TRABAJO FIN DE GRADO

UCAM
UNIVERSIDAD CATÓLICA
DE MURCIA

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería Informática

MODELO DE DATOS

Autor:

D. Marc Hernández Montesinos

Directora:

Dra. Dña. *Angélica Guzmán Ponce*

Murcia, Junio de 2026

TRABAJO FIN DE GRADO

UCAM
UNIVERSIDAD CATÓLICA
DE MURCIA

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería Informática

MODELO DE DATOS

Autor:

D. Marc Hernández Montesinos

Directora:

Dra. Dña. *Angélica Guzmán Ponce*

Murcia, Junio de 2026

https://tfg.marchernandez.es/videos/video-demostrativo.mp4

ÍNDICE

MODELO DE DATOS

Esquema completo de base de datos: DDL, índices, migraciones

Documento separado del Trabajo de Fin de Grado *"Diseño e implementación de un sistema integrado de PKI y SSO para organizaciones pequeñas"* de Marc Hernández Montesinos (UCAM, Grado en Ingeniería Informática).

Campo	Valor
Documento	Modelo de Datos
Versión	1.0
Fecha	1 junio 2026
URL pública	https://tfg.marchernandez.es/manuales/Manual_Modelo_Datos.pdf

Campo	Valor
TFG asociado	https://tfg.marchernandez.es

Este manual conserva la numeración interna original (sección E.x) por trazabilidad con las versiones previas del documento. Las referencias cruzadas que apuntan al cuerpo del TFG (capítulos 1-10, anexos A-D) se mantienen tal cual y son válidas frente al PDF principal del TFG.

Documenta el modelo de datos operativo real del sistema, separado en dos bases de datos lógicamente independientes - una para el SSO y otra para la PKI - y consolida el rastro de migraciones aplicadas a lo largo del desarrollo. Para cada tabla se sigue un formato fijo: finalidad, campos críticos, claves e índices, observaciones de seguridad/RGPD y un DDL resumido o completo según el tamaño.

E.1 Visión global

E.1.1 Dos bases de datos separadas, no una

El sistema utiliza dos bases de datos MariaDB 10.5 completamente separadas en la misma instancia, accedidas por *socket* UNIX local:

Base de datos	Subsistema	Responsabilidad
`marchernandezmo_sso_nuevo_2`	SSO	Identidades, sesiones, aplicaciones cliente OAuth 2.0/OIDC, MFA, auditoría de seguridad, *keypairs* RS256, certificados de confianza para mTLS
`marchernandezmo_pki_nuevo`	PKI	Autoridades certificadoras (CA), plantillas, solicitudes (CSR), certificados emitidos, listas de revocación (CRL), histórico OCSP, auditoría de operaciones PKI

Esta separación es una decisión de diseño deliberada, no accidental. Sus implicaciones se discuten en Sección 7.3.3 del cuerpo del TFG y se resumen aquí:

1. **Aislamiento de blast radius.** Un compromiso del SSO no expone directamente las CSR, claves públicas históricas ni el histórico de revocaciones de la PKI, y viceversa.

2. **Modelo de permisos diferenciado.** Cada subsistema usa un usuario MariaDB distinto con privilegios mínimos (`SELECT`, `INSERT`, `UPDATE`, `DELETE` sobre sólo sus tablas; `CREATE`/`ALTER` se reservan al usuario de despliegue, fuera de la cuenta runtime).

3. **Ciclos de vida independientes.** El esquema PKI puede modificarse sin invalidar sesiones activas del SSO, y los *backups* pueden segmentarse y restaurarse de forma independiente.

4. **No hay claves foráneas entre bases.** El acoplamiento se hace a nivel aplicativo: la columna `pki_users.sso_user_id` referencia un UUID del SSO (`sso_users.id`) sin restricción referencial; la integridad se gestiona en la capa PHP.

E.1.2 Charset y collation: una herencia honesta

Existe una discrepancia consciente entre los dos esquemas, documentada explícitamente porque conviene explicarla:

BD	Charset por defecto	Collation	Origen
SSO	`utf8mb4`	`utf8mb4_unicode_ci`	Diseño nuevo desde cero (2026-02), siguiendo buenas prácticas modernas (soporte completo de Unicode, emojis, 4 bytes/char)
PKI	`utf8mb3`	`utf8mb3_general_ci`	Heredado del esquema PKI inicial (2026-02-09) creado mediante phpMyAdmin, que aplicaba `utf8mb3` como charset por defecto

La PKI mantiene `utf8mb3` por inercia de despliegue: migrar la base de datos completa a `utf8mb4` exigiría regenerar índices `varchar(255)` que actualmente

ocupan ≤765 bytes y migrar el histórico de certificados (>1500 entradas en `crl_records`). En la práctica:

- Los certificados X.509 v3 sólo usan caracteres ASCII en el DN (excepto algunos campos con UTF-8 que se almacenan como secuencias `\xC3\xA8` legibles).
- Los nombres de organizaciones con caracteres no-ASCII (p. ej. *Real Acadèmia de Cultura Valenciana*) se almacenan correctamente porque PHP escribe UTF-8 y MariaDB lo acepta dentro del rango BMP que cubre `utf8mb3`.
- La migración a `utf8mb4` figura como deuda técnica conocida (véase Sección 9.3 del TFG).

E.1.3 Tipos de identificador

Subsistema	Tipo de ID en tablas principales	Justificación
SSO	`CHAR(36)` (UUID v4)	Identificadores no enumerables, generados con `random_bytes(16)` en PHP, evitan enumeración secuencial y permiten despliegue federado sin colisiones
PKI	`INT UNSIGNED AUTO_INCREMENT`	Identificadores cortos, compatibles con la operativa de OpenSSL (que también usa contadores enteros en `index.txt` y `serial`), más eficientes en índices

El "puente" entre ambos subsistemas se hace mediante el campo `pki_users.sso_user_id` (`VARCHAR(36)`), reparado en la **migración 008** (véase Sección E.5) después de detectar un bug crítico de conversión silenciosa UUID→INT.

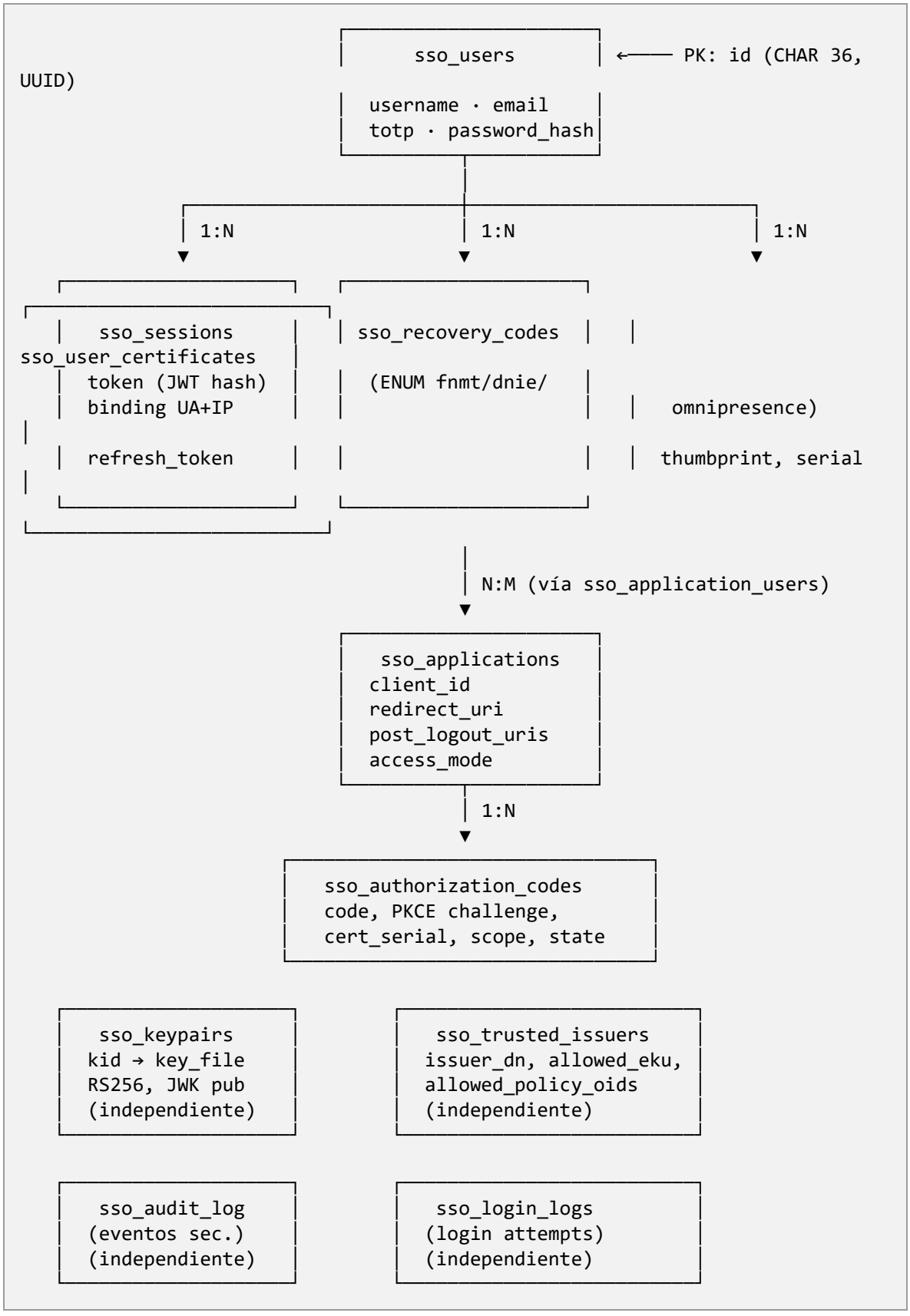
E.1.4 Resumen cuantitativo

Subsistema	Tablas	Tablas con FK	Tablas con datos sensibles	Tamaño aproximado en producción
SSO	13	6	5 (`sso_users`, `sso_sessions`, `sso_recovery_codes`, `sso_password_resets`, `sso_user_certificates`)	~310 KB con datos (mayoritariamente `sso_audit_log` y `sso_login_logs`)
PKI	11	4	3 (`pki_users`, `certificate_authorities`, `certificates`)	~5,4 MB con datos (mayoritariamente PEM de certificados y CRL históricas)

