

**TALLER DE BASE DE DATOS DIURNO 2-2025**  
**Enunciado 2**

**Ayudante: Pablo Macuada**  
**Fernando Solis**

**Profesor: Matías Calderón**

## Enunciado Laboratorio 2

Entrega: 19 enero de 2026

### Proyecto:

#### Plataforma de Urbanismo y Planificación de Ciudades

En el presente laboratorio se trabajará con herramientas de bases de datos espaciales aplicadas a un problema real de gestión logística. A través de este ejercicio, se busca que los estudiantes integren conceptos de modelado de datos, consultas espaciales y desarrollo de aplicaciones.

**Objetivo: Desarrollar una herramienta digital para planificadores urbanos que integre datos demográficos y espaciales para visualizar el crecimiento de la ciudad mediante sistemas de información geográfica (SIG).**

#### Tecnologías y Herramientas Requeridas

- **Base de Datos:** PostgreSQL con extensión **PostGIS**.
- **Backend:** Spring Boot con Java.
- **Frontend:** Vue.js 3 con librerías de mapas (Leaflet o OpenLayers).
- **Comunicación:** Axios.
- **Seguridad:** JWT.

## Requisitos Específicos

### 1. Requisitos de la Base de Datos (PostgreSQL + PostGIS) Diseñar un esquema que soporte datos espaciales:

- **zonas\_urbanas:** Delimitación geográfica (ID, nombre, **geometria** de tipo **GEOMETRY(POLYGON, 4326)**).
- **puntos\_interes:** Ubicaciones clave (ID, tipo, **geometria** de tipo **GEOMETRY(POINT, 4326)**).
- **proyectos\_urbanos:** Área del proyecto (ID, estado, **geometria** de tipo **GEOMETRY** para definir el polígono de obra).
- **datos\_demograficos:** Estadísticas asociadas a las zonas.
- **Elementos:**
  - **Índices Espaciales (GIST):** Obligatorios sobre todas las columnas de geometría.
  - **Triggers:** Validación topológica (ej. evitar que un edificio se inserte fuera de una zona urbana válida usando **ST\_Contains**).

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

### Consultas SQL a desarrollar (Enfoque Espacial)

1. **Cálculo de Densidad Real:** Calcular la densidad (población / área en km<sup>2</sup>) utilizando **ST\_Area(geometria::geography)** para obtener la superficie real de la zona urbana.
2. **Análisis de Proximidad:** Encontrar escuelas (**POINT**) que se encuentren a menos de 500 metros de un proyecto en curso usando **ST\_DWithin** o **ST\_Distance**.
3. **Superposición de Proyectos:** Identificar pares de proyectos que se superpongan geográficamente (**ST\_Intersects**) y calcular el área de conflicto en metros cuadrados (**ST\_Intersection**).
4. **Cobertura de Servicios:** Calcular qué porcentaje del área de una zona urbana está cubierta por el radio de servicio (buffer de 1km) de los hospitales existentes usando **ST\_Buffer** y **ST\_Intersection**.

## 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 conteniendo lo previamente realizado incluyendo para esta entrega las modificaciones para adaptar al sistema con la base de datos georeferenciada.
- **Script de Creación y Carga de Datos:** Un archivo **.sql** que permita recrear la base de datos completa, incluyendo todo lo previo más las adaptaciones de una base de datos georeferenciada.
- **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 previas, y además, se debe incluir en el caso de usar mapas vectoriales ya generados obtenidos de internet, las instrucciones del cómo configurar y cargar esta misma a la base de datos.