientes();

**18. Procedimiento para obtener las herramientas que no han sido utilizadas.**

delimiter //

create procedure herramientas\_sin\_usao()

begin

select tipo from herramientas

left join areas\_finca on herramientas.id\_area\_finca=areas\_finca.id\_area\_finca where herramientas.id\_area\_finca is null;

end;

// delimiter ;

call herramientas\_sin\_usao();

**19. Procedimiento para obtener un informe de las plantas enfermas y los problemas detectados.**

delimiter //

create procedure cosechasProblemas()

begin

select id\_planta\_enferma, nombre\_enfermedad, tipo\_enfermedad, plantas\_enfermas.fecha\_entrada as fecha\_enfermedad,

plantas\_enfermas.fecha\_salida as fecha\_enfermedad\_salida, numero\_surco\_ubicacion,numero\_planta\_ubicacion, plantas\_enfermas.descripcion,

monitoreos.descripcion\_resultado as monitoreoResultado, monitoreos.observaciones as observacionesMonitoreo,

fecha\_cuidado,tipo\_cuidado,

herramientas.tipo as tipoHerramientas,

agroquimicos.categoria as categoriaAgroquimico,

agroquimicos.tipo as tipoAgroquimico, cuidado\_cultivo.medida\_dosis,cuidado\_cultivo.unidad\_de\_medida,

agroquimicos.metodo\_aplicacion from plantas\_enfermas

inner join monitoreos on plantas\_enfermas.id\_monitoreo= monitoreos.id\_monitoreo

inner join cuidado\_cultivo on monitoreos.id\_cuidado\_cultivo= cuidado\_cultivo.id\_cuidado\_cultivo

inner join herramientas on cuidado\_cultivo.id\_herramienta=herramientas.id\_herramienta

inner join agroquimicos on cuidado\_cultivo.id\_agroquimico= agroquimicos.id\_agroquimico;

end;

// delimiter ;

-- drop procedure cosechasProblemas;

call cosechasProblemas();

**20. Procedimiento para mostrar el horario de los empleados, con la hora de llegada, hora de salida**

-- y organizarlos de manera asendente

delimiter //

create procedure horarioEmpleados()

begin

select nombre1, apellido1, time(hora\_entrada), time(hora\_salida) from empleados

inner join horarios\_empleados on empleados.id\_empleado= horarios\_empleados.id\_empleado

order by 3 asc;

end;

// delimiter ;

call horarioEmpleados();

# **Triggers**

A continuación, se presentan los trigger para automatizar procesos en la base de datos.

1. **Triguer para cambiar el estado en una maquinaria**

delimiter //

create trigger cambio\_estado\_maquinaria

after update on maquinaria

for each row

begin

insert into daños (tipo, descripcion, fecha\_daño, id\_maquina)

values ('maquinaria', concat('se cambio a:', new.estado), now(), new.id\_maquinaria);

end //

delimiter ;

-- select \* from daños;

1. **Trigger para cambiar el stock de productos despues de una venta**

delimiter //

create trigger cambiar\_stock\_producto

after insert on ventas

for each row

begin

update productos

set stock= stock - new.cantidad\_producto

where id\_producto= new.id\_producto;

end //

delimiter ;

select \* from ventas;

1. **trigger para actualizar el salario de los empleados una vez otorgado el bono**

delimiter //

create trigger cambiar\_salario\_bono

after insert on pagos\_empleados

for each row

begin

if new.id\_bonificacion\_descuento is not null then

update empleados

set estado = 'Bono realizado'

where id\_empleado = new.id\_empleado;

end if;

end //

delimiter ;

1. **trigger para registrar el problema de un envio.**

delimiter //

create trigger registrar\_problema\_envio

before update on registro\_envio

for each row

begin

if new.problema\_envio= 1 then

set new.descripcion\_problema= concat('Problema en el envio: ', now());

end if;

end //

delimiter ;

**5. trigger para actualizar el stock de un agroquimico despues de una compra.**

-- select nombre\_marca from agroquimicos;

