



ADMINISTRADOR DE DEPOSITO AUTONOMO

Base de datos

[Resumen](#)

Información total para depósitos de última generación

Gabriel Shulman

Gabriel.esteban.shulman@gmail.com

Tabla de contenido

Problemática	2
Objetivo	2
Situación problemática	2
Modelo de negocio	2
Qué se está haciendo.....	2
Qué se entrega	3
Beneficios y ahorro para la empresa.....	3
Diagrama E-R	4
Listado de tablas	5
Listado vistas	9
Listado de procedimientos.....	10
Procedimientos de Ubicaciones y Productos	10
Procedimientos de Modelos y Vinculación	11
Procedimientos de Pedidos	11
Procedimientos de Droids.....	12
Listado de triggers	13
Listado de functions	14

Problemática

Una fábrica quiere hacer su sector de envíos de piezas de repuesto, pero con la particularidad de que los encargados de reponer el stock y cumplir con los pedidos, son drones experimentales y se precisa un control de todas las partes para verificar su funcionamiento, su mantenimiento y brindar información, tanto en los pedidos que se hace como en las posibles incidencias con los drones.

Para eso se requiere una base de datos con la cual se pueda obtener información de los pedidos que se realizan, su estado, el estado de los drones e informes de posibles incidencias.

La información obtenida servirá para descubrir la tendencia a incidentes relacionados a piezas, modelos y otros factores.

Para asegurar un control imparcial y completo del stock, se contrato a otra empresa la cual realiza auditorias de stock. Se le proporcionaran los métodos necesarios para controlar el stock que se declara en la base de datos.

Objetivo

La base de datos propuesta tiene dos objetivos:

1. Averiguar cuales son las fallas mas recurrentes en los productos por medio de las tendencias a la hora de pedir repuestos.
2. Controlar y analizar el desempeño de los drones experimentales.

Situación problemática

El sector de despacho de repuestos (tanto envíos como recepción) se reemplazo con drones, los cuales se les comunica las ordenes de pedidos y estos las cumplen. Se quiere analizar que modelos de drones y repuestos son aptos para el control del stock, a la vez que se quiere ver la tendencia en los pedidos de repuestos a fin de averiguar las fallas más recurrentes en los productos y proporcionar una solución.

Modelo de negocio

Qué se está haciendo

a. Modelado de la base de datos

Se diseñó una estructura completa que permite almacenar y relacionar información de múltiples áreas:

- Drones: registro de cada unidad, su estado operativo y modelo.
- Pedidos: solicitudes de piezas o movimientos operativos vinculados a los droides.
- Reparaciones: seguimiento de incidentes y partes utilizadas para mantenimiento.
- Personas y direcciones: registro de operarios, transportistas o clientes asociados a cada pedido.

- Auditorías y logs: control y trazabilidad de todos los movimientos y modificaciones.
- Historial de droides: trazabilidad de cada unidad, sus fallas, reparaciones y cambios de estado.
- Archivos asociados: generación automática de documentos con datos completos en formato JSON/PDF.

b. Automatización mediante procedimientos y triggers

Se implementaron procedimientos almacenados (Stored Procedures) y triggers para:

- Registrar pedidos, personas y direcciones de manera automática.
- Actualizar el stock de piezas en tiempo real.
- Registrar incidentes y reparaciones sin intervención manual.
- Crear registros de auditoría para mantener trazabilidad.
- Generar archivos automáticos con toda la información relacionada a un pedido o reparación.

c. Interoperabilidad

La base está diseñada para integrarse fácilmente con:

- Aplicaciones web o móviles.
- Paneles administrativos o dashboards.
- APIs RESTful.

Qué se entrega

El proyecto entrega un sistema base con:

1. Base de datos estructurada y documentada
Listo para implementarse en entornos de producción o desarrollo.
2. Procedimientos y triggers automatizados
Que reducen la carga de tareas manuales y mejoran la consistencia de los datos.
3. Modelo escalable
Preparado para crecer con nuevos módulos como control de usuarios, roles, métricas o analítica.
4. Historial y auditoría integrados
Cada movimiento en el sistema queda registrado con hora, responsable y descripción.
5. Interfaz API y compatibilidad con aplicaciones
Facilitando su integración con sistemas internos o externos.