E.4 Modelo entidad-relación simplificado

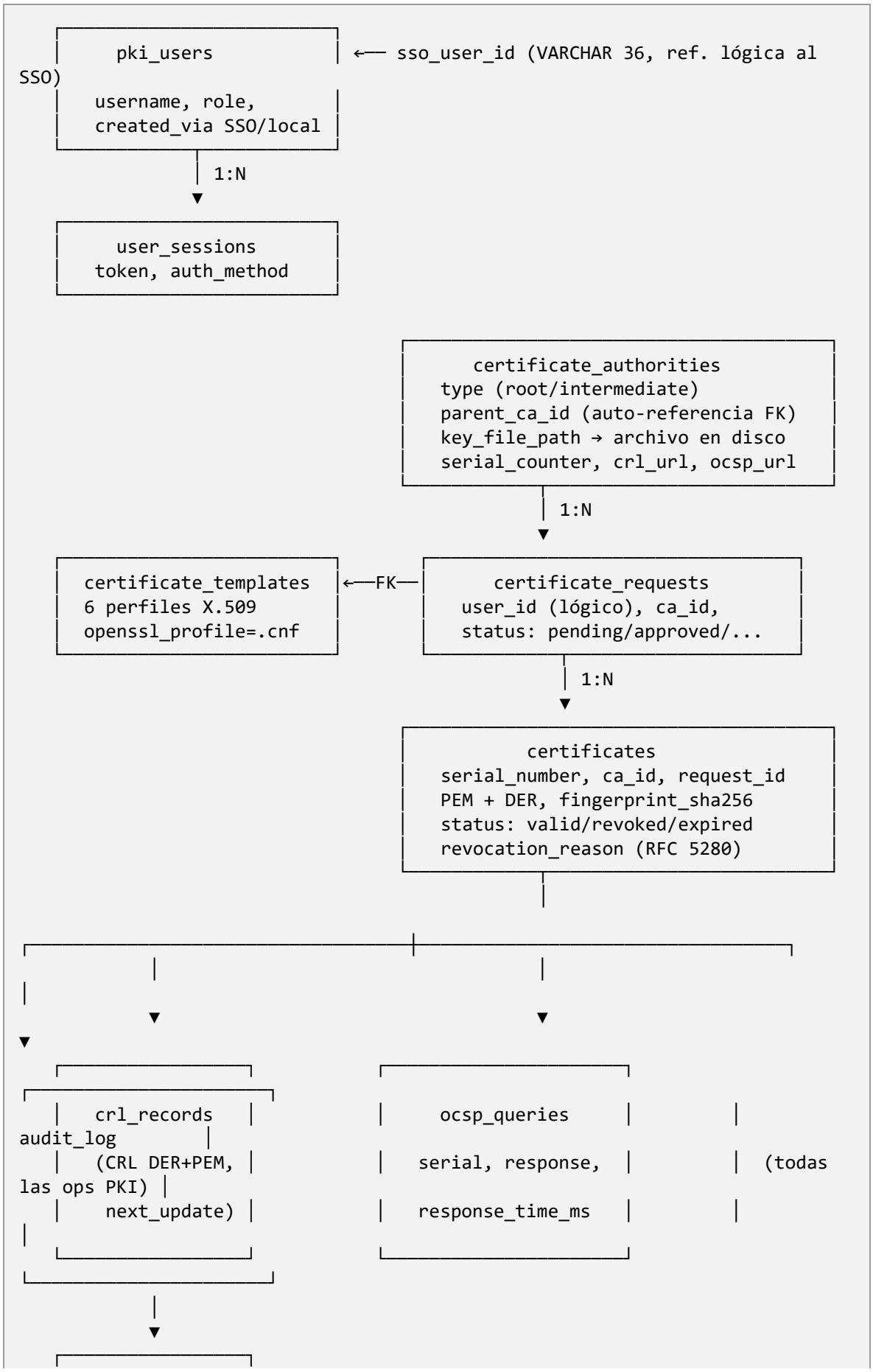
Se presenta el diagrama ER antes del DDL para que el lector tenga un mapa mental antes de leer las definiciones detalladas. Sólo se muestran las relaciones que existen físicamente (claves foráneas declaradas o relaciones lógicas activas en la aplicación). Se omiten tablas auxiliares (`sso_rate_limits`, `sso_password_resets`, etc.) por claridad.

E.4.1 Diagrama del subsistema SSO



E.4.2 Diagrama del subsistema PKI

Modelo de Datos



```
  │ crl_downloads  │
  │ (histórico)    │
  └────────────────┘

  ┌────────────────┐
  │   pki_config   │  (clave-valor de configuración runtime)
  └────────────────┘
```

Convenciones del diagrama. `FK` indica clave foránea declarada en MariaDB. Las relaciones marcadas como "ref. lógica" no tienen restricción de integridad referencial pero son comprobadas por la capa de aplicación en cada operación crítica.

E.2 Esquema del subsistema SSO (`marchernandezmo_sso_nuevo_2`)

E.2.1 `sso_users` - núcleo de identidades

Finalidad. Tabla maestra de usuarios del SSO. Almacena credenciales primarias (contraseña Argon2id), estado de TOTP, identificadores de los métodos de autenticación alternativa (certificado, *smartcard*), metadatos de bloqueo y *audit fields*.

Campos críticos.

- `id CHAR(36)` - UUID v4 generado por la aplicación; es la clave primaria a la que apuntan todas las tablas relacionadas.
- `password_hash VARCHAR(255)` - formato `$argon2id$v=19$m=65536,t=4,p=1$...` (NIST SP 800-63B nivel AAL2 con MFA activo).
- `totp_secret VARCHAR(255)` - soporta dos formatos legibles desde PHP: base32 en claro (legado) o `enc:<base64>` cifrado AES-256-CBC con clave en `.env` (formato actual; véase Sección 7.8, capa 4 - Cifrado en reposo).
- `certificate_thumbprint VARCHAR(128)` - huella SHA-256 hex del certificado vinculado al usuario; los detalles completos viven en `sso_user_certificates` (1:N).
- `failed_login_attempts INT` + `locked_until DATETIME` - implementan el *backoff* y el bloqueo automático tras 5 intentos fallidos (`sso_rate_limits` cubre la otra capa, *rate limiting* por IP).

- `last_login_at`, `last_login_ip` - usados por el panel admin y por `sso_audit_log`.

Claves e índices.

- `PRIMARY KEY (id)`.
- `UNIQUE KEY (username)`, `UNIQUE KEY (email)` - exclusión a nivel BD (la capa aplicativa también valida).
- `KEY (certificate_dn(191))`, `KEY (certificate_thumbprint)`, `KEY (smartcard_thumbprint)` - soportan *lookup* directo por huella en autenticación por certificado/smartcard sin necesidad de JOIN.

Observaciones de seguridad/RGPD.

- **PII almacenada:** `email`, `display_name`, `phone_prefix`, `phone_number`, `last_login_ip`. Base jurídica: ejecución de contrato (acceso al servicio) y consentimiento (datos opcionales como teléfono). Política de retención: hasta baja del usuario + 1 año por obligaciones contables del prestador del servicio.
- **Secretos:** `password_hash` (irrecuperable, *one-way*), `totp_secret` (cifrado en reposo en despliegues nuevos; en migración progresiva en cuentas legadas).
- **Right to be forgotten:** `DELETE` cascada limpia `sso_sessions`, `sso_recovery_codes`, `sso_user_certificates`, `sso_password_resets`, `sso_application_users` y `sso_authorization_codes` por las FK `ON DELETE CASCADE`. El `sso_audit_log` conserva las entradas históricas, pero pseudonimizadas (sólo `user_id` UUID, sin nombre).

DDL completo.

```sql
CREATE TABLE `sso_users` (
  `id` CHAR(36) NOT NULL,
  `username` VARCHAR(100) NOT NULL,
  `email` VARCHAR(255) NOT NULL,
  `password_hash` VARCHAR(255) NOT NULL,
  `display_name` VARCHAR(200) NOT NULL DEFAULT '',
  `phone_prefix` VARCHAR(10) NOT NULL DEFAULT '+34',
  `phone_number` VARCHAR(30) NOT NULL DEFAULT '',
  `role` ENUM('user','admin') NOT NULL DEFAULT 'user',
  `totp_secret` VARCHAR(255) DEFAULT NULL
      COMMENT 'Secreto TOTP base32 o cifrado (prefijo enc:)',
  `totp_enabled` TINYINT(1) NOT NULL DEFAULT 0,
  `totp_verified_at` DATETIME DEFAULT NULL,
  `certificate_dn` VARCHAR(500) DEFAULT NULL,
  `certificate_issuer` VARCHAR(500) DEFAULT NULL,
  `certificate_serial` VARCHAR(128) DEFAULT NULL,
  `certificate_thumbprint` VARCHAR(128) DEFAULT NULL,
  `smartcard_serial` VARCHAR(128) DEFAULT NULL,
  `smartcard_thumbprint` VARCHAR(128) DEFAULT NULL,
  `smartcard_label` VARCHAR(255) DEFAULT NULL,
  `failed_login_attempts` INT(11) NOT NULL DEFAULT 0,
  `locked_until` DATETIME DEFAULT NULL,
  `last_login_at` DATETIME DEFAULT NULL,
  `last_login_ip` VARCHAR(45) DEFAULT NULL,
  `email_verified` TINYINT(1) NOT NULL DEFAULT 0,
  `is_active` TINYINT(1) NOT NULL DEFAULT 1,
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `updated_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP
      ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  UNIQUE KEY (`username`),
  UNIQUE KEY (`email`),
  KEY `idx_certificate_dn` (`certificate_dn`(191)),
  KEY `idx_certificate_thumbprint` (`certificate_thumbprint`),
  KEY `idx_smartcard_thumbprint` (`smartcard_thumbprint`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

E.2.2 `sso_applications` - aplicaciones cliente OAuth 2.0 / OIDC

Finalidad. Catálogo de aplicaciones cliente registradas en el SSO. Cada fila representa un *relying party* OIDC con su `client_id`, secreto *hasheado*, `redirect_uri` autorizada, dominios permitidos y modo de acceso (público vs. restringido).

Campos críticos.

- `client_id VARCHAR(64) UNIQUE` - identificador público del cliente, presentado en la URL `/authorize` y `/token`.

- `client_secret_hash VARCHAR(255)` - Argon2id sobre el secreto en claro; nunca se almacena en claro.
- `redirect_uri VARCHAR(500)` - única URI autorizada para el *callback*; validación *exact match* sin *wildcards* (mitigación de *Open Redirect*).
- `allowed_domains TEXT` - lista de dominios separados por coma usada para validar el `Referer` y CORS si procede.
- `post_logout_redirect_uris TEXT` - añadido por la migración 012; URIs autorizadas para `RP-Initiated Logout` (OIDC), una por línea, *exact match*.
- `access_mode ENUM('all','restricted')` - añadido por la migración 010; en modo `restricted`, sólo los usuarios listados en `sso_application_users` pueden iniciar sesión en esta aplicación.

Claves e índices.

- `PRIMARY KEY (id)` (UUID).
- `UNIQUE KEY (client_id)`, `KEY (idx_client_id)`.
- `KEY (created_by)` + FK a `sso_users(id)`.

Observaciones de seguridad/RGPD.

- `client_secret_hash` se trata como secreto: cualquier acceso a esta columna se audita (`sso_audit_log` con `event_type='application_secret_viewed'`).
- En el panel admin, el secreto se muestra una sola vez en el momento de la creación o regeneración; tras eso solo se ve el hash.
- No contiene PII directa: `name`, `description`, `allowed_domains` son metadatos técnicos del *relying party*.

DDL resumido (los campos no críticos se omiten):