-- select \* from compras;

delimiter //

create trigger actualizar\_stock\_agroquimico

after insert on compras

for each row

begin

if exists (select \* from agroquimicos where nombre\_marca= new.nombre\_producto) then

update agroquimicos

set stock = stock - new.cantidad

where nombre\_marca = new.nombre\_producto;

end if;

end //

delimiter ;

-- drop trigger actualizar\_stock\_agroquimico;

1. **trigger que impide que el precio de un producto se actualice a un valor negativo**

delimiter //

create trigger validacion\_precio\_producto

before update on productos

for each row

begin

if new.precio < 0 then

signal sqlstate '45000' set message\_text = 'El precio no puede ser negativo';

end if;

end //

delimiter ;

1. **trigger para actualizar el stock de un producto si se llega a eliminar una venta entonces se recupera la cantidad inicial que tenía el producto.**

delimiter //

create trigger recuperar\_stock\_producto

after delete on ventas

for each row

begin

update productos

set stock = stock + OLD.cantidad\_producto

where id\_producto= OLD.id\_producto;

end //

delimiter ;

1. **triguer para actualizar el estado de una herramienta si ya fue reparada**

delimiter //

create trigger actualizar\_estado\_herramienta

after update on daños

for each row

begin

if new.fecha\_reparacion\_terminada is not null then

update herramientas

set estado= 'Bien'

where id\_herramienta= new.id\_herramienta;

end if;

end //

delimiter ;

-- drop trigger actualizar\_estado\_herramienta;

9. **triguer para registrar las herramientas que cambien del estado bian a dañado automaticamente se crea el registro en la tabla daños**

delimiter //

create trigger registrar\_herramienta\_dañada

after update on herramientas

for each row

begin

if new.estado= 'Dañado' then

insert into daños(tipo, descripcion, fecha\_daño,id\_herramienta)

values('herramienta', 'herramienta dañada', now(), new.id\_herramienta);

end if;

end //

delimiter ;

-- select \* from ventas;

1. **triguer para verificar el limite de stock antes de realizar una compra**

delimiter //

create trigger verificar\_cantidad\_stock

before insert on ventas

for each row

begin

if stock > cantidad\_producto then

signal sqlstate '45100' set message\_text = 'la cantidad del producto excede el stock actual disponible del producto';

end if;

end //

delimiter ;

select \* from clientes;

1. **Cuando se agregan o restan cantidades de los productos, es importante asegurar que no haya valores negativos que representen un inventario inexistente**

delimiter //

create trigger cacao\_cantidad\_negativa

before update on productos

for each row

begin

if new.stock < 0 then

signal sqlstate '45000' set message\_text = 'La cantidad de productos no puede ser negativa';

end if;

end;

// delimiter ;

1. **Evitar la duplicación de empleados**

delimiter //

create trigger no\_duplicar\_empleados

before insert on empleados

for each row

begin

if exists(select \* from empleados where cedula= new.cedula) then

signal sqlstate '45000' set message\_text = 'la cedula del nuevo empleado ya existe';

end if;

end;

// delimiter ;

1. **registrar la cosecha con la fecha actual automaticamente**

delimiter //

create trigger fecha\_cosecha

before insert on cosecha

for each row

begin

set new.fecha\_cosecha= now();

end;

// delimiter ;

insert into cosecha (cantidad,unidad\_de\_medida,problema\_cosecha,descripcion\_problema,id\_empleado,id\_herramienta,id\_cultivo)

values (100, 'Kilogramos', 0,null,10,32,1);

1. registrar la fecha actual automaticamente en los nuevos registros de compras

delimiter //

create trigger fecha\_actual\_compras

before insert on compras

for each row

begin

set new.fecha\_compra= now();

end;

// delimiter ;

insert into compras (nombre\_producto,cantidad,costo, estado, deuda, id\_finca,id\_proveedor)

values ('Sistémico',30,30000,'Pagado',0,1,1 );

1. **cambiar el estado de la planta cuando la fecha de salida es registrada**

delimiter //

create trigger cambiar\_estado\_planta

after update on plantas\_enfermas

for each row

begin

if new.fecha\_salida is not null then

update plantas\_enfermas

set estado= 'Sana' where id\_plantas\_enfermas=new.id\_planta\_enferma;

end if;

end;

// delimiter ;

