Documentación de Base de Datos

Vallestelar / IoT Sentinel

Objetivo general del modelo

Esta base de datos está diseñada para operar un sistema IoT **multi-cliente (multi-tenant)** con un único PostgreSQL, soportando cuatro dominios funcionales:

- **Agua**
- **Riego**
- **Seguridad**
- **Energía**

El diseño separa claramente:

- Gestión SaaS (clientes, usuarios, permisos, terrenos)
- Inventario IoT (dispositivos, sensores, actuadores)
- Datos (mediciones agregadas, métricas diarias, eventos)
- Control (comandos, reglas, programaciones)

Principios clave del diseño:

- Escalabilidad sin duplicar infraestructura
- Auditabilidad completa
- Control del volumen de datos
- Evolución sin migraciones constantes (uso de JSON)

1. Núcleo SaaS (multi-tenant)

Tabla: `tenants`

Descripción
Representa al **cliente** (persona, familia o empresa) que contrata el servicio.

Campos

- `id` (UUID, PK): identificador único del cliente.
- `name`: nombre comercial o identificador humano.
- `rut` (nullable): identificador tributario.
- `plan`: plan contratado (`agua`, `terreno`, `total`, `enterprise`).
- `status`: estado operativo (`active`, `suspended`).
- `metadata` (JSON): información flexible adicional.
- `created_at`: fecha de creación.
- `created_by`: quién lo creó.

- `updated_at`: fecha de última actualización.
- `updated_by`: quién lo modificó.

Razón de ser
Permite operar como SaaS, aislar datos por cliente y habilitar facturación y planes.

Tabla: `sites`

Descripción
Representa un **terreno o instalación física** perteneciente a un tenant.

Campos

- `id` (UUID, PK)
- `tenant_id` (FK)
- `name`: nombre del sitio.
- `address_text` (nullable): dirección textual.
- `timezone`: zona horaria del sitio.
- `lat`, `lng` (nullable): coordenadas.
- `metadata` (JSON)
- `created_at`, `created_by`, `updated_at`, `updated_by`

Razón de ser
Un cliente puede gestionar múltiples terrenos de forma independiente.

Tabla: `users`

Descripción
Usuarios que acceden al sistema (propietarios, familiares, técnicos, soporte).

Campos

- `id` (UUID, PK)
- `email` (unique)
- `password_hash`
- `full_name` (nullable)
- `status` (`active`, `disabled`)
- `metadata` (JSON)
- `created_at`, `created_by`, `updated_at`, `updated_by`

Razón de ser
Gestión de identidad, autenticación y trazabilidad.

Tabla: `user_memberships`

Descripción
Relación entre usuarios y tenants con un rol asociado.

Campos

- `id` (UUID, PK)
- `user_id` (FK)
- `tenant_id` (FK)
- `role` (`owner`, `admin`, `member`, `installer`, `support`)
- `metadata` (JSON)
- `created_at`, `created_by`, `updated_at`, `updated_by`

Razón de ser
Permite que un usuario pertenezca a múltiples clientes con roles distintos.

---

2. Inventario IoT

Tabla: `devices`

Descripción
Dispositivo IoT físico (ESP32, gateway, Raspberry, etc.).

Campos

- `id` (UUID, PK)
- `tenant_id` (FK)
- `site_id` (FK)
- `name`
- `device_type`
- `serial` (unique)
- `fw_version` (nullable)
- `status` (`online`, `offline`)
- `last_seen_at` (nullable)
- `metadata` (JSON)
- `created_at`, `created_by`, `updated_at`, `updated_by`

Razón de ser
Agrupa sensores y actuadores bajo un mismo hardware controlador.

Tabla: `sensors`

Descripción
Sensor lógico que genera mediciones.

Campos

- `id` (UUID, PK)
- `tenant_id` (FK)
- `site_id` (FK)
- `device_id` (FK)

- `name`
- `sensor_type`
- `unit` (nullable)
- `calibration_json` (JSON)
- `is_enabled`
- `metadata` (JSON)
- `created_at`, `created_by`, `updated_at`, `updated_by`

Razón de ser
Separar cada tipo de medición como entidad independiente.

Tabla: `actuators`

Descripción
Elemento controlable (válvula, relé, bomba, portón, sirena).

Campos

- `id` (UUID, PK)
- `tenant_id` (FK)
- `site_id` (FK)
- `device_id` (FK)
- `name`
- `actuator_type`
- `channel` (nullable)
- `state` (JSON)
- `is_enabled`
- `metadata` (JSON)
- `created_at`, `created_by`, `updated_at`, `updated_by`

Razón de ser
Permite enviar comandos y auditar acciones sobre el entorno físico.

---

3. Dominio Agua

Tabla: `tanks`

Descripción
Depósito de agua físico.