Beneficios y ahorro para la empresa

Optimización de procesos operativos

La automatización de tareas repetitivas —como registrar pedidos, actualizar estados de droides o generar informes— elimina errores humanos y acelera la gestión diaria.

Reducción de tiempos y costos administrativos

La información de todas las áreas (pedidos, personas, droides, piezas) se encuentra centralizada y sincronizada automáticamente, evitando duplicidades y búsquedas manuales.

Trazabilidad total

Cada evento queda registrado: qué droide se reparó, qué pieza se usó, quién intervino y cuándo. Esto permite auditorías internas y externas sin esfuerzo adicional.

Control de stock en tiempo real

Los triggers y procedimientos actualizan las cantidades de piezas en el momento exacto del pedido o reparación, evitando sobrestock o faltantes inesperados.

Toma de decisiones basada en datos

Los historiales, logs y auditorías permiten detectar patrones de fallas, optimizar rutas de mantenimiento y reducir tiempos de inactividad de los droides.

Escalabilidad y compatibilidad

El modelo se adapta fácilmente a nuevos requerimientos (más tipos de droides, centros logísticos, reportes, etc.) sin necesidad de rediseñar la base.

Diagrama E-R

El diagrama de base de datos para el sistema de gestión de pedidos y mantenimiento de drones se estructura de la siguiente manera:

- Tablas y sus campos:**
 - ubicacion:** id INT, X TINYINT, Y TINYINT, Z TINYINT, descripcion VARCHAR(100).
 - log_movimientos:** id INT, tabla_afectada VARCHAR(100), operation VARCHAR(50), description TEXT, fecha TIMESTAMP.
 - auditor:** id INT, nombre VARCHAR(100), apellido VARCHAR(100).
 - auditoria:** id INT, fecha_registro TIMESTAMP, detalles JSON, auditor_id INT.
 - modelo_droide:** id INT, nombre VARCHAR(100), tipo ENUM(...).
 - droide:** id INT, name VARCHAR(100), fecha_servicio TIMESTAMP, en_servicio TINYINT, modelo_id INT.
 - direccion:** id INT, calle VARCHAR(100), altura INT.
 - persona_pedido:** id INT, nombre VARCHAR(100), apellido VARCHAR(100), empresa VARCHAR(150), dni_cul VARCHAR(20), pedido_id INT, direccion_id INT, patente VARCHAR(20), fecha_registro TIMESTAMP.
 - productos:** id INT, name VARCHAR(100), qty INT, ubicacion_pieza INT.
 - pedido:** id INT, pieza INT, cantidad INT, droideAsignado INT, tipo_movimiento TINYINT, tiempo VARCHAR(45).
 - historial_droides:** id INT, droide_id INT, nombre_droide VARCHAR(100), fecha_evento TIMESTAMP, descripcion TEXT, pieza_usada_id INT, pieza_usada_nombre VARCHAR(100), en_servicio TINYINT(1).
 - piezas_reparacion_droide:** id INT, nombre_pieza VARCHAR(100), partida_mes TINYINT, partida_anio YEAR, cantidad INT.
 - reparacion_droide:** id INT, droide_id INT, pieza_id INT, fecha_reparacion TIMESTAMP, observation VARCHAR(255).
 - producto_modelo:** id INT, producto_id INT, modelo_id INT.
 - modelo:** id INT, nombre VARCHAR(100).
- Vistas y Procedimientos Almacenados:**
 - vw_reporte:** Procedimiento almacenado.
 - vw_stock:** Vista.
 - vw_archivos_pedidos:** Vista.
 - vw_piezas_modelos_droides_failure:** Vista.
 - vista_persona_pedido_detalle:** Vista.
- Relaciones:**
 - Se establecen relaciones de uno a muchos (1:N) y muchos a muchos (M:N) entre las tablas, indicando restricciones de integridad referencial.
 - Por ejemplo, **persona_pedido** es el dueño de **pedido** (1:N), **pedido** es el dueño de **historial_droides** (1:N), **historial_droides** es el dueño de **reparacion_droide** (1:N), **reparacion_droide** es el dueño de **piezas_reparacion_droide** (1:N).
 - Existen relaciones de muchos a muchos entre **productos** y **pedido**, y entre **modelo** y **producto_modelo**.

