

# Instructivo Operativo

## Turnos de Inspectores y Módulo de Móviles

Este documento explica, de forma clara y ordenada, las tablas creadas en Supabase/Postgres para gestionar **turnos de inspectores** y **móviles**, junto con sus índices, triggers y funciones. Incluye guía de uso para Portal Web (Next.js) y App Móvil (React Native + Expo).

## Parte I — Turnos de Inspectores

### 1.1 Descripción general

El módulo de Turnos permite planificar jornadas, registrar entrada/salida y pausas de colación, y cerrar turnos automáticamente (autocierre nocturno).

### 1.2 Tablas del módulo

Tabla	¿Para qué sirve?	Relaciones clave (FK)
turno_tipo	Catálogo de tipos de turno (DIURNO, NOCTURNO, etc.).	—
evento_turno_tipo	Catálogo de eventos de marcación (ENTRADA, SALIDA, PAUSA_INICIO, PAUSA_FIN, AUTO_CIERRE...).	—
turnos_planificados	Agenda por inspector/día/horario. Base para notificaciones y control de asistencia.	inspectores, cuadrantes, turno_tipo, (creado_por)
turnos_excepciones	Cambios puntuales: anula o ajusta un turno planificado (licencia, capacitación, etc.).	turnos_planificados, (creado_por)
turnos	Instancia real de la jornada por día/inspector (estado, horas reales y colación).	inspectores, cuadrantes, (planificado_id), (creado_por)
eventos_turno	Registro de marcaciones y eventos con timestamp y actor (usuario).	turnos, evento_turno_tipo, (actor_user_id)

### 1.3 Índices recomendados

- **turnos\_planificados**: (inspector\_id, fecha), (cuadrante\_id, fecha) — listas por fecha y cuadrante.
- **turnos**: (inspector\_id, fecha), (estado, fecha), (cuadrante\_id, fecha) — tableros operativos del día.
- **eventos\_turno**: (turno\_id, ts desc), (tipo\_id), (actor\_user\_id) — recuperar últimas marcaciones rápidamente.

### 1.4 Funciones y Triggers

[fn\\_evento\\_id\(codigo\)](#)

Devuelve el id de un código en **evento\_turno\_tipo**. Facilita no consultar catálogos desde el cliente.

[trg\\_eventos\\_turno\\_apply](#) (AFTER INSERT en eventos\_turno)

Actualiza **turnos.estado**, **hora\_inicio\_real**, **hora\_fin\_real** y **minutos\_colacion** según el evento (ENTRADA/PAUSA/SALIDA/AUTO\_CIERRE).

`fn_turno_registrar_evento(turno_id, codigo, ...)`

Función SECURITY DEFINER que inserta el evento por código y permite exponer la acción como RPC en el BFF.

`fn_turnos_autocierre(hora_limite, max_batch)`

Cierra turnos abiertos (EN\_CURSO/EN\_PAUSA) al sobrepasar una hora límite o al día siguiente, registrando AUTO\_CIERRE.

## Parte II — Módulo de Móviles (Vehículos)

### 2.1 Descripción general

Gestiona móviles (patente, estado, kilometraje), sus usos por inspector y bitácoras opcionales de mantenimiento y combustible.

### 2.2 Tablas del módulo

Tabla	¿Para qué sirve?	Relaciones clave (FK)
movil_tipo	Catálogo de tipos de móvil (AUTO, MOTO, CAMIONETA...).	—
moviles	Maestro de vehículos (patente única). Mantiene <b>kilometraje_actual</b> y <b>estado</b> .	movil_tipo, (creado_por)
movil_usos	Asignación temporal de un móvil a un inspector (pueden ser varias en un día).	moviles, inspectores, (turnos)
movil_uso_kilometraje	Lecturas por uso: <b>INICIO</b> , <b>FIN</b> e <b>INTERMEDIA</b> . Permite calcular km recorridos.	movil_usos
movil_mantenciones) (opcional)	Bitácora de mantenciones (tipo, fecha, km, costo).	moviles
movil_cargas_combustible) (opcional)	Registro de combustible (litros, costo, km).	moviles

### 2.3 Índices recomendados

- **moviles**: (estado) para listar disponibles/asignados; (tipo\_id) para filtrar por tipo.
- **movil\_usos**: (movil\_id, inicio\_ts, fin\_ts) para detectar solapes y listar por rango; (inspector\_id, inicio\_ts); (turno\_id).
- **movil\_uso\_kilometraje**: (uso\_id, lectura\_ts) para recuperar lecturas en orden temporal.

