UNIVERSIDAD DE SANTIAGO DE CHILE
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA INFORMÁTICA

TALLER DE BASE DE DATOS DIURNO 2-2025

Enunciado 1

Ayudante: Pablo Macuada **Fernando Solis**	**Profesor:** Matías Calderón

Enunciado Laboratorio 1

Entrega: 20 octubre de 2025

Proyecto:

Sistema de Gestión de Flotas de Drones

Objetivo: Desarrollar una plataforma integral para una empresa de logística o inspección que utilice drones. El sistema permitirá monitorear en tiempo real la ubicación, el estado y las tareas de cada dron, así como planificar nuevas misiones, sus rutas y generar reportes de su desempeño.

Tecnologías y Herramientas Requeridas

- **Base de Datos:** PostgreSQL
- **Backend:** Spring Boot con Java
- **Frontend:** Vue.js 3
- **Comunicación:** Axios para las llamadas HTTP
- **Seguridad:** JSON Web Tokens (JWT) para la autenticación y autorización
- **Control de Versiones:** Git y un repositorio en GitHub

Requisitos Específicos

1. Requisitos de la Base de Datos (PostgreSQL)

Deberás diseñar un **esquema de base de datos** normalizado que incluya, al menos, las siguientes tablas:

- **usuarios**: Información de los operadores y administradores del sistema (nombre, email, contraseña hasheada).
- **drones**: Información de cada dron (ID único, modelo, capacidad de carga, autonomía de la batería, estado: 'Disponible', 'En Vuelo', 'En Mantenimiento').
- **misiones**: Detalles de cada misión asignada a un dron (fecha, hora de inicio/fin, tipo de misión, estado: 'Pendiente', 'En Progreso', 'Completada' y su ruta).
- **registro_vuelo**: Tabla que registra la telemetría del dron durante una misión (ID de vuelo, timestamp, coordenadas de ubicación, nivel de batería, etc.).

No se permite el uso de JPA/Hibernate. La comunicación entre la aplicación y la base de datos debe ser exclusivamente a través de **sentencias SQL nativas**.

Deberás implementar los siguientes elementos en la base de datos:

- **Triggers:** Para automatizar procesos como la actualización del estado de un dron.
- **Procedimientos Almacenados:** Para encapsular la lógica de negocio compleja, como la asignación de drones a misiones o la validación de datos.
- **Vistas Materializadas:** Para pre-calcular resúmenes de datos que se consultan con frecuencia, mejorando el rendimiento de la aplicación.
- **Índices:** Para optimizar las consultas más comunes, especialmente las búsquedas por ubicación o estado.

2. Requisitos del Backend (Spring Boot)

- Crear una **API RESTful** que exponga los endpoints necesarios para la gestión de usuarios, drones y misiones.
- Implementar un sistema de autenticación de usuarios utilizando **JWT**.

- Desarrollar un endpoint de **login** que devuelva un token válido para las subsiguientes peticiones.
- Proteger las rutas de la API, permitiendo el acceso solo a usuarios autenticados.
- Conectar a la base de datos PostgreSQL y ejecutar consultas utilizando sentencias SQL puras.

3. Requisitos del Frontend (Vue.js)

- Crear una **interfaz de usuario** intuitiva y atractiva.
- Implementar una página de **login** para que los usuarios puedan autenticarse.
- Una vez autenticados, los usuarios deben poder ver un **mapa interactivo** con la ubicación en tiempo real de los drones y la ruta restante de cada uno. (Para esta entrega solo es necesario tener en cuenta en el diseño el espacio adecuado para el mapa)
- La interfaz debe permitir a los usuarios ver el estado de cada dron y las misiones asignadas.
- Utilizar **Axios** para todas las peticiones HTTP al backend.

Funcionalidades Clave del Sistema

- **Login de Usuario:** Autenticación de operadores y administradores a través de un token JWT.
- **Monitoreo en Tiempo Real:** Visualización en un mapa de la ubicación, batería y estado actual de la flota de drones.
- **Gestión de Misiones:** Posibilidad de crear, asignar y monitorear misiones.
- **Optimización de Rutas:** La API debe tener un endpoint que, a partir de una lista de misiones, genere una ruta óptima (considerando la distancia, el tiempo y el tipo de misión) para las tareas asignadas para uno o más drones.