```
CREATE TABLE `sso_applications` (
  `id` CHAR(36) NOT NULL,
  `name` VARCHAR(200) NOT NULL,
  `client_id` VARCHAR(64) NOT NULL,
  `client_secret_hash` VARCHAR(255) NOT NULL,
  `redirect_uri` VARCHAR(500) NOT NULL,
  `allowed_domains` TEXT NOT NULL,
  `post_logout_redirect_uris` TEXT DEFAULT NULL
      COMMENT 'URIs de post-logout permitidas, una por línea (OIDC RP-Init
Logout)',
  `description` TEXT DEFAULT NULL,
  `is_active` TINYINT(1) NOT NULL DEFAULT 1,
  `access_mode` ENUM('all','restricted') NOT NULL DEFAULT 'all',
  `created_by` CHAR(36) NOT NULL,
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `updated_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP
      ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  UNIQUE KEY (`client_id`),
  KEY (`created_by`),
  CONSTRAINT `sso_applications_ibfk_1`
      FOREIGN KEY (`created_by`) REFERENCES `sso_users` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

E.2.3 `sso_application_users` - control de acceso por aplicación

Finalidad. Tabla de relación N:M que materializa la lista de usuarios autorizados para aplicaciones en modo `access_mode='restricted'`. Añadida por la migración 010.

Campos críticos. `application_id`, `user_id` (FK a sus respectivas tablas maestras), `granted_by` (auditoría del admin que otorgó el acceso).

Claves e índices.

- `PRIMARY KEY (id)` (UUID).
- `UNIQUE KEY uk_app_user (application_id, user_id)` - impide duplicados.
- FK con `ON DELETE CASCADE` a `sso_applications` y `sso_users`.

Observaciones de seguridad/RGPD. La eliminación lógica de un usuario (`is_active=0` en `sso_users`) no borra estas filas; el efecto bloqueante se logra porque el `JOIN` filtra por `sso_users.is_active=1`. La eliminación física sí limpia esta tabla por cascada.

DDL resumido.

```
CREATE TABLE `sso_application_users` (
  `id` CHAR(36) NOT NULL,
  `application_id` CHAR(36) NOT NULL,
  `user_id` CHAR(36) NOT NULL,
  `granted_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `granted_by` CHAR(36) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `uk_app_user` (`application_id`, `user_id`),
  KEY `idx_user_id` (`user_id`),
  CONSTRAINT `fk_appuser_application`
      FOREIGN KEY (`application_id`) REFERENCES `sso_applications` (`id`)
      ON DELETE CASCADE,
  CONSTRAINT `fk_appuser_user`
      FOREIGN KEY (`user_id`) REFERENCES `sso_users` (`id`) ON DELETE
CASCADE,
  CONSTRAINT `fk_appuser_granted_by`
      FOREIGN KEY (`granted_by`) REFERENCES `sso_users` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

E.2.4 `sso_sessions` - sesiones JWT con binding

Finalidad. Registro persistente de sesiones activas. Cada fila representa un par `(access token, refresh token)` emitido al usuario, con el *fingerprint* contextual (IP + User-Agent normalizado) que se valida en cada *refresh*.

Campos críticos.

- `id CHAR(36)` - UUID, expuesto como `sid` en el JWT (`aud=...` interno).
- `token_hash VARCHAR(255)` - SHA-256 hex del *access token*. Nunca se almacena el token en claro.
- `refresh_token_hash VARCHAR(255)` - SHA-256 hex del *refresh token*; permite invalidación granular sin desencriptar el JWT.
- `ua_fingerprint VARCHAR(64)` - añadido por la migración 004; SHA-256 del User-Agent normalizado (OS + navegador, descartando minor versions). Usado como señal contextual de riesgo, no como garantía criptográfica (véase Sección 7.5.3 del cuerpo).
- `auth_method VARCHAR(50)` - propaga el método primario de autenticación (`password`, `totp`, `certificate`, `smartcard`); influye en el nivel de aseguramiento (LoA) que se incluye en el `id_token`.

- `expires_at`, `refresh_expires_at` - ventanas de validez del *access* y *refresh* respectivamente; las purga el cron `cleanup.php`.

Claves e índices.

- `PRIMARY KEY (id)`.
- `UNIQUE KEY (token_hash)`, `KEY (idx_token_hash)`, `KEY (idx_refresh_token)` - *lookup* O(1) por hash.
- `KEY (idx_expires)` - usado por `cleanup.php` para purgar caducadas.
- FK `ON DELETE CASCADE` a `sso_users(id)` - un *delete* del usuario invalida todas sus sesiones.

Observaciones de seguridad/RGPD.

- `user_agent TEXT` se almacena en claro para auditoría; se incluye en el `dump` sólo si es necesario para la investigación (RGPD: principio de minimización de datos al consultar).
- `ip_address` es dato personal indirecto según el TJUE (sentencia *Breyer*); se conserva 90 días en sesiones activas y se borra en cascada al revocar.
- El *binding* IP+UA puede generar falsos positivos en CGNAT/proxy; por eso se trata como señal de riesgo, no como exclusión definitiva.

DDL resumido.

```
CREATE TABLE `sso_sessions` (
  `id` CHAR(36) NOT NULL,
  `user_id` CHAR(36) NOT NULL,
  `token_hash` VARCHAR(255) NOT NULL,
  `refresh_token_hash` VARCHAR(255) DEFAULT NULL,
  `ip_address` VARCHAR(45) NOT NULL,
  `user_agent` TEXT DEFAULT NULL,
  `ua_fingerprint` VARCHAR(64) DEFAULT NULL
      COMMENT 'SHA-256 del User-Agent normalizado (OS + navegador)',
  `auth_method` VARCHAR(50) NOT NULL,
  `last_activity` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `expires_at` DATETIME NOT NULL,
  `refresh_expires_at` DATETIME DEFAULT NULL,
  `is_active` TINYINT(1) NOT NULL DEFAULT 1,
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  UNIQUE KEY (`token_hash`),
  KEY `idx_user_id` (`user_id`),
  KEY `idx_expires` (`expires_at`),
  CONSTRAINT `sso_sessions_ibfk_1`
      FOREIGN KEY (`user_id`) REFERENCES `sso_users` (`id`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

E.2.5 `sso_authorization_codes` - códigos OAuth 2.0 + PKCE

Finalidad. Almacenamiento transitorio (TTL ≤ 60 s) de códigos de autorización emitidos en el endpoint `/api/authorize.php`. Incluye soporte PKCE (RFC 7636) añadido por la migración 011.

Campos críticos.

- `code VARCHAR(128) UNIQUE` - código de autorización (32 bytes aleatorios, base64url); de un solo uso.
- `code_challenge VARCHAR(128)` + `code_challenge_method VARCHAR(10)` - PKCE; si `code_challenge` está presente, el endpoint `/token` **exige** un `code_verifier` válido (SHA-256). El método `plain` se rechaza explícitamente por seguridad; sólo se acepta `S256`.
- `cert_serial VARCHAR(100)` - añadido por la migración 005; número de serie del certificado usado en la autenticación (vacío si fue password/TOTP). Se propaga en el JWT como *claim* personalizado para que las aplicaciones puedan verificar la cadena de confianza.
- `auth_method VARCHAR(50)` - propagado igual que en `sso_sessions`.

- `state VARCHAR(255)` - *anti-CSRF* del cliente, no se procesa en servidor más allá de almacenarlo.
- `expires_at DATETIME`, `used TINYINT(1)` - control de unicidad y caducidad; las filas expiradas se purgan por `cleanup.php`.

Claves e índices.

- `PRIMARY KEY (id)`.
- `UNIQUE KEY (code)`, `KEY (idx_code)` - *lookup* directo.
- `KEY (idx_expires)`.
- FK `ON DELETE CASCADE` a `sso_users` y a `sso_applications`.
- Índice adicional `idx_pkce_method` sobre `code_challenge_method` (auditoría de uso de PKCE).

Observaciones de seguridad/RGPD.

- **TTL agresivo** (60 s): si el *callback* tarda más, el código ya no sirve. Mitiga ataques de *code injection*.
- **`used=1` tras canje exitoso:** segundo `/token` con el mismo código devuelve `invalid_grant`. Esta verificación es complementaria al *delete* por `expires_at`.
- `cert_serial` no es PII por sí mismo (es un número de serie del certificado, no del DNI del titular), aunque pueda correlacionarse con identidades a través de `sso_user_certificates`.

