Modelo de Datos Relacional

TT2025

https://github.com/dfleper/

18/01/2026

El Modelo de Datos Relacional de TT2025 define la estructura de la base de datos, especificando las tablas, sus campos y las relaciones ente ellas, garantizando la integridad y coherencia de la información del sistema.

Contenido

Diseño del modelo de datos

Diagrama Entidad Relación

Datos del documento

Nombre del proyecto:

Turbo Taller – Sistema web de gestión de citas para taller de mecánica rápida

Código del proyecto *(APCODE):*

TT2025

Integrantes del equipo:

dfleper (Domingo Fleitas) · GitHub

Fecha y versión del documento:

18/01/2026 – Versión 1.2

Versión	Fecha	Autor	Cambios Realizados	Motivos / Referencia
1.0	**21/12/2025**	**https://github.com/dlfeper**	**Creación del modelo inicial (tablas principales y relaciones)**	**Entrega inicial UT01.3**
1.1	**22/12/2025**	**https://github.com/dlfeper**	**Ajuste de cardinalidades e índices; revisión de reglas de integridad**	**Mejora de consistencia del modelo**
1.2	**18/01/2026**	**https://github.com/dlfeper**	**Incorporación de **campos de trazabilidad** en todas las tablas y `USERS.last_login_at` (LOPDGDD); mejora de formato y redacción**	**Corrección por feedback (UT01.3)**

1. Introducción y contexto

El objetivo de esta actividad es definir el **modelo de datos relacional** del proyecto **Turbo Taller (TT2025)**, identificando entidades