INFORME TÉCNICO & AUDITORÍA

Proyecto CORDAMI

Evaluación Estructural y Recomendación de Migración de Base de Datos

Enfoque: Base de Datos (MariaDB) **Realizado por:** Equipo de Tecnología de la FIIIDT **Fecha:** Agosto 2026

1. Resumen Ejecutivo

El presente documento detalla el análisis del esquema de base de datos del proyecto CORDAMI (actualmente implementado en MariaDB/MySQL). Tras la revisión del script DDL (`cordamim_form_carnet.sql`) y su interacción con el backend, se han identificado vicios de diseño críticos que comprometen la integridad, escalabilidad y rendimiento del sistema. Debido a la naturaleza censal y masiva de los datos que manejará el proyecto, la arquitectura actual es insostenible a mediano y largo plazo.

2. Hallazgos y Deficiencias Estructurales

2.1. Almacenamiento Ineficiente de Objetos JSON

SEVERIDAD: CRÍTICA

Ubicación: Tabla `censos_registrados`, columna `datos_json_snapshot`

Problema: Se está almacenando un objeto JSON masivo (que contiene toda la información vital del censo: geolocalización, actividades, infraestructura, etc.) utilizando el tipo de dato `longtext`.

Impacto: En MariaDB, este enfoque guarda la información como texto plano. Si en el futuro se requiere realizar analíticas, filtros o búsquedas basadas en campos internos de ese JSON (por ejemplo, extraer productores por tipo de riego o ubicación), las consultas requerirán un escaneo secuencial de texto (Full Table Scan). Esto resultará en un rendimiento inaceptablemente lento y en una imposibilidad de indexación eficiente.

2.2. Tipado de Datos Incorrecto y Permisivo

SEVERIDAD: ALTA

Ubicación: Múltiples tablas, ej. `actividades_dinamicas`

Problema: El esquema utiliza tipos de datos inadecuados para representar la información real. Por ejemplo:

- **Fechas como texto:** Los campos `fecha_inicio` y `fecha_fin` están definidos como `varchar(15)` (ej. `'01/05/2026'`).
- **Booleanos como texto:** El campo `infraestructura` está definido como `varchar(10)` con valor por defecto `'false'`.

Impacto: Impedirá realizar operaciones nativas de base de datos (como cálculos de rangos de fechas). Además, al permitir la inserción de texto arbitrario, la base de datos se llenará silenciosamente de datos inconsistentes y erróneos (basura), rompiendo la integridad del sistema.

2.3. Ausencia de Integridad Referencial (Llaves Foráneas)

SEVERIDAD: ALTA

Ubicación: Definición de tablas relacionales (`actividades_agricolas` , `actividades_dinamicas` , etc.)

Problema: Aunque las tablas referencian entidades externas (ej. `predio_id` , `productor_id`), en las sentencias `CREATE TABLE` no existen las restricciones de llaves foráneas (`CONSTRAINT FOREIGN KEY`).

Impacto: Un error en el aplicativo podría eliminar un Productor o un Predio, dejando sus actividades asociadas como registros "huérfanos" en la base de datos, corrompiendo la estructura relacional de los datos censales.

2.4. Incompatibilidad Transaccional con Modificaciones DDL

SEVERIDAD: CRÍTICA

Problema: Tal como se identificó en la auditoría del código fuente, el motor actual (MariaDB/MySQL) no soporta DDL Transaccional. Ejecutar comandos como `ALTER TABLE` o `CREATE TABLE` dentro de una transacción genera un *commit implícito*.

Impacto: Se anula la protección ACID. Si un proceso multifase falla, el comando `ROLLBACK` no podrá deshacer las operaciones previas, resultando en datos incompletos y corruptos.

3. Veredicto y Recomendación Estratégica

Recomendación Oficial: Migración a PostgreSQL

Considerando que el proyecto CORDAMI funcionará como un sistema censal gubernamental, la precisión, integridad y capacidad analítica de los datos son requisitos innegociables. El motor actual (MariaDB) presenta limitaciones estructurales que dificultan estos objetivos. Se recomienda encarecidamente migrar el motor de base de datos a **PostgreSQL**.

Argumentos Técnicos para la Migración:

- **Soporte Nativo JSONB:** PostgreSQL cuenta con el tipo de dato `JSONB` , el cual almacena documentos JSON en un formato binario indexado. Esto permitirá a CORDAMI realizar consultas analíticas profundas, filtros y extracciones directamente sobre los datos del censo almacenados en el *snapshot*, con un rendimiento excepcional.
- **DDL Transaccional:** PostgreSQL protege las transacciones incluso ante modificaciones estructurales (`ALTER TABLE`). Si ocurre un error, la base de datos completa un `ROLLBACK` perfecto, evitando los commits implícitos que actualmente afectan al proyecto.
- **Tipado Estricto:** PostgreSQL es riguroso con los tipos de datos. No permitirá insertar cadenas de texto en columnas booleanas o de fecha, forzando la validación desde el backend y garantizando la limpieza de la base de datos.
- **Soporte Geoespacial Futuro (PostGIS):** Dado que el censo recolecta datos de geolocalización (latitud, longitud), PostgreSQL permite la integración con PostGIS, el estándar de la industria para mapeo y sistemas de información geográfica, abriendo la puerta a futuras analíticas espaciales.

Plan de Acción Correctivo (Durante la Migración):

El esquema SQL no debe migrarse "tal cual". Durante la transición a PostgreSQL, el equipo de desarrollo deberá aplicar las siguientes correcciones obligatorias en la definición del esquema:

1. Cambiar la columna `datos_json_snapshot` de `longtext` a **`JSONB`**.
2. Refactorizar los tipos de datos: Modificar las columnas de fecha (ej. `fecha_inicio`) a tipo **`DATE`** y las columnas lógicas (ej. `infraestructura`) a tipo **`BOOLEAN`**.
3. Establecer explícitamente las **Llaves Foráneas** (ej. `FOREIGN KEY (predio_id) REFERENCES predios(id) ON DELETE CASCADE`) para asegurar la integridad referencial.