productos: Contiene los productos o piezas disponibles, su nombre, cantidad y la ubicación donde están guardados.

	productos			
	id	name	qty	ubicación_pieza
dato	INT AI PK	VARCHAR(100)	INT	INT
tabla referida				ubicación
dato referido				id

droide: Registra los droides, su nombre, fecha de servicio, estado (en servicio o no) y el modelo al que pertenecen.

	droide				
	id	name	fecha_servicio	en_servicio	modelo_id
dato	INT AI PK	VARCHAR(100)	TIMESTAMP	TINYINT	INT
tabla referida					modelo
dato referido					id

modelo_droide: Define los distintos modelos de droides, indicando su nombre y tipo (terrestre o aéreo).

	modelo_droide		
	id	nombre	tipo
dato	INT AI PK	VARCHAR(100)	enum('terrestre','aereo')
tabla referida			
dato referido			

pedido: Representa los pedidos de piezas, indicando qué producto se pidió, la cantidad, el droide asignado y el tipo de movimiento.

	pedido					
	id	pieza	cantidad	droideAsignado	tipo_movimiento	tiempo
dato	INT AI PK	INT	INT	INT	TINYINT	VARCHAR(45)
tabla referida				droide		
dato referido				id		

log_movimientos: Registra operaciones realizadas (insert, update, delete) sobre las tablas, con una descripción y fecha.

	log_movimientos				
	id	tabla_afectada	operación	descripcion	fecha
dato	INT AI PK	VARCHAR(100)	VARCHAR(50)	TEXT	TIMESTAMP
tabla referida					
dato referido					

modelo: Define los modelos asociados a los productos (no confundir con modelo_droide).

	modelo	
	id	nombre
dato	INT AI PK	VARCHAR(100)
tabla referida		
dato referido		

producto_modelo: Relaciona productos con modelos, permitiendo saber qué modelo usa cada pieza o producto.

	producto_modelo		
	id	producto_id	modelo_id
dato	INT AI PK	INT	INT
tabla referida		producto	modelo
dato referido		id	id

reparacion_droide: Registra las reparaciones hechas a los droides, qué pieza se usó y la fecha.

	reparacion_droide				
	id	droide_id	pieza_id	fecha_reparacion	observacion
dato	INT AI PK	INT	INT	TIMESTAMP	VARCHAR(255)
tabla referida		droide	pieza		
dato referido		id	id		

persona_pedido: Guarda los datos personales de quien realiza un pedido, incluyendo nombre, empresa y dirección.

	persona_pedido			
	id	nombre	apellido	empresa
dato	INT AI PK	VARCHAR(100)	VARCHAR(100)	VARCHAR(100)

tabla referida				
dato referido				

	persona_pedido				
	dni_cuil	pedido_id	direccion_id	patente	fecha_registro
dato	VARCHAR(20)	INT	INT	VARCHAR(20)	TIMESTAMP
tabla referida		pedido	direccion		
dato referido		id	id		

direccion: Contiene las direcciones (calle y altura) asociadas a las personas o pedidos.

	direccion		
	id	calle	altura
dato	INT AI PK	VARCHAR(100)	INT
tabla referida			
dato referido			

piezas_reparacion_droide: Lista las piezas disponibles para reparaciones de droides, con su nombre, partida y cantidad.