- **Telemetría de Vuelo:** La API debe tener un endpoint para recibir y registrar los datos de telemetría de los drones.
- **Reportes de Desempeño:** Los administradores deben poder generar reportes sobre el tiempo de vuelo, misiones completadas y fallos de la flota.

Documentación y Entrega

Todo el proyecto debe ser subido a un repositorio de **GitHub**. La entrega debe incluir:

- **Documentación de la Base de Datos:** Un documento que describa el esquema, las relaciones entre tablas y el propósito de cada trigger, procedimiento almacenado, vista materializada e índice.
- **Script de Creación y Carga de Datos:** Un archivo .sql que permita recrear la base de datos completa, incluyendo las tablas, índices, triggers y un conjunto de datos de prueba para realizar las pruebas.
- **Código Fuente:** El código completo del backend (Spring Boot) y el frontend (Vue.js), subido al repositorio de GitHub.
- **README.md:** Un archivo README.md en el repositorio principal que contenga instrucciones claras sobre cómo clonar, configurar y ejecutar la aplicación.

Consultas SQL a desarrollar

1. **Análisis de Duración de Vuelo:** Escribe una consulta SQL que calcule el tiempo total de vuelo de cada dron, por modelo, en el último mes. La consulta debe mostrar el modelo del dron y el tiempo de vuelo total en horas, ordenado de mayor a menor.
2. **Identificación de Drones con Fallos Recurrentes:** Utilizando una subconsulta o una CTE (Common Table Expression), identifica los 5 drones que han tenido el mayor número de misiones con estado `'Fallida'`. Muestra el ID del dron y el número total de misiones fallidas.
3. **Comparación de Desempeño por Tipo de Misión:** Crea una consulta que compare la cantidad de misiones completadas por tipo (por ejemplo, 'Entrega', 'Inspección', 'Vigilancia') para los dos modelos de drones más

usados. Muestra el tipo de misión, la cantidad de misiones para el modelo A y la cantidad para el modelo B.

4. **Detección de Patrones de Consumo de Batería:** Escribe una consulta que identifique las 3 misiones más largas que, sin embargo, consumieron menos batería que el 20% de las misiones más cortas. Muestra el ID de la misión, su duración en minutos y el consumo total de batería.
5. **Análisis de Desempeño Mensual:** Utilizando funciones de ventana, calcula el promedio de misiones completadas por semana para cada mes en el último año. La consulta debe mostrar el mes y el promedio semanal. Además, debe incluir una columna que muestre la diferencia del promedio del mes actual con respecto al mes anterior.
6. **Simulación de Asignación de Misiones:** Escribe un procedimiento almacenado llamado `asignar_mision_a_dron` que reciba un ID de misión y un ID de dron. El procedimiento debe validar que el dron esté disponible (`estado` = 'Disponible') y que la misión no esté ya asignada. Si la validación es exitosa, debe actualizar el estado de la misión a 'En Progreso' y el estado del dron a 'En Vuelo'.
7. **Actualización Masiva de Estado de Mantenimiento:** Implementa un procedimiento almacenado que, dado un modelo de dron, actualice el estado de todos los drones de ese modelo a 'En Mantenimiento' si su tiempo de vuelo total supera un umbral de 100 horas.
8. **Listado de Drones Inactivos:** Crea una consulta que muestre todos los drones que no han sido asignados a una misión en los últimos 30 días. La consulta debe incluir el ID del dron, su modelo, su estado actual y la fecha de su última misión.
9. **Análisis Geográfico de Puntos de Interés:** Utilizando las coordenadas de vuelo, escribe una consulta para encontrar los 5 drones que han sobrevolado más cerca de un punto de interés específico (por ejemplo, una zona de entrega) en el último mes. La consulta debe mostrar el ID del dron y la distancia mínima en metros al punto de interés.
10. **Resumen de Misiones por Tipo:** Crea una vista materializada llamada `resumen_misiones_completadas` que muestre la cantidad total y el tiempo de vuelo promedio de las misiones completadas por tipo de misión. Esta vista debe ser refrescada concurrenemente y debe poder ser consultada rápidamente por la aplicación para generar informes.