Campos

- `id` (UUID, PK)
- `tenant_id` (FK)
- `site_id` (FK)
- `name`
- `capacity_liters`

- `shape`
- `height_cm` (nullable)
- `min_level_pct_alert`
- `critical_level_pct_alert`
- `sensor_level_id` (FK nullable)
- `sensor_pressure_id` (FK nullable)
- `metadata` (JSON)
- `created_at`, `created_by`, `updated_at`, `updated_by`

Razón de ser
Entidad de negocio clave para control de agua y alertas.

Tabla: `pumps`

Descripción
Bomba asociada al llenado o distribución de agua.

Campos

- `id` (UUID, PK)
- `tenant_id` (FK)
- `site_id` (FK)
- `name`
- `actuator_id` (FK nullable)
- `mode` (`manual`, `auto`)
- `metadata` (JSON)
- `created_at`, `created_by`, `updated_at`, `updated_by`

Razón de ser
Permite automatizar y proteger el sistema de llenado.

---

4. Dominio Riego

Tabla: `irrigation_zones`

Descripción
Zona de riego independiente.

Campos

- `id` (UUID, PK)
- `tenant_id` (FK)
- `site_id` (FK)
- `name`
- `actuator_id` (FK nullable)
- `sensor_moisture_id` (FK nullable)
- `sensor_flow_id` (FK nullable)

- `metadata` (JSON)
- `created_at`, `created_by`, `updated_at`, `updated_by`

Tabla: `irrigation_schedules`

Descripción
Programación de riego por zona.

Campos

- `id` (UUID, PK)
- `tenant_id` (FK)
- `site_id` (FK)
- `zone_id` (FK)
- `cron`
- `duration_seconds`
- `is_enabled`
- `metadata` (JSON)
- `created_at`, `created_by`, `updated_at`, `updated_by`

Tabla: `rules`

Descripción
Reglas automáticas del sistema.

Campos

- `id` (UUID, PK)
- `tenant_id` (FK)
- `site_id` (FK)
- `name`
- `area`
- `condition_json` (JSON)
- `action_json` (JSON)
- `is_enabled`
- `metadata` (JSON)
- `created_at`, `created_by`, `updated_at`, `updated_by`

---

5. Control y eventos

Tabla: `commands`

Descripción
Comandos enviados a actuadores.

Campos

- `id` (UUID, PK)
- `tenant_id` (FK)
- `site_id` (FK)
- `actuator_id` (FK)
- `command_type`
- `payload` (JSON)
- `status`
- `requested_by_id` (FK nullable)
- `sent_at` (nullable)
- `acked_at` (nullable)
- `metadata` (JSON)
- `created_at`, `created_by`, `updated_at`, `updated_by`

Tabla: `events`

Descripción
Eventos relevantes del sistema.

Campos

- `id` (UUID, PK)
- `tenant_id` (FK)
- `site_id` (FK)
- `event_type`
- `severity`
- `source_type` (nullable)
- `source_id` (nullable)
- `ts`
- `title`
- `description` (nullable)
- `ack_status`
- `ack_by_id` (nullable)
- `ack_at` (nullable)
- `meta` (JSON)
- `created_at`, `created_by`, `updated_at`, `updated_by`

---

6. Datos y métricas

Tabla: `sensor_readings_5m`

Descripción
Lecturas de sensores agregadas (ej. cada 5 minutos).

Campos

- `id` (UUID, PK)

- tenant_id (FK)
- site_id (FK)
- sensor_id (FK)
- ts
- value
- quality
- meta (JSON)
- created_at, created_by, updated_at, updated_by

Tabla: daily_metrics

Descripción
Métricas diarias calculadas.

Campos

- id (UUID, PK)
- tenant_id (FK)
- site_id (FK)
- metric_date
- metric_key
- value
- meta (JSON)
- created_at, created_by, updated_at, updated_by

---

7. Seguridad

Tabla: security_modes

Descripción
Estado de seguridad del sitio.

Campos

- id (UUID, PK)
- tenant_id (FK)
- site_id (FK)
- mode
- is_armed
- metadata (JSON)
- created_at, created_by, updated_at, updated_by

---

8. Energía

Tabla: `energy_systems`

Descripción
Sistema energético del sitio.

Campos

- `id` (UUID, PK)
- `tenant_id` (FK)
- `site_id` (FK)
- `name`
- `battery_capacity_ah` (nullable)
- `sensor_battery_voltage_id` (FK nullable)
- `sensor_solar_power_id` (FK nullable)
- `metadata` (JSON)
- `created_at`, `created_by`, `updated_at`, `updated_by`

Convenciones transversales

- **Campos de auditoría**: trazabilidad completa.
- **Campos JSON (`metadata`, `meta`)**: evolución sin romper el esquema.
- **Relación `tenant_id` en todas las tablas**: aislamiento lógico multi-tenant.

Resumen

Este modelo permite:

- Operar múltiples clientes con una sola infraestructura
- Escalar en número de sensores sin saturar la base de datos
- Automatizar decisiones mediante reglas
- Auditar y controlar cada acción del sistema IoT

Es una base sólida para un producto profesional, vendible y escalable.