	piezas_reparacion_droide				
	id	nombre_pieza	partida_mes	partida_anio	cantidad
dato	INT AI PK	VARCHAR(100)	TINYINT	YEAR	INT
tabla referida					
dato referido					

historial_droides: Almacena los eventos importantes de cada droide (reparaciones, cambios de estado, piezas usadas, etc.).

	historial_droides			
	id	droide_id	nombre_droide	fecha_evento
dato	INT AI PK	INT	VARCHAR(100)	TIMESTAMP
tabla referida		droide		
dato referido		id		

	historial_droides			
	descripcion	pieza_usada_id	pieza_usada_nombre	en_servicio
dato	VARCHAR(100)	INT	VARCHAR(100)	TINYINT
tabla referida		piezas_reparacion_droide		
dato referido		id		

archivo_pedido: Registra los archivos vinculados a un pedido (por ejemplo, documentos o comprobantes).

	archivo_pedido				
	id	persona_pedido_id	nombre_archivo	datos_pedido	fecha_creacion
dato	INT AI PK	INT	VARCHAR(255)	JSON	TIMESTAMP
tabla referida		persona_pedido			
dato referido		id			

auditor: Contiene los datos de los auditores del sistema.

	auditor		
	id	nombre	apellido
dato	INT AI PK	VARCHAR(100)	VARCHAR(100)
tabla referida			
dato referido			

auditoria: Guarda los registros de auditoría con detalles en formato JSON y el auditor que los realizó.

	auditoria			
	id	fecha_registro	detalles	auditor_id
dato	INT AI PK	TIMESTAMP	JSON	INT
tabla referida				auditor
dato referido				id

Listado vistas

vw_reporte

Tablas afectadas: *log_movimientos*

Descripción: Esta vista muestra un reporte general de los movimientos registrados en el sistema, incluyendo operaciones como inserciones, actualizaciones o eliminaciones en cualquier tabla.

Utilidad: Sirve para auditoría y trazabilidad, ya que muestra el contenido completo de log_movimientos.

vw_archivos_pedidos

Tablas afectadas: *archivo_pedido*

Descripción: Genera una ruta completa de archivo para cada registro en archivo_pedido, combinando el nombre del archivo con un prefijo fijo (/archivos/pedidos/).

Utilidad: Permite consultar fácilmente las ubicaciones finales de los archivos PDF asociados a pedidos.

vw_stock

Tablas afectadas: *productos, ubicación, producto_modelo, modelo*

Descripción: Esta vista consolida toda la información del inventario o stock de productos, mostrando:

- Los datos del producto (nombre, cantidad)
- Su ubicación física (coordenadas x, y, z)
- El modelo al que pertenece

Utilidad: Es útil para control de inventario, trazabilidad de piezas y gestión de modelos.

vw_persona_pedido_detalle

Tablas afectadas: *persona_pedido, pedido, productos, producto_modelo, modelo*

Descripción: Devuelve un detalle completo de cada pedido, incluyendo:

- Datos de la persona (nombre, apellido, DNI/CUIL, empresa)
- Información del pedido (tipo, cantidad, fecha)
- Producto y modelo asociados

Utilidad: Sirve para consultar pedidos con todos los datos del cliente y del producto solicitado.

vw_piezas_modelos_droides_failure

Tablas afectadas: *reparacion_droide, piezas_reparacion_droide, droide, modelo_droide*

Descripción: Muestra un historial técnico de reparaciones de droides, vinculando:

- El droide reparado
- La pieza usada
- El modelo del droide

Utilidad: Permite analizar fallas, frecuencia de reparaciones y tipos de piezas utilizadas según modelo.