DDL resumido.

```
CREATE TABLE `sso_authorization_codes` (
  `id` INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,
  `code` VARCHAR(128) NOT NULL,
  `user_id` CHAR(36) NOT NULL,
  `application_id` CHAR(36) NOT NULL,
  `redirect_uri` VARCHAR(500) NOT NULL,
  `scope` VARCHAR(255) NOT NULL DEFAULT 'openid profile email',
  `state` VARCHAR(255) DEFAULT NULL,
  `code_challenge` VARCHAR(128) DEFAULT NULL
      COMMENT 'PKCE code_challenge (RFC 7636, BASE64URL(SHA256(verifier)))',
  `code_challenge_method` VARCHAR(10) DEFAULT NULL
      COMMENT 'PKCE method: S256 (plain rechazado por seguridad)',
  `auth_method` VARCHAR(50) NOT NULL,
  `cert_serial` VARCHAR(100) DEFAULT NULL
      COMMENT 'Número de serie del certificado (sólo cert/smartcard)',
  `expires_at` DATETIME NOT NULL,
  `used` TINYINT(1) NOT NULL DEFAULT 0,
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  UNIQUE KEY (`code`),
  KEY `idx_expires` (`expires_at`),
  KEY `idx_pkce_method` (`code_challenge_method`),
  CONSTRAINT FOREIGN KEY (`user_id`) REFERENCES `sso_users` (`id`) ON DELETE
CASCADE,
  CONSTRAINT FOREIGN KEY (`application_id`)
      REFERENCES `sso_applications` (`id`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

E.2.6 `sso_keypairs` - claves de firma JWT RS256

Finalidad. Catálogo de pares de claves RS256 utilizados para firmar los JWT emitidos por el SSO. Soporta rotación con `kid` (Key ID) según OIDC: el `id_token` lleva el `kid` en el *header* y el *relying party* obtiene la clave pública correspondiente desde `/.well-known/jwks.json`.

Campos críticos.

- `kid VARCHAR(64) UNIQUE` - identificador del *keypair*, anunciado en el JWK y en el JWT *header*.
- `key_file_path VARCHAR(255)` - ruta relativa al fichero `.pem` de la clave privada dentro de `keys/`. Añadido por la migración 002; antes estaba en una columna `private_key_pem` ya eliminada.
- `public_key_pem TEXT` - clave pública en claro (única en BD); el JWKS la transforma a JWK.

- `algorithm VARCHAR(10)` - siempre `RS256` en el despliegue actual; preparado para `ES256` futuro.
- `is_active TINYINT(1)`, `expires_at` - permiten rotación: la clave nueva se marca `is_active=1`, la antigua permanece `is_active=1` hasta que todos los JWT emitidos con ella caduquen, y luego se desactiva.

Claves e índices. `PRIMARY KEY (id)`, `UNIQUE KEY (kid)`, `KEY (idx_kid)`.

Observaciones de seguridad/RGPD.

- No PII. Sólo material criptográfico.
- La clave privada NO está en BD. Se almacena en `sso/keys/<kid>.pem` con permisos `0600` y propietario `apache:apache`. Esto fue una **decisión consciente** de la migración 002: un `SELECT *` sobre `sso_keypairs` ya no puede comprometer las claves de firma.
- La rotación actualmente es manual y a demanda; figura como vía futura la rotación periódica programada (Sección 9.4.10 del cuerpo).

DDL resumido.

```
CREATE TABLE `sso_keypairs` (
  `id` INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,
  `kid` VARCHAR(64) NOT NULL,
  `key_file_path` VARCHAR(255) DEFAULT NULL
      COMMENT 'Nombre del archivo en keys/ (sin ruta completa)',
  `public_key_pem` TEXT NOT NULL,
  `algorithm` VARCHAR(10) NOT NULL DEFAULT 'RS256',
  `is_active` TINYINT(1) NOT NULL DEFAULT 1,
  `expires_at` DATETIME DEFAULT NULL,
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  UNIQUE KEY (`kid`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

E.2.7 `sso_trusted_issuers` - emisores de confianza para mTLS

Finalidad. Lista blanca de Autoridades Certificadoras aceptadas para la autenticación por certificado (FNMT, DNIe, Omnipresence TrustCA, RACV, etc.). Combina criterios de filtrado adicionales (EKU y *policy OIDs*) introducidos por la migración 003.

Campos críticos.

- `issuer_dn VARCHAR(500)` - Distinguished Name canónico del emisor; comparado con tolerancia a orden de atributos por la capa aplicativa.
- `issuer_type ENUM('certificate','smartcard','both')` - restringe si el emisor puede usarse como certificado *soft*, *smartcard* o ambos.
- `allowed_eku VARCHAR(500)` - OIDs de Extended Key Usage permitidos, separados por coma. `NULL`=cualquiera. Cierra el ataque de presentar un certificado de tipo *server auth* como si fuera *client auth*.
- `allowed_policy_oids VARCHAR(500)` - OIDs de *Certificate Policies* permitidos. Por ejemplo, para FNMT se exige `2.16.724.1.2.2.4.1` (persona física) o `2.16.724.1.2.2.4.2` (persona jurídica); rechaza certificados de prueba o de tipos no contemplados por el TFG.
- `ca_certificate_pem TEXT` - PEM del certificado raíz (opcional; útil para verificación local *offline*).

Claves e índices. `PRIMARY KEY (id)`, `KEY (idx_issuer_dn(191))`.

Observaciones de seguridad/RGPD.

- **No PII directa**, pero almacena la cadena de confianza configurada para el sistema; un atacante con acceso a esta tabla podría inferir qué tipos de certificado pueden iniciar sesión y eventualmente buscar uno coincidente.
- **Mantenimiento manual:** las CAs FNMT y DNIe se actualizan cuando hay rotación pública; el TFG documenta una vía futura de actualización automática vía *trust store* del sistema operativo.

DDL completo.

```
CREATE TABLE `sso_trusted_issuers` (
  `id` INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(255) NOT NULL,
  `issuer_dn` VARCHAR(500) NOT NULL,
  `issuer_type` ENUM('certificate','smartcard','both') NOT NULL DEFAULT
'both',
  `allowed_eku` VARCHAR(500) DEFAULT NULL
      COMMENT 'OIDs de EKU permitidos (separados por coma), NULL=todos',
  `allowed_policy_oids` VARCHAR(500) DEFAULT NULL
      COMMENT 'OIDs de Certificate Policies permitidos, NULL=todos',
  `ca_certificate_pem` TEXT DEFAULT NULL,
  `is_active` TINYINT(1) NOT NULL DEFAULT 1,
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `idx_issuer_dn` (`issuer_dn`(191))
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

E.2.8 `sso_user_certificates` - vínculo usuario↔certificado por tipo

Finalidad. Tabla 1:N que permite vincular un certificado por tipo a cada usuario. Añadida por la migración 006 al observar que un mismo usuario podía tener simultáneamente un certificado FNMT y un DNIe.

Campos críticos.

- `cert_type ENUM('fnmt','dnie','omnipresence')` - clasificación funcional del certificado, no técnica.
- `thumbprint VARCHAR(64) UNIQUE` - huella SHA-256 hex; punto de entrada del *lookup* durante el login.
- `serial_number`, `subject_dn`, `issuer_dn` - auditoría y depuración.

Claves e índices.

- `PRIMARY KEY (id)`.
- `UNIQUE KEY uk_user_cert_type (user_id, cert_type)` - un usuario sólo puede tener UN certificado de cada tipo.
- `UNIQUE KEY uk_thumbprint` - un mismo certificado no puede estar vinculado a dos usuarios distintos.
- FK `ON DELETE CASCADE` a `sso_users`.

Observaciones de seguridad/RGPD.

- **PII de alto valor:** el `subject_dn` contiene típicamente nombre completo y DNI del titular (formato FNMT/DNIe). Acceso restringido al panel admin y al propio usuario en `/profile`.
- **Si el usuario revoca el vínculo** desde el panel, se borra la fila; el certificado en sí sigue siendo válido externamente (es emitido por la FNMT, no por nosotros).

DDL resumido.

```
CREATE TABLE `sso_user_certificates` (
  `id` INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,
  `user_id` CHAR(36) NOT NULL,
  `cert_type` ENUM('fnmt','dnie','omnipresence') NOT NULL,
  `thumbprint` VARCHAR(64) NOT NULL,
  `serial_number` VARCHAR(100) DEFAULT NULL,
  `subject_dn` VARCHAR(500) DEFAULT NULL,
  `issuer_dn` VARCHAR(500) DEFAULT NULL,
  `linked_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  UNIQUE KEY `uk_user_cert_type` (`user_id`, `cert_type`),
  UNIQUE KEY `uk_thumbprint` (`thumbprint`),
  KEY `idx_user_id` (`user_id`),
  CONSTRAINT `fk_user_cert`
      FOREIGN KEY (`user_id`) REFERENCES `sso_users` (`id`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

E.2.9 `sso_recovery_codes` - códigos de recuperación TOTP

Finalidad. Códigos de un solo uso que permiten recuperar acceso si el usuario pierde su factor TOTP. Añadida por la migración 007. Se generan 10 códigos por usuario al activar TOTP.

Campos críticos.

- `code_hash VARCHAR(255)` - Argon2id sobre el código en claro (12 caracteres alfanuméricos, formato `xxxx-xxxx-xxxx`).
- `used TINYINT(1)`, `used_at DATETIME` - consumo *one-shot*.

Claves e índices. `PRIMARY KEY (id)`, `KEY (idx_user_id)`, FK `ON DELETE CASCADE` a `sso_users`.

Observaciones de seguridad/RGPD. Mismo tratamiento que `password_hash`: irrecuperable. Cuando un código se usa, se marca `used=1` pero no se borra, para que el panel admin pueda mostrar "7/10 códigos quedan" y auditar.

DDL resumido.

```
CREATE TABLE `sso_recovery_codes` (
  `id` INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,
  `user_id` CHAR(36) NOT NULL,
  `code_hash` VARCHAR(255) NOT NULL,
  `used` TINYINT(1) NOT NULL DEFAULT 0,
  `used_at` DATETIME DEFAULT NULL,
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `idx_user_id` (`user_id`),
  CONSTRAINT FOREIGN KEY (`user_id`) REFERENCES `sso_users` (`id`) ON DELETE
CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

E.2.10 `sso_password_resets` - one-time tokens de reset

Finalidad. Tokens criptográficos para flujo "olvidé mi contraseña" (TTL 30 min, *one-time use*).

Campos críticos. `token_hash` (SHA-256 hex del token enviado al correo), `expires_at`, `used`.

Claves e índices. `PRIMARY KEY (id)`, `KEY (idx_token)`, `KEY (idx_expires)`, FK CASCADE a `sso_users`.

Observaciones de seguridad/RGPD. El token va al *email* del usuario en claro; en BD vive sólo el hash. Tras `used=1` la fila se conserva 7 días por auditoría y se purga.

DDL resumido.

```
CREATE TABLE `sso_password_resets` (
  `id` INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,
  `user_id` CHAR(36) NOT NULL,
  `token_hash` VARCHAR(255) NOT NULL,
  `expires_at` DATETIME NOT NULL,
  `used` TINYINT(1) NOT NULL DEFAULT 0,
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `idx_token` (`token_hash`),
  KEY `idx_expires` (`expires_at`),
  CONSTRAINT FOREIGN KEY (`user_id`) REFERENCES `sso_users` (`id`) ON DELETE
CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

E.2.11 `sso_audit_log` - auditoría de seguridad estructurada

Finalidad. Registro centralizado de eventos relevantes para seguridad (intentos de login, emisión de códigos, *redirect_uri* bloqueadas, etc.). JSON estructurado en `details`.