1. **registrar automaticamente la fecha del monitoreo**

delimiter //

create trigger auto\_fecha\_monitoreo

before insert on monitoreos

for each row

begin

set new.fecha\_monitoreo= now();

end;

// delimiter ;

INSERT INTO monitoreos (descripcion\_resultado, observaciones, frecuencia, id\_cuidado\_cultivo, id\_empleado)

VALUES ('Observación de Mildiu en algunas plantas', 'Aplicación de fungicida en plantas afectadas', 'Mensual', 1, 1);

1. **actualizar stock de productos**

delimiter //

create trigger actu\_stock\_productos

after insert on compras

for each row

begin

update productos

set stock= stock + new.cantidad

where nombre= new.nombre\_producto;

end;

// delimiter ;

1. **no se pueden ingresar cantidades negativas en el total de ventas**

delimiter //

create trigger total\_negativo

before insert on ventas

for each row

begin

if new.precio\_a\_pagar < 0 then

signal sqlstate '45000' set message\_text = 'Total de compra es negativo';

end if;

end;

// delimiter ;

1. **Limite para que no se exceda el precio de un producto**

delimiter //

create trigger producto\_excedido

before insert on productos

for each row

begin

if new.nombre='Granos de cacao'

and new.precio > 50.00 then

signal sqlstate '45000' set message\_text = 'El precio del producto es exagerado';

end if;

end;

// delimiter ;

insert into productos (nombre, empaque, stock, precio, peso, unidad\_de\_medida, fecha\_vencimiento, fecha\_produccion, id\_proceso)

values ('Granos de cacao', 'Bolsa de 10kg', 1000, 70.00, 10.00, 'kg', '2025-12-31', '2024-11-05', 1)

1. **Pagos que exceden el límite de la tarjeta**

delimiter //

create trigger pagos\_limite\_excedido

before insert on ventas

for each row

begin

declare tarjeta decimal(10,2);

set tarjeta= 10.00;

if new.precio\_a\_pagar > tarjeta then

signal sqlstate '45000' set message\_text = 'El pago no puede realizarse. El precio excede el límite de la tarjeta';

end if;

end;

// delimiter ;

insert into ventas (cantidad\_producto, total, precio\_a\_pagar, id\_empleado, id\_cliente, id\_producto, id\_descuento, id\_envio)

values

(5, 125.00, 20.00, 1, 1, 1, 1, 1);

# **Eventos**

A continuación, te presentamos una serie de eventos que se ejecutan de manera automática al cumplir el tiempo estipulado.

**1. evento que elimina los productos que no sean vendidos durante mas de un año.**

delimiter //

create event eliminar\_productoss

on schedule every 1 year

do

begin

delete from productos where id\_producto not in (select id\_producto from ventas where fecha\_venta>= curdate()- interval 1 year );

end //

delimiter ;

**2. evento para cambiar el estado de la maquinaria a en reparacion despues de que se daño**

delimiter //

create event cambiar\_estado\_maquinaria

on schedule every 1 day

do

begin

update maquinaria set estado= 'En reparacion' where id\_maquinaria in(select id\_maquinaria from daños where fecha\_reparacion\_terminada is null);

end //

delimiter ;

**3. evento para cambiar el estado de un producto cuando se vence**

delimiter //

create event estado\_producto\_vencido

on schedule every 1 day

do

begin

update productos

set estado= 'vencido'

where fecha\_vencimiento <= curdate();

end //

delimiter ;

**4. evento para actualizar el stock automaticamente despues de una venta**

delimiter //

create event actualizar\_stock\_ventas

on schedule every 1 day

do

begin

update productos p

inner join ventas v on p.id\_producto=v.id\_producto set p.stock=p.stock - v.cantidad;

end //

delimiter ;

**5. evento para enviar a maquinaria a mantenimiento general**

delimiter //

create event mantenimiento\_maquinaria

on schedule every 3 month

do

begin

update maquinaria set estado='mantenimiento general' where estado='Operativa';

end //

delimiter ;

**6. evento para verificar los agroqumicos vencidos**

delimiter //

create event verificar\_agroquimicos

on schedule every 1 day

do

update agroquimicos set estado= 'Vencido'

where fecha\_vencimiento < now();

end //

delimiter ;