Listado de procedimientos

Procedimientos de Ubicaciones y Productos

pr_crear_ubicacion

Tablas: *ubicacion*

Datos requeridos: p_id INT, p_X INT, p_Y INT, p_Z INT, p_descripcion VARCHAR

Utilidad: Inserta una nueva ubicación en el sistema y genera un mensaje de confirmación.

pr_obtener_ubicaciones

Tablas: *ubicacion*

Datos requeridos: Ninguno

Utilidad: Devuelve la lista completa de ubicaciones ordenadas por ID.

pr_crear_producto

Tablas: *productos*

Datos requeridos: p_name VARCHAR, p_qty INT, p_ubicacion_id INT

Utilidad: Crea un nuevo producto vinculado a una ubicación y devuelve el ID generado.

pr_obtener_productos

Tablas: *productos, ubicacion*

Datos requeridos: Ninguno

Utilidad: Retorna todos los productos junto con su información de ubicación.

pr_editar_producto

Tablas: *productos*

Datos requeridos: p_id INT, p_name VARCHAR, p_qty INT, p_ubicacion_id INT

Utilidad: Actualiza los datos de un producto existente y registra un mensaje de confirmación.

Procedimientos de Modelos y Vinculación

pr_crear_modelo

Tablas: modelo

Datos requeridos: *p_nombre*

Utilidad: Crea un nuevo modelo si no existe previamente; si ya existe, lanza un error.

pr_vincular_producto_modelo

Tablas: *producto_modelo*

Datos requeridos: p_producto_id INT, p_modelo_id INT

Utilidad: Relaciona un producto con un modelo; evita duplicaciones mediante validación previa.

Procedimientos de Pedidos

pr_crear_pedido_restar_sumar

Tablas: *pedido, productos, droide*

Datos requeridos: p_pieza INT, p_cantidad INT, p_tipo_movimiento TINYINT

Utilidad:

- Crea un nuevo pedido asignando un droide disponible aleatoriamente.
- Ajusta el stock de productos (suma o resta).
- Valida disponibilidad y stock antes de ejecutar.

pr_deshacer_pedido_autoasignado

Tablas: *pedido, productos*

Datos requeridos: p_pedido_id INT

Utilidad: Revierte un pedido existente, restaurando el stock afectado y eliminando el registro del pedido.

pr_gestionar_direccion

Tablas: *direccion*

Datos requeridos: p_calle VARCHAR, p_altura INT

Utilidad: Inserta una nueva dirección o actualiza una existente, según si el ID fue provisto o no.

pr_registrar_persona_pedido

Tablas: *persona_pedido*

Datos requeridos: p_nombre VARCHAR, p_apellido VARCHAR, p_empresa VARCHAR, p_dni_cuil VARCHAR, p_pedido_id INT, p_patente VARCHAR, p_direccion_id INT

Utilidad: Crea un registro de persona asociada a un pedido existente, validando que el pedido exista.

Procedimientos de Droids

pr_agregar_droide

Tablas: *droide*

Datos requeridos: p_name VARCHAR, p_en_servicio TINYINT, p_modelo_id INT

Utilidad: Inserta un nuevo droide con su estado y modelo correspondiente.

pr_agregar_modelo

Tablas: *modelo_droide*

Datos requeridos: p_nombre_modelo VARCHAR, p_tipo ENUM('terrestre', 'aereo')

Utilidad: Crea un nuevo modelo de droide (terrestre o aéreo).

pr_nueva_pieza

Tablas: *piezas_reparacion_droide*

Datos requeridos: nombre_pieza VARCHAR, partida_mes TINYINT, partida_anio YEAR, cantidad INT

Utilidad: Inserta una nueva pieza de reparación con información de lote y cantidad.

pr_modificar_droide

Tablas: *droide*

Datos requeridos: p_id INT, p_en_servicio TINYINT

Utilidad: Cambia el estado operativo de un droide (activo o fuera de servicio).

pr_agregar_reparacion_droide

Tablas: *reparacion_droide*

Datos requeridos: p_droide_id INT, p_pieza_id INT, p_observacion VARCHAR

Utilidad: Registra una reparación realizada sobre un droide y la pieza usada.

Procedimientos de Auditoría

pr_agregar_auditor

Tablas: *auditor*

Datos requeridos: p_nombre VARCHAR, p_apellido VARCHAR

Utilidad: Inserta un nuevo auditor en el sistema.

pr_registrar_auditoria

Tablas: *auditoria*

Datos requeridos: p_auditor_id INT, p_detalle JSON

Utilidad: Registra una nueva auditoría con los detalles (en formato JSON) asociados a un auditor existente.