Campos críticos.

- `event_type VARCHAR(50)` - taxonomía controlada (`auth_failed`, `authorization_code_issued`, `token_exchange_success`, `redirect_uri_blocked`, etc.).
- `severity ENUM('info','warning','critical')` - filtra alertas Telegram (sólo `critical`).
- `details LONGTEXT (CHECK json_valid)` - payload semiestructurado con contexto del evento.
- `user_id CHAR(36) NULL` - `NULL` en eventos anónimos (intentos de login antes de identificar usuario).

Claves e índices. `PRIMARY KEY (id)`, índices en `event_type`, `severity`, `user_id`, `ip_address`, `created_at`, y compuesto `(severity, created_at)` para *dashboards* de seguridad.

Observaciones de seguridad/RGPD. Conservación recomendada: 365 días. La columna `ip_address` es PII (TJUE *Breyer*); se incluye en exports RGPD sólo bajo solicitud autenticada del titular. No es un registro inalterable: cualquiera con

permisos `DELETE`/`UPDATE` sobre la tabla puede manipularlo. La evolución a *hash-chain* figura en Sección 9.4.9 del cuerpo del TFG.

DDL resumido.

```
CREATE TABLE `sso_audit_log` (
  `id` BIGINT(20) UNSIGNED NOT NULL AUTO_INCREMENT,
  `event_type` VARCHAR(50) NOT NULL,
  `severity` ENUM('info','warning','critical') NOT NULL DEFAULT 'info',
  `user_id` CHAR(36) DEFAULT NULL,
  `ip_address` VARCHAR(45) NOT NULL,
  `details` LONGTEXT DEFAULT NULL
      CHECK (json_valid(`details`)),
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `idx_audit_event` (`event_type`),
  KEY `idx_audit_severity` (`severity`),
  KEY `idx_audit_user` (`user_id`),
  KEY `idx_audit_ip` (`ip_address`),
  KEY `idx_audit_created` (`created_at`),
  KEY `idx_audit_sev_created` (`severity`, `created_at`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci
  COMMENT='Registro de auditoría de eventos de seguridad';
```

E.2.12 Tablas auxiliares del SSO

Tres tablas con función operativa que se documentan compactamente porque no tienen relaciones FK ni complejidad funcional reseñable:

Tabla	Finalidad	Observaciones
`sso_login_logs`	Log granular de cada intento de login (éxito/fallo/bloqueo). Más detallado que `sso_audit_log` (incluye `username_attempted`, `application_id`, `failure_reason`).	Conservación 90 días. Sirve para detección de *credential stuffing* y *brute force* a nivel de usuario.
`sso_rate_limits`	Contador deslizante de intentos por `(identifier, action)` en una ventana temporal. Implementa *rate limiting* en *login*, OTP, reset.	Sin FK (necesario para limitar también accesos anónimos por IP). Purga por TTL desde `cleanup.php`.

Tabla	Finalidad	Observaciones
`sso_password_resets`	Ya documentada en Sección E.2.10.	-

DDL resumido.

```
CREATE TABLE `sso_login_logs` (
  `id` BIGINT(20) UNSIGNED NOT NULL AUTO_INCREMENT,
  `user_id` CHAR(36) DEFAULT NULL,
  `username_attempted` VARCHAR(100) DEFAULT NULL,
  `application_id` CHAR(36) DEFAULT NULL,
  `application_name` VARCHAR(200) DEFAULT NULL,
  `auth_method` VARCHAR(50) NOT NULL,
  `ip_address` VARCHAR(45) NOT NULL,
  `user_agent` TEXT DEFAULT NULL,
  `status` ENUM('success','failed','blocked','totp_required','cert_invalid')
NOT NULL,
  `failure_reason` VARCHAR(255) DEFAULT NULL,
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `idx_user_id` (`user_id`),
  KEY `idx_ip` (`ip_address`),
  KEY `idx_created` (`created_at`),
  KEY `idx_status` (`status`),
  KEY `idx_application` (`application_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

CREATE TABLE `sso_rate_limits` (
  `id` BIGINT(20) UNSIGNED NOT NULL AUTO_INCREMENT,
  `identifier` VARCHAR(255) NOT NULL,
  `action` VARCHAR(50) NOT NULL,
  `attempts` INT(11) NOT NULL DEFAULT 1,
  `window_start` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `idx_identifier_action` (`identifier`, `action`),
  KEY `idx_window` (`window_start`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

E.3 Esquema del subsistema PKI (`marchernandezmo_pki_nuevo`)

E.3.1 `pki_users` - usuarios del portal PKI

Finalidad. Catálogo de usuarios autorizados a usar el portal PKI. Soporta dos orígenes: cuenta local (`created_via='local'`, con `password_hash`) o vinculación al SSO (`created_via='sso'`, con `sso_user_id`).

Campos críticos.

- `sso_user_id VARCHAR(36)` - referencia lógica al UUID en `sso_users.id`. Sin FK (BD distinta).
- `created_via ENUM('local','sso')` - añadido por la migración `add_oauth2_support`.
- `role ENUM('user','operator','admin','superadmin')` - modelo de roles más granular que en SSO (que sólo tiene `user`/`admin`), porque la PKI requiere distinguir operadores (revisan solicitudes) de admins (gestionan CAs).
- `password_hash` - opcional; `NULL` en usuarios vinculados al SSO (no necesitan contraseña local).
- `failed_login_attempts`, `locked_until` - paralelos a `sso_users`.

Claves e índices. `PRIMARY KEY (id)`, `UNIQUE KEY (username)`, índices en `sso_user_id`, `email`, `role`, `is_active`.

Observaciones de seguridad/RGPD. Misma política que `sso_users`. La columna `sso_user_id` se reparó en la migración 008 después de un bug crítico (véase Sección E.5).

DDL resumido.

```
CREATE TABLE `pki_users` (
  `id` INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,
  `sso_user_id` VARCHAR(36) DEFAULT NULL
      COMMENT 'UUID del usuario en el SSO',
  `username` VARCHAR(100) NOT NULL,
  `email` VARCHAR(255) NOT NULL,
  `full_name` VARCHAR(255) NOT NULL,
  `role` ENUM('user','operator','admin','superadmin') NOT NULL DEFAULT
'user',
  `is_active` TINYINT(1) NOT NULL DEFAULT 1,
  `created_via` ENUM('local','sso') DEFAULT 'local',
  `password_hash` VARCHAR(255) DEFAULT NULL,
  `last_login` DATETIME DEFAULT NULL,
  `last_login_ip` VARCHAR(45) DEFAULT NULL,
  `failed_login_attempts` INT(10) UNSIGNED NOT NULL DEFAULT 0,
  `locked_until` DATETIME DEFAULT NULL,
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `updated_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP
      ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  UNIQUE KEY (`username`),
  KEY `idx_user_sso` (`sso_user_id`),
  KEY `idx_user_email` (`email`),
  KEY `idx_user_role` (`role`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8mb3_general_ci;
```

E.3.2 `certificate_authorities` - catálogo de CAs

Finalidad. Almacena las Autoridades Certificadoras del sistema (Root y todas las intermedias), su estado y metadatos. No contiene claves privadas: la ruta al fichero `.key` en disco vive en `key_file_path`. Eliminada la columna `private_key_encrypted` por la migración consolidada PKI v2.0.

Campos críticos.

- `type ENUM('root','intermediate')` - discrimina raíz (única, *self-signed*) vs intermedias (firmadas por raíz o por otra intermedia).
- `parent_ca_id INT UNSIGNED` - auto-referencia FK con `ON DELETE SET NULL`. La raíz tiene `parent_ca_id=NULL`. Crítico para reconstruir cadenas (`Root → Int → Int_subord`) aunque el sistema emite siempre con `pathlen:0` desde el cuerpo de la migración del Anexo F (las intermedias actualmente desplegadas tienen `pathlen:1` por herencia histórica).
- `serial_counter BIGINT UNSIGNED` - contador local de cada CA, sincronizado con el fichero `serial` físico de OpenSSL (`pki/ca/<slug>/serial`). Cada vez

que se firma un certificado, se incrementa atomicamente en BD y se vuelca al fichero.

- `signature_algorithm` - `sha384WithRSAEncryption` para la raíz al firmar intermedias; `sha256WithRSAEncryption` para intermedias al firmar certificados finales. SHA-384 en raíz es el endurecimiento añadido al inicio del desarrollo PKI.
- `key_file_path VARCHAR(500)` - ruta relativa al fichero `.key` desde `ca_storage` (definido en `pki_config`). Permisos `0600` y propietario `apache:apache`.
- `crl_url`, `ocsp_url`, `ca_url` - URLs que se inyectan en cada certificado emitido (AIA y CDP).

Claves e índices. `PRIMARY KEY (id)`, FK auto-referencial `parent_ca_id`, índices en `type`, `is_active`.

Observaciones de seguridad/RGPD.

- **No PII.** El DN del *issuer* es metadatos técnicos sobre la organización emisora (no sobre personas físicas).
- **La columna `private_key_encrypted`** fue eliminada intencionalmente; el TFG documenta esta decisión como mitigación de "BD comprometida ≠ CA comprometida" (Sección 7.4.4 del cuerpo).
- El campo `certificate_pem` puede contener el certificado en formato `Certificate:\n Data:\n Version:...` (volcado *text* de OpenSSL) o en formato PEM (`-----BEGIN CERTIFICATE-----`), por compatibilidad histórica con dos iteraciones de la implementación.

DDL completo.