### 2.4 Funciones y Triggers

#### trg\_usos\_no\_solapados (BEFORE INSERT/UPDATE en movil\_usos)

Previene intervalos de uso **solapados** para el mismo móvil. Si detecta cruce, lanza excepción y evita inconsistencias.

#### trg\_usos\_set\_estado\_movil (AFTER INSERT/UPDATE en movil\_usos)

Mantiene el **estado** del móvil: al crear uso → ASIGNADO; al cerrar y no quedar usos abiertos → DISPONIBLE.

#### trg\_kilometraje\_validar\_y\_sumar (AFTER INSERT/UPDATE en movil\_uso\_kilometraje)

En lectura **FIN**: valida existencia de **INICIO**, asegura  $FIN \geq INICIO$ , calcula **km\_recorridos** y actualiza **kilometraje\_actual**.

#### fn\_movil\_iniciar\_uso(...)

Crea un uso y registra la lectura de **INICIO**. Usa auth.uid() si no se entrega actor (SECURITY DEFINER).

`fn_movil_cerrar_uso(...)`

Registra la lectura **FIN**, cierra el uso si faltaba `fin_ts` y consolida los kilómetros recorridos.

## Parte III — Uso en Portal Web y App Móvil

### 3.1 Portal Web (Next.js)

- **Flota:** tabla/kanban por estado/tipo para asignar o poner en mantención.
- **Planificación de turnos:** calendario semanal para crear turnos\_planificados y notificar.
- **Supervisión del día:** tablero con turnos EN\_CURSO/EN\_PAUSA y usos de móviles activos.
- **Bitácoras:** mantenciones y combustible con filtros por fecha/móvil.

**BFF/RPC:** expón endpoints que llamen a `fn_movil_iniciar_uso`, `fn_movil_cerrar_uso` y `fn_turno_registrar_evento`. Aplica RLS según perfiles.

### 3.2 App Móvil (React Native + Expo)

- **Elegir móvil:** lista de móviles DISPONIBLES + el móvil en uso (si aplica).
- **Iniciar uso:** ingresar kilometraje inicio → RPC a `fn_movil_iniciar_uso`.
- **Cambiar móvil:** cerrar uso actual con kilometraje fin → iniciar uso con otro móvil.
- **Cerrar jornada:** cerrar uso pendiente y marcar SALIDA del turno (RPC a `fn_turno_registrar_evento`).

## Parte IV — Clasificación de Denuncias (Pública e Interna)

### 4.1 Objetivo

Separar lo que el ciudadano declara (**categoría pública**) de la **clasificación interna** que asigna el funcionario (Familia → Grupo → Subgrupo → Requerimiento). Esto permite lenguaje ciudadano claro y, a la vez, análisis operativo detallado.

### 4.2 Tablas creadas / modificadas

Tabla	¿Para qué sirve?	Relaciones clave (FK)
categorias_publicas	Catálogo <b>público</b> (8 categorías) para que el ciudadano elija en el formulario.	—
cat_familias	Nivel 1 de la taxonomía interna (operativa).	—
cat_grupos	Nivel 2 (dentro de una familia).	familia_id → cat_familias.id
cat_subgrupos	Nivel 3 (dentro de un grupo).	grupo_id → cat_grupos.id
cat_requerimientos	Nivel 4 (requerimiento específico). Puede llevar prioridad/SLA.	subgrupo_id → cat_subgrupos.id
cat_req_mapeo_publico	Asocia cada <b>requerimiento interno</b> a una <b>categoría pública</b> (1→1) para reportes.	requerimiento_id → cat_requerimientos.id; categoria_publica_id → categorias_publicas.id
denuncia_clasificaciones	Histórico de clasificaciones por denuncia; deja una <b>vigente=true</b> .	denuncia_id → denuncias.id; requerimiento_id → cat_requerimientos.id; clasificado_por → auth.users.id
denuncias (modificada)	Agrega <b>categoria_publica_id</b> (lo que selecciona el ciudadano).	categoria_publica_id → categorias_publicas.id