Listado de triggers

trg_droide_insert

Tabla: *droide*

Tabla de destino: *log_movimientos*

Evento: AFTER INSERT

Datos usados: NEW.id, NEW.name, NEW.en_servicio

Utilidad: Registra en log_movimientos cada nuevo droide agregado al sistema, dejando una traza del alta.

trg_pedido_stock_update

Tabla: *pedido*

Tabla de destino: *log_movimientos*

Evento: AFTER INSERT

Datos usados: NEW.pieza, NEW.cantidad, NEW.tipo_movimiento, NEW.id

Utilidad: Actualiza el stock de productos (suma o resta cantidad según el tipo de movimiento) y registra la operación en log_movimientos.

trg_productos_ingresados

Tabla: *productos*

Tabla de destino: *log_movimientos*

Evento: AFTER INSERT

Datos usados: NEW.id, NEW.name, NEW.qty

Utilidad: Inserta en log_movimientos un registro que documenta la entrada de un nuevo producto al sistema.

trg_log_cambio_en_servicio

Tabla: *droide*

Tabla de destino: *log_movimientos*

Evento: AFTER UPDATE

Datos usados: OLD.en_servicio, NEW.en_servicio, NEW.id

Utilidad: Detecta cambios en el estado de servicio del droide y los registra en log_movimientos para auditoría.

trg_droide_estado_update

Tabla: *droide*

Tabla de destino: *historial_droide*

Evento: AFTER UPDATE

Datos usados: OLD.en_servicio, NEW.en_servicio, NEW.id, NEW.name

Utilidad: Registra en historial_droides los cambios de estado (en servicio / fuera de servicio) de cada droide.

trg_reparacion_droide_insert

Tabla: *reparacion_droide*

Tabla de destino: *historial_droide*

Evento: AFTER INSERT

Datos usados: NEW.droide_id, NEW.pieza_id, NEW.observacion, más los nombres del droide y la pieza obtenidos con consultas a droide y piezas_reparacion_droide.

Utilidad: Agrega un registro en historial_droides con el detalle de la reparación y la pieza usada.

trg_pedido_insert

Tabla: *pedido*

Tabla de destino: *historial_droide*

Evento: AFTER INSERT

Datos usados: NEW.droideAsignado, NEW.pieza, NEW.tipo_movimiento, NEW.cantidad

Utilidad: Si el pedido está asociado a un droide, agrega una entrada al historial_droides indicando qué pieza se pidió y con qué cantidad.

trg_persona_pedido_after_insert

Tabla: *persona_pedido*

Tabla de destino: *log_movimientos*

Evento: AFTER INSERT

Datos usados: Datos de la persona (nombre, apellido, empresa, etc.), dirección (calle, altura) y pedido (pieza, cantidad, droideAsignado, etc.).

Utilidad: Crea automáticamente un registro en archivo_pedido con un nombre de archivo generado y un JSON completo del pedido y los datos personales, automatizando la generación de archivos vinculados.

trg_auditoria_after_insert

Tabla: *auditoria*

Tabla de destino: *log_movimientos*

Evento: AFTER INSERT

Datos usados: NEW.id, NEW.auditor_id, NEW.fecha_registro, NEW.detalles

Utilidad: Cada vez que se inserta una nueva auditoría, guarda un registro en log_movimientos para mantener trazabilidad del control interno del sistema.

Listado de functions

f_mensaje_personalizado

Datos: p_tipo VARCHAR (20), p_texto VARCHAR (255)

Utilidad: Retorna mensajes personalizados para adaptarlos a cualquier API.

f_verificar_stock

Datos: p_producto_id INT, p_cantidad INT

Utilidad: Evita que los pedidos de repuestos no se realicen si se pide más del stock existentes.

Herramientas utilizadas

MySQL Workbench.

Office (Word, Excel).