```
CREATE TABLE `certificate_authorities` (
  `id` INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(255) NOT NULL,
  `common_name` VARCHAR(255) NOT NULL,
  `type` ENUM('root','intermediate') NOT NULL,
  `parent_ca_id` INT(10) UNSIGNED DEFAULT NULL,
  `serial_counter` BIGINT(20) UNSIGNED NOT NULL DEFAULT 1,
  `certificate_pem` TEXT DEFAULT NULL,
  `certificate_der` MEDIUMBLOB DEFAULT NULL,
  `key_file_path` VARCHAR(500) DEFAULT NULL
      COMMENT 'Ruta relativa al archivo de clave privada',
  `public_key_pem` TEXT DEFAULT NULL,
  `subject_dn` TEXT NOT NULL,
  `key_algorithm` VARCHAR(20) NOT NULL DEFAULT 'RSA',
  `key_size` INT(10) UNSIGNED NOT NULL DEFAULT 4096,
  `signature_algorithm` VARCHAR(50) NOT NULL DEFAULT
'sha384WithRSAEncryption',
  `not_before` DATETIME NOT NULL,
  `not_after` DATETIME NOT NULL,
  `crl_url` VARCHAR(500) DEFAULT NULL,
  `ocsp_url` VARCHAR(500) DEFAULT NULL,
  `ca_url` VARCHAR(500) DEFAULT NULL,
  `is_active` TINYINT(1) NOT NULL DEFAULT 1,
  `fingerprint_sha256` VARCHAR(95) DEFAULT NULL,
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `updated_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP
      ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `fk_parent_ca` (`parent_ca_id`),
  KEY `idx_ca_type` (`type`),
  KEY `idx_ca_active` (`is_active`),
  CONSTRAINT `fk_parent_ca`
      FOREIGN KEY (`parent_ca_id`) REFERENCES `certificate_authorities`
(`id`)
      ON DELETE SET NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8mb3_general_ci;
```

E.3.3 `certificate_templates` - plantillas X.509

Finalidad. Catálogo de las 6 plantillas/perfiles disponibles para solicitar un certificado, cada una con sus restricciones (KU, EKU, tamaños de clave, validez). Vinculada al fichero `.cnf` correspondiente mediante `openssl_profile`. Las plantillas son genéricas (`ca_id=NULL`) en el modelo actual.

Campos críticos.

- `slug VARCHAR(100) UNIQUE` - identificador estable usado por la aplicación (`client-auth`, `server-auth`, `code-signing`, `document-signing`, `smime-email`, `vpn`).

- `openssl_profile VARCHAR(100)` - nombre del fichero `.cnf` (sin extensión) en `pki/public/config/openssl/profiles/`. Añadido por la migración consolidada PKI v2.0.
- `validity_days`, `key_usage`, `extended_key_usage`, `min_key_size`, `max_key_size`, `requires_approval` - definen las reglas que valida el frontend antes de enviar la CSR y que la PKI vuelve a validar al emitir.

Claves e índices. `PRIMARY KEY (id)`, `UNIQUE KEY (slug)`, FK opcional a `certificate_authorities`.

Observaciones de seguridad/RGPD. No PII. Cambios en plantillas se auditan en `audit_log` con `action='template_modified'`.

DDL resumido.

```sql
CREATE TABLE `certificate_templates` (
  `id` INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(100) NOT NULL,
  `slug` VARCHAR(100) NOT NULL,
  `description` TEXT DEFAULT NULL,
  `ca_id` INT(10) UNSIGNED DEFAULT NULL,
  `validity_days` INT(10) UNSIGNED NOT NULL DEFAULT 365,
  `key_usage` VARCHAR(500) DEFAULT NULL,
  `extended_key_usage` VARCHAR(500) DEFAULT NULL,
  `allowed_key_algorithms` VARCHAR(200) NOT NULL DEFAULT 'RSA',
  `min_key_size` INT(10) UNSIGNED NOT NULL DEFAULT 2048,
  `max_key_size` INT(10) UNSIGNED NOT NULL DEFAULT 4096,
  `requires_approval` TINYINT(1) NOT NULL DEFAULT 1,
  `is_active` TINYINT(1) NOT NULL DEFAULT 1,
  `openssl_profile` VARCHAR(100) DEFAULT NULL,
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `updated_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP
      ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  UNIQUE KEY (`slug`),
  KEY `fk_template_ca` (`ca_id`),
  CONSTRAINT `fk_template_ca`
      FOREIGN KEY (`ca_id`) REFERENCES `certificate_authorities` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8mb3_general_ci;
```

E.3.4 `certificate_requests` - solicitudes de emisión (CSR)

Finalidad. Cola de trabajo del *workflow* de emisión. Cada solicitud lleva la CSR del usuario, los metadatos del DN deseado, el estado en el flujo (`pending` →

`approved`/`rejected`/`info_requested` → `issued`/`cancelled`) y la traza del operador que la revisó.

Campos críticos.

- `status ENUM(...)` - máquina de estados con 6 valores. `info_requested` permite que el operador pida documentación adicional sin rechazar la solicitud.
- `csr_pem TEXT` - CSR en formato PEM. La clave privada nunca pasa por el servidor (se genera en navegador con Web Crypto API).
- `requested_validity_days`, `approved_validity_days` - añadidos en migraciones sucesivas; permiten que el admin apruebe una validez menor que la solicitada por motivos de política.
- `san_dns`, `san_ip`, `san_email` - Subject Alternative Names solicitados; se vuelven a validar contra la plantilla en el momento de la emisión.
- `request_ip`, `request_user_agent` - auditoría del momento de solicitud.

Claves e índices. `PRIMARY KEY (id)`, índices en `user_id`, `status`, `ca_id`, `created_at`.

Observaciones de seguridad/RGPD.

- **PII potencial** en `common_name`, `email`, `subject_*`: comúnmente nombre y DNI del solicitante. Acceso restringido a los roles `operator`/`admin`/`superadmin` y al propio solicitante.
- **Las CSR rechazadas se conservan** (status `rejected`) durante 1 año por trazabilidad; tras ese plazo se pseudonimizan (`common_name → 'REDACTED'`).

DDL resumido.

```sql
CREATE TABLE `certificate_requests` (
  `id` INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,
  `user_id` INT(10) UNSIGNED NOT NULL,
  `template_id` INT(10) UNSIGNED DEFAULT NULL,
  `ca_id` INT(10) UNSIGNED NOT NULL,
  `common_name` VARCHAR(255) NOT NULL,
  `organization` VARCHAR(255) DEFAULT NULL,
  `organizational_unit` VARCHAR(255) DEFAULT NULL,
  `locality` VARCHAR(255) DEFAULT NULL,
  `state_province` VARCHAR(255) DEFAULT NULL,
  `country` VARCHAR(2) DEFAULT NULL,
  `email` VARCHAR(255) DEFAULT NULL,
  `san_dns` TEXT DEFAULT NULL,
  `san_ip` TEXT DEFAULT NULL,
  `san_email` TEXT DEFAULT NULL,
  `csr_pem` TEXT DEFAULT NULL,
  `key_algorithm` VARCHAR(20) NOT NULL DEFAULT 'RSA',
  `key_size` INT(10) UNSIGNED NOT NULL DEFAULT 2048,
  `status`
ENUM('pending','approved','rejected','issued','cancelled','info_requested')
      NOT NULL DEFAULT 'pending',
  `admin_notes` TEXT DEFAULT NULL,
  `reviewed_by` INT(10) UNSIGNED DEFAULT NULL,
  `reviewed_at` DATETIME DEFAULT NULL,
  `requested_validity_days` INT(10) UNSIGNED DEFAULT NULL,
  `approved_validity_days` INT(10) UNSIGNED DEFAULT NULL,
  `request_ip` VARCHAR(45) NOT NULL,
  `request_user_agent` TEXT DEFAULT NULL,
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `updated_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP
      ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `idx_req_user` (`user_id`),
  KEY `idx_req_status` (`status`),
  KEY `idx_req_ca` (`ca_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8mb3_general_ci;
```

E.3.5 `certificates` - certificados emitidos

Finalidad. Catálogo maestro de todos los certificados X.509 emitidos por el sistema, su estado actual (válido/revocado/expirado), datos completos (PEM, DER, *fingerprints*) y metadatos de revocación.

Campos críticos.

- `serial_number VARCHAR(64)` + `ca_id INT UNSIGNED` - clave compuesta única `UNIQUE KEY idx_cert_serial_ca (serial_number, ca_id)`. El mismo `serial_number` puede repetirse entre CAs distintas (cada intermediate tiene su propio contador), pero **nunca** dentro de la misma CA.

- `certificate_pem TEXT`, `certificate_der MEDIUMBLOB` - almacenamiento dual: PEM para frontend/descarga humana, DER para verificación binaria (OCSP, Authority Info Access).
- `fingerprint_sha256 VARCHAR(95)`, `fingerprint_sha1 VARCHAR(59)` - anchos generosos por compatibilidad con formatos `aa:bb:cc:...` (con dos puntos cada 2 bytes). Indexados.
- `status ENUM('valid','revoked','expired','suspended')` - estado actual; `expired` se actualiza por cron al pasar `not_after`.
- `revocation_reason ENUM(...)` - los 9 motivos canónicos de RFC 5280 (`keyCompromise`, `cACompromise`, `affiliationChanged`, `superseded`, `cessationOfOperation`, `certificateHold`, `removeFromCRL`, `privilegeWithdrawn`, `aACompromise`) + `unspecified`.
- `key_usage`, `extended_key_usage`, `san` - copia desnormalizada de las extensiones X.509 para *queries* sin parsear PEM.
- `is_renewed TINYINT(1)`, `renewed_cert_id INT UNSIGNED` - soportan la operación "renovar": el certificado nuevo apunta al antiguo, y el antiguo se marca `is_renewed=1`.

Claves e índices. `PRIMARY KEY (id)`, `UNIQUE KEY idx_cert_serial_ca (serial_number, ca_id)`, FK a `certificate_authorities` y `certificate_requests`, índices en `user_id`, `status`, `common_name`, `not_after`, `fingerprint_sha256`.

Observaciones de seguridad/RGPD.

- PII significativa en `subject_dn`, `common_name`, `san_email`. Tratamiento equivalente al de `certificate_requests`.
- El campo `certificate_pem` es público de facto**:** un certificado X.509 está diseñado para distribuirse libremente. Sin embargo, la consulta directa al endpoint `/certificate-download.php` requiere autenticación y rol adecuado, por minimización de exposición.

- No se almacena ninguna clave privada del titular**.** La clave se genera en navegador y nunca llega al servidor (Web Crypto API).

DDL resumido.