### 4.3 Índices clave

- **cat\_grupos**(familia\_id), **cat\_subgrupos**(grupo\_id), **cat\_requerimientos**(subgrupo\_id).
- **denuncia\_clasificaciones**(denuncia\_id, vigente) + índice único parcial para garantizar una sola vigente por denuncia.
- **cat\_req\_mapeo\_publico**(categoria\_publica\_id) para agregaciones rápidas por categoría ciudadana.

### 4.4 Flujo de uso (UI)

#### Formulario ciudadano

Carga **categorias\_publicas**; guarda **denuncias.categoria\_publica\_id**.

#### Portal funcionario

Selector jerárquico: **Familia** → **Grupo** → **Subgrupo** → **Requerimiento**. Inserta en **denuncia\_clasificaciones** con *vigente=true*. Un trigger apaga clasificaciones anteriores.

## 4.5 Categorías públicas (8)

- **Emergencias** — Accidentes, incendios, rescates y situaciones urgentes.
- **Violencia y agresiones** — Peleas, amenazas, lesiones, VIF/riñas.
- **Robos y daños** — Robos, hurtos, daños, vandalismo, receptación.
- **Drogas** — Consumo o venta de drogas, microtráfico.
- **Armas** — Porte, uso o tenencia de armas y municiones.
- **Incivilidades** — Ruidos, desórdenes, consumo en vía pública, mal uso de espacios.
- **Patrullaje municipal** — Presencia preventiva/operativa: rondas, puntos fijos, acompañamientos.
- **Otros** — Hechos no incluidos en las anteriores.

## 4.6 ¿Qué entra en “Patrullaje municipal”?

- Solicitudes de **ronda preventiva** por horario o zona.
- **Presencia disuasiva** en salidas de colegios, ferias, plazas.
- **Vigilancia por conductas sospechosas** sin delito activo.
- **Punto fijo temporal** para eventos comunitarios.
- **Acompañamiento/custodia preventiva** (retirar dinero, cierre de local).
- **Refuerzo de patrullaje** en áreas con denuncias reiteradas o apoyo a operativos municipales.

### ¿Qué no va en “Patrullaje municipal”?

- **Emergencias o delitos en curso:** usar “Emergencias”, “Violencia y agresiones” o “Robos y daños” según corresponda.
- **Drogas:** consumo/venta → “Drogas”.
- **Armas:** avistamiento/porte → “Armas”.
- **Incivilidades:** ruidos, desórdenes, mal estacionados, basura → “Incivilidades”.

## 4.7 Consultas útiles

### Clasificación vigente por denuncia (con categoría pública)

```
select d.id, cp.nombre as categoria_publica, f.nombre as familia, g.nombre as grupo, sg.nombre as subgrupo,
r.nombre as requerimiento from public.denuncias d left join public.categorias_publicas cp on cp.id =
d.categoria_publica_id left join public.denuncia_clasificaciones dc on dc.denuncia_id = d.id and dc.vigente =
true left join public.cat_requerimientos r on r.id = dc.requerimiento_id left join public.cat_subgrupos sg on sg.id =
r.subgrupo_id left join public.cat_grupos g on g.id = sg.grupo_id left join public.cat_familias f on f.id =
g.familia_id order by d.created_at desc;
```

### Requerimientos mapeados por categoría pública

```
select cp.nombre as categoria_publica, count(*) as requerimientos from public.cat_req_mapeo_publico m join
public.categorias_publicas cp on cp.id = m.categoria_publica_id group by cp.nombre order by min(cp.orden);
```

### Requerimientos sin mapeo (debería ser 0)

```
select r.id, r.nombre from public.cat_requerimientos r left join public.cat_req_mapeo_publico m on
m.requerimiento_id = r.id where m.requerimiento_id is null order by r.id;
```