**7. evento para borrar registros antiguos de envios**

delimiter //

create event borrar\_registro\_envio

on schedule every 1 month

do

begin

delete from registro\_envio where fecha\_entrega< date\_sub(now(), interval 2 year);

end //

delimiter ;

**8. evento para saber cuando esta bajito el stock de los agroquimicos**

delimiter //

create event stock\_agroquimicos\_bajo

on schedule every 1 day

do

begin

update agroquimicos set estado='stock bajo'

where stock<10;

end //

delimiter ;

9**. evento para saber cuando esta bajito el stock de las herramientas**

delimiter //

create event stock\_herramientas\_bajo

on schedule every 1 day

do

begin

update herramientas set estado='stock bajo'

where stock<3;

end //

delimiter ;

**10. evento para actualizar el estado de una deuda cada semana mediante se va pagando**

delimiter //

create event verificar\_estado\_deuda

ON schedule every 1 week

do

update compras set estado= case when deuda>0 then 'Pendiente'

else 'pagada'

end //

delimiter

create table reportes\_ventas(

id int primary key auto\_increment,

total\_ventas int,

producto\_mas\_vendido int,

fecha\_reporte int

);

**11. Se debe generar un reporte semanal de ventas para revisar el rendimiento y los productos más vendidos.**

create event generarReporteVentas

on schedule every 1 week

starts '2024-10-01 00:00:00'

do

insert into reportes\_ventas(total\_ventas, producto\_mas\_vendido, fecha\_reporte)

select sum(total),

(select nombre from ventas

inner join productos on ventas.id\_producto=productos.id\_producto

group by nombre order by SUM(total) desc limit 1),

curdate();

**12. mensaje de stock bajo en productos**

create table notificaciones (

id int auto\_increment primary key,

mensaje varchar(200),

fecha date

);

create event notificacion\_stock

on schedule every 1 day

do

insert into notificaciones (mensaje, fecha)

select concat('Stock bajo de productos', nombre, stock)

from productos

where stock < 20;

**13. mensaje de herramientas agotadas**

create event herramientas\_agotadas

on schedule every 1 day

do

insert into notificaciones(mensaje, fecha)

select concat('Herramientas agotadas', categoria, tipo)

from herramientas

where stock < 5;

**14. Actualizar stock de productos**

create event actualizar\_productos\_stock

on schedule every 1 day

do

update productos

inner join ventas on productos.id\_producto=ventas.id\_producto

set productos.stock= productos.stock - ventas.cantidad\_producto

where productos.id\_producto= ventas.id\_producto;

**15 actualizar stock de compras**

create event stock\_compras

on schedule every 1 day

do

update compras

inner join agroquimicos on compras.id\_compra=agroquimicos.id\_compra

set agroquimicos.stock= agroquimicos.stock + compras.cantidad

where agroquimicos.categoria = compras.nombre\_producto;

**16 productos caducados**

create event productos\_vencidos

on schedule every 1 day

do

select concat('El Productos', nombre, 'esta a punto de caducar')

from productos

where fecha\_vencimiento<= curdate()+ interval 30 day;

**17 eliminar registros antiguos**

create event eliminar\_registros\_antiguos

on schedule every 1 month

do

delete from compras

where fecha\_compra<= curdate()- interval 2 year;

**18 evento para actualizar el precio de un producto cada año**

create event precio\_producto\_año

on schedule every 1 year

starts '2023-01-01'

do

update productos

set precio= precio \* 1.05

where nombre='Granos de cacao';

**19 Recordatorio del mantenimiento de la maquinaria**

delimiter //

create event if not exists

recordatorio\_mantenimiento

on schedule every 1 day

starts '2024-01-01 8:00:00'

do

begin

declare mensaje varchar(200);

set mensaje = 'Recordatorio de mantenimiento: Revise el estado de la maquinaria';

insert into logs(mensaje,fecha)

values (mensaje,now());

end;

// delimiter ;

**20. mensaje de contratos vencidos**

create event contratos\_vencidos

on schedule every 1 day

starts '2024-01-01 8:00:00'

do

update empleados

set estado= 'Inactivo'

where fecha\_vencimiento <= now()