```sql
CREATE TABLE `certificates` (
  `id` INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,
  `serial_number` VARCHAR(64) NOT NULL,
  `request_id` INT(10) UNSIGNED DEFAULT NULL,
  `ca_id` INT(10) UNSIGNED NOT NULL,
  `user_id` INT(10) UNSIGNED NOT NULL,
  `template_id` INT(10) UNSIGNED DEFAULT NULL,
  `common_name` VARCHAR(255) NOT NULL,
  `subject_dn` TEXT NOT NULL,
  `issuer_dn` TEXT NOT NULL,
  `certificate_pem` TEXT NOT NULL,
  `certificate_der` MEDIUMBLOB DEFAULT NULL,
  `public_key_pem` TEXT DEFAULT NULL,
  `key_algorithm` VARCHAR(20) NOT NULL,
  `key_size` INT(10) UNSIGNED NOT NULL,
  `signature_algorithm` VARCHAR(50) NOT NULL,
  `not_before` DATETIME NOT NULL,
  `not_after` DATETIME NOT NULL,
  `key_usage` VARCHAR(500) DEFAULT NULL,
  `extended_key_usage` VARCHAR(500) DEFAULT NULL,
  `san` TEXT DEFAULT NULL,
  `fingerprint_sha256` VARCHAR(95) NOT NULL,
  `fingerprint_sha1` VARCHAR(59) NOT NULL,
  `status` ENUM('valid','revoked','expired','suspended') NOT NULL DEFAULT
'valid',
  `revocation_date` DATETIME DEFAULT NULL,
  `revocation_reason` ENUM('unspecified','keyCompromise','cACompromise',
      'affiliationChanged','superseded','cessationOfOperation',
      'certificateHold','removeFromCRL','privilegeWithdrawn','aACompromise')
      DEFAULT NULL,
  `revoked_by` INT(10) UNSIGNED DEFAULT NULL,
  `issued_by` INT(10) UNSIGNED NOT NULL,
  `is_renewed` TINYINT(1) NOT NULL DEFAULT 0,
  `renewed_cert_id` INT(10) UNSIGNED DEFAULT NULL,
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `updated_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP
      ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  UNIQUE KEY `idx_cert_serial_ca` (`serial_number`, `ca_id`),
  KEY `idx_cert_user` (`user_id`),
  KEY `idx_cert_status` (`status`),
  KEY `idx_cert_cn` (`common_name`),
  KEY `idx_cert_expiry` (`not_after`),
  KEY `idx_cert_fingerprint` (`fingerprint_sha256`),
  CONSTRAINT `fk_cert_ca`
      FOREIGN KEY (`ca_id`) REFERENCES `certificate_authorities` (`id`),
  CONSTRAINT `fk_cert_request`
      FOREIGN KEY (`request_id`) REFERENCES `certificate_requests` (`id`)
      ON DELETE SET NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8mb3_general_ci;
```

E.3.6 `crl_records` - listas de revocación históricas

Finalidad. Histórico de todas las CRL generadas, indexadas por CA emisora y número de CRL. Cada cron de rotación (cada 4 h en producción) genera un nuevo registro.

Campos críticos.

- `crl_number BIGINT UNSIGNED` - número monotónico por CA, asignado por OpenSSL desde el fichero `crlnumber`.
- `this_update`, `next_update` - ventanas de validez de la CRL. `next_update = this_update + crl_validity_hours` (parámetro de `pki_config`).
- `crl_pem TEXT`, `crl_der MEDIUMBLOB` - almacenamiento dual; la última CRL válida por CA se exporta a disco en `/var/www/.../crl/` para servir por HTTP en `crl.marchernandez.es`.
- `entries_count INT UNSIGNED` - número de certificados revocados incluidos; permite *dashboards* sin parsear PEM.
- `sha256_hash VARCHAR(64)` - huella de la CRL completa; usado para detectar duplicados o corrupciones.
- `source VARCHAR(20)` - añadido por la migración 009; distingue entre `manual`/`auto`/`cron` para auditoría.

Claves e índices. `PRIMARY KEY (id)`, FK a `certificate_authorities`, `KEY idx_crl_ca_number (ca_id, crl_number)`.

Observaciones de seguridad/RGPD. No PII directa (la CRL contiene serial numbers, no nombres). Las CRL son públicas por diseño (RFC 5280).

DDL completo.

```sql
CREATE TABLE `crl_records` (
  `id` INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,
  `ca_id` INT(10) UNSIGNED NOT NULL,
  `crl_number` BIGINT(20) UNSIGNED NOT NULL,
  `this_update` DATETIME NOT NULL,
  `next_update` DATETIME NOT NULL,
  `crl_pem` TEXT NOT NULL,
  `crl_der` MEDIUMBLOB NOT NULL,
  `entries_count` INT(10) UNSIGNED NOT NULL DEFAULT 0,
  `sha256_hash` VARCHAR(64) NOT NULL,
  `file_size` INT(10) UNSIGNED NOT NULL DEFAULT 0,
  `source` VARCHAR(20) NOT NULL DEFAULT 'manual',
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `idx_crl_ca_number` (`ca_id`, `crl_number`),
  KEY `idx_crl_created` (`created_at`),
  CONSTRAINT `fk_crl_ca`
      FOREIGN KEY (`ca_id`) REFERENCES `certificate_authorities` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8mb3_general_ci;
```

E.3.7 `ocsp_queries` - histórico de consultas OCSP

Finalidad. Histórico de cada consulta OCSP recibida por el *responder* (`ocsp.marchernandez.es`), con la respuesta dada y el tiempo de procesamiento. Sirve para diagnóstico (latencia p95, tasa de *unknown*) y para *dashboards* del estado operativo.

Campos críticos.

- `serial_number VARCHAR(64)` - número de serie consultado.
- `response_status ENUM('good','revoked','unknown','error')` - RFC 6960 + estado `error` interno (timeout BD, fallo de firma, etc.).
- `response_time_ms INT UNSIGNED` - métrica operativa para SLOs.
- `cache_hit TINYINT(1)` - placeholder de caché. Honestidad técnica: el *responder* actual genera respuestas firmadas dinámicamente; este flag está reservado pero raramente activo.

Claves e índices. `PRIMARY KEY (id)`, índices en `serial_number`, `created_at`, `response_status`.

Observaciones de seguridad/RGPD.

- `request_ip` es PII potencial. Se conserva 90 días para diagnóstico y luego se pseudonimiza (`IP/24` para análisis agregado).
- **Tasa de consultas OCSP esperada:** baja en este despliegue (60 entradas en 4 meses según el dump), lo que indica que la mayoría de los clientes prefieren validar contra la CRL cacheada o no validan revocación en absoluto (problema generalizado del ecosistema X.509).

DDL resumido.

```
CREATE TABLE `ocsp_queries` (
  `id` BIGINT(20) UNSIGNED NOT NULL AUTO_INCREMENT,
  `serial_number` VARCHAR(64) NOT NULL,
  `ca_id` INT(10) UNSIGNED DEFAULT NULL,
  `request_ip` VARCHAR(45) NOT NULL,
  `response_status` ENUM('good','revoked','unknown','error') NOT NULL,
  `revocation_reason` VARCHAR(50) DEFAULT NULL,
  `response_time_ms` INT(10) UNSIGNED NOT NULL DEFAULT 0,
  `cache_hit` TINYINT(1) NOT NULL DEFAULT 0,
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `idx_ocsp_serial` (`serial_number`),
  KEY `idx_ocsp_created` (`created_at`),
  KEY `idx_ocsp_status` (`response_status`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8mb3_general_ci;
```

E.3.8 `audit_log` - auditoría de operaciones PKI

Finalidad. Registro centralizado de operaciones PKI (login, emisión, revocación, generación de CRL, creación de CA, modificación de plantilla, etc.). Modelo análogo a `sso_audit_log` pero adaptado al dominio PKI.

Campos críticos.

- `action VARCHAR(100)` - taxonomía controlada (`login`, `logout`, `certificate_issued`, `certificate_revoked`, `crl_generated`, `intermediate_ca_created`, `root_ca_created`, `request_approved`, `template_modified`, ...).
- `category ENUM('auth','certificate','request','revocation','crl','ocsp','admin','user','system')` - agrupa por dominio funcional.

- `target_type VARCHAR(50)`, `target_id INT UNSIGNED` - tipo y ID del objeto afectado (p. ej. `('certificate', 42)`); permite navegar desde el log al objeto auditado.
- `result ENUM('success','failure','error')`.
- `application VARCHAR(100)` - distingue `pki_portal` de fuentes externas (`crl_endpoint`, `ocsp_responder`, `cron`).

Claves e índices. `PRIMARY KEY (id)`, índices en `user_id`, `action`, `category`, `created_at`, `result`, `(target_type, target_id)`.

Observaciones de seguridad/RGPD. Las mismas que `sso_audit_log`: no es un registro inalterable, conservación de 365 días, IP y user_agent considerados PII indirecta.

DDL resumido.

```
CREATE TABLE `audit_log` (
  `id` BIGINT(20) UNSIGNED NOT NULL AUTO_INCREMENT,
  `user_id` INT(10) UNSIGNED DEFAULT NULL,
  `username` VARCHAR(100) DEFAULT NULL,
  `action` VARCHAR(100) NOT NULL,
  `category` ENUM('auth','certificate','request','revocation','crl','ocsp',
      'admin','user','system') NOT NULL,
  `target_type` VARCHAR(50) DEFAULT NULL,
  `target_id` INT(10) UNSIGNED DEFAULT NULL,
  `details` TEXT DEFAULT NULL,
  `ip_address` VARCHAR(45) NOT NULL,
  `user_agent` TEXT DEFAULT NULL,
  `application` VARCHAR(100) DEFAULT 'pki_portal',
  `result` ENUM('success','failure','error') NOT NULL DEFAULT 'success',
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `idx_audit_user` (`user_id`),
  KEY `idx_audit_action` (`action`),
  KEY `idx_audit_category` (`category`),
  KEY `idx_audit_created` (`created_at`),
  KEY `idx_audit_result` (`result`),
  KEY `idx_audit_target` (`target_type`, `target_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8mb3_general_ci;
```

E.3.9 Tablas auxiliares de la PKI

Tres tablas con función operativa documentadas de forma compacta:

Tabla	Finalidad	Observaciones
`user_sessions`	Sesiones locales del portal PKI (cookie + token). Independientes de las del SSO.	TTL `session_lifetime_hours` (8 h por defecto). Purgada por cron `cleanup`.
`crl_downloads`	Histórico de descargas de CRL (IP + UA). Diagnóstico de uso real del *endpoint* `crl.marchernandez.es`.	Sin FK a `crl_records` (la descarga puede ser de un fichero cacheado), sólo a `certificate_authorities` opcionalmente.
`pki_config`	Tabla clave-valor de configuración runtime de la PKI (`base_url`, `crl_validity_hours`, `default_cert_days`, etc.).	`PRIMARY KEY (config_key)`. Modificable desde panel admin. Cambios se auditan en `audit_log`.

DDL resumido.

```sql
CREATE TABLE `user_sessions` (
  `id` INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,
  `user_id` INT(10) UNSIGNED NOT NULL,
  `session_token` VARCHAR(128) NOT NULL,
  `ip_address` VARCHAR(45) NOT NULL,
  `user_agent` TEXT DEFAULT NULL,
  `auth_method` VARCHAR(50) NOT NULL DEFAULT 'password',
  `application` VARCHAR(100) DEFAULT 'pki_portal',
  `is_active` TINYINT(1) NOT NULL DEFAULT 1,
  `expires_at` DATETIME NOT NULL,
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `updated_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP
      ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  UNIQUE KEY (`session_token`),
  KEY `idx_session_user` (`user_id`),
  KEY `idx_session_expires` (`expires_at`),
  CONSTRAINT `fk_session_user`
      FOREIGN KEY (`user_id`) REFERENCES `pki_users` (`id`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8mb3_general_ci;

CREATE TABLE `crl_downloads` (
  `id` BIGINT(20) UNSIGNED NOT NULL AUTO_INCREMENT,
  `ca_id` INT(10) UNSIGNED DEFAULT NULL,
  `crl_id` INT(10) UNSIGNED DEFAULT NULL,
  `ip_address` VARCHAR(45) NOT NULL,
  `user_agent` TEXT DEFAULT NULL,
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `idx_crl_dl_created` (`created_at`),
  KEY `idx_crl_dl_ca` (`ca_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8mb3_general_ci;

CREATE TABLE `pki_config` (
  `config_key` VARCHAR(100) NOT NULL,
  `config_value` TEXT DEFAULT NULL,
  `description` VARCHAR(500) DEFAULT NULL,
  `updated_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP
      ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY (`config_key`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8mb3_general_ci;
```

E.5 Catálogo de migraciones y estado consolidado

Esta sección documenta el rastro completo de migraciones aplicadas durante el desarrollo y distingue las que pertenecen al esquema final operativo de las que fueron pasos intermedios o correcciones puntuales ya consolidadas.

E.5.1 Convenciones del catálogo

- **Estado:** `Aplicada (consolidada)` significa que la migración está reflejada en el `CREATE TABLE` del dump de producción; `Aplicada (parcial)` significa

que sólo afecta a datos existentes y no añade DDL nuevo; `Reemplazada` significa que su efecto ha sido absorbido por una migración posterior.

- **Idempotencia:** indica si la migración usa `IF NOT EXISTS`/`IF EXISTS` y por tanto puede re-aplicarse sin fallar.
- **Rollback:** indica si existe un procedimiento de retroceso documentado.

E.5.2 Migraciones del subsistema SSO

ID	Nombre	Fecha	Estado	Idempotente	Efecto principal
002	`keypairs_filesystem`	2026-02-15	Aplicada (consolidada)	Parcial	Mueve la clave privada RSA de `sso_keypairs.private_key_pem` a fichero `.pem` en disco; añade `key_file_path`. Hito de seguridad.
003	`trusted_issuers_eku`	2026-02-16	Aplicada (consolidada)	Sí	Añade `allowed_eku` y `allowed_policy_oids` a `sso_trusted_issuers`. Refuerza la validación de certificados.

ID	Nombre	Fecha	Estado	Idempotente	Efecto principal
004	`hardening`	2026-02-16	Aplicada (consolidada)	Sí	Añade `ua_fingerprint` a `sso_sessions`; crea `sso_password_resets` y `sso_audit_log`.
005	`oauth2_integration`	2026-02-17	Aplicada (consolidada)	Sí	Añade `cert_serial` a `sso_authorization_codes`; registra la aplicación PKI como cliente OAuth.
006	`multi_certificate_types`	2026-02-	Aplicada (consolidada)	Sí	Crea `sso_user_certificates`; migra datos de `sso_users.certificate_*` a la nueva tabla.

ID	Nombre	Fecha	Estado	Idempotente	Efecto principal
		25			
007	`recovery_codes`	2026-03-13	Aplicada (consolidada)	Sí	Crea `sso_recovery_codes`; permite TOTP con backup.
010	`application_user_access`	2026-03-31	Aplicada (consolidada)	No	Añade `access_mode` a `sso_applications`; crea `sso_application_users`.
011	`pkce_support`	2026-0	Aplicada (consolidada)	Sí	Añade `code_challenge` y `code_challenge_method` a `sso_authorization_codes` (RFC 7636 S256).

ID	Nombre	Fecha	Estado	Idempotente	Efecto principal
		5-01			
012	`post_logout_redirect_uris`	2026-05-10	Aplicada (consolidada)	Sí	Añade `post_logout_redirect_uris` a `sso_applications`; habilita OIDC RP-Initiated Logout completo.
-	`recovery_codes_prod.sql`	2026-03-13	Reemplazada	-	*Snapshot* puntual de códigos de recuperación generados para administradores; absorbido por migración 007.

Las migraciones 001, 008 y 009 no existen en el SSO (la numeración salta porque históricamente se usaron en la PKI).

E.5.3 Migraciones del subsistema PKI

ID	Nombre	Fecha	Estado	Idempotente	Efecto principal
-	`pki_migration.sql` (consolidada v2.0)	2026-02-15	Aplicada (consolidada)	No	Elimina `private_key_encrypted` de `certificate_authorities`, añade `key_file_path`; inserta las 6 plantillas X.509; añade `openssl_profile` y `approved_validity_days`. Hito de seguridad y de diseño funcional.
-	`add_oauth2_support`	2026-02-17	Aplicada (consolidada)	Sí	Añade `created_via` a `pki_users`; permite distinguir cuentas locales de cuentas SSO.
008	`fix_sso_user_mapping`	2026-03-	Aplicada (consolidada)	No	**Reparación crítica:** cambia `pki_users.sso_user_id` de `INT UNSIGNED` a `VARCHAR(36)`. Antes, MariaDB convertía silenciosamente UUIDs a entero (`'95c49423-...' → 95`) y el admin (con ID 0) recibía todos los UUIDs que empezaban

ID	Nombre	Fecha	Estado	Idempotente	Efecto principal
		18			por letra, comprometiendo la separación entre cuentas.
009	`crl_auto_generation`	2026-03-18	Aplicada (consolidada)	No	Añade `source` a `crl_records` para distinguir CRLs manuales/automáticas/cron.
-	`fix_intermediate_ca_directories.sql`	2026-04-10	Aplicada (parcial)	Sí	Corrige rutas erróneas en `certificate_authorities.key_file_path` tras una reorganización de directorios. Sólo afecta a datos.
-	`reset_all_cas.sql`	(no aplic	Reservada	-	*Script* de emergencia: limpia todas las CAs y certificados emitidos. Sólo para entornos de desarrollo; no se ejecuta en producción.

ID	Nombre	Fecha	Estado	Idempotente	Efecto principal
		ada)			

El catálogo PKI no usa numeración estricta 001-012 como el SSO; mezcla nombres descriptivos con números secuenciales por motivos históricos.

E.5.4 Migraciones intersistema (entrega del proyecto)

Estos artefactos sirven como guías operativas y no son migraciones DDL adicionales, pero se documentan aquí por completitud:

Artefacto	Propósito
`APLICAR_MIGRACIONES_BD_ACTUAL.sql`	Script *runbook* que ordena la aplicación de migraciones SSO 002→012 y PKI v2→009 sobre una BD ya inicializada.
`database.sql`	Esquema preliminar inicial (febrero 2026, nombres en español: `usuarios`, `certificados`, etc.). Reemplazado completamente por los dumps consolidados. Conservado únicamente por trazabilidad histórica.
`sso_migration.sql`	Wrapper consolidado de las migraciones SSO 002→007 + comentarios para despliegue manual.
`pki_migration.sql`	Wrapper consolidado de las migraciones PKI v2.0 (CA, plantillas, perfiles OpenSSL).

E.5.5 Reglas de oro del modelo final

Para no perder la coherencia del esquema en futuras evoluciones, se documentan las invariantes que deben preservarse:

1. **Las claves privadas nunca viven en BD.** Ni las de las CAs (`certificate_authorities.key_file_path` → fichero `.key`), ni las del firmante JWT (`sso_keypairs.key_file_path` → fichero `.pem`), ni las de los titulares (se generan y permanecen en el navegador).
2. **Las contraseñas/secretos se almacenan siempre con hash adecuado.** Argon2id para autenticación humana (`password_hash`, `code_hash`, `client_secret_hash`). SHA-256 hex para tokens efímeros (`token_hash`, `refresh_token_hash`).
3. **Toda tabla con PII tiene política de retención documentada** en este anexo (y de forma transversal en Sección 7.8 capa 9 - Auditoría y monitorización). Las purgas se ejecutan vía cron `cleanup.php` (SSO) y `cleanup.php` (PKI).
4. **Las claves foráneas con `ON DELETE CASCADE` se reservan para tablas dependientes** del ciclo de vida del padre (`sso_sessions`, `sso_recovery_codes`, `sso_user_certificates`, `sso_application_users`). Las tablas con datos auditables (`sso_audit_log`, `sso_login_logs`, `audit_log`, `certificates`, `certificate_requests`) usan `ON DELETE SET NULL` para preservar el rastro histórico, aunque el usuario sea borrado.
5. **El acoplamiento entre las dos bases de datos vive sólo en `pki_users.sso_user_id`,** sin FK física. Esta es una decisión consciente para no acoplar los ciclos de despliegue de ambos subsistemas.

E.6 Referencias

- Sección 7.3.3 *Modelo de datos completo* del cuerpo del TFG.
- Sección 7.4.4 *Almacenamiento y desencriptado de claves de CA*.
- Sección 7.5.1 *OAuth 2.0 Authorization Code + PKCE + OIDC*.

- Sección 7.8 (capa 9 - Auditoría y monitorización) y RNF-07 (cumplimiento RGPD/LOPDGDD) en Sección 7.2.2.
- Sección 9.3 *Limitaciones del proyecto* (charset utf8mb3 en PKI, auditoría no inalterable).
- Anexo F (plantillas OpenSSL) - complementa este anexo al describir cómo las `certificate_templates` se materializan en ficheros `.cnf`.
- Reglamento (UE) 2016/679 (RGPD).
- RFC 5280 (X.509), RFC 6960 (OCSP), RFC 6749 (OAuth 2.0), RFC 7636 (PKCE), RFC 6238 (TOTP), RFC 7519 (JWT), RFC 7517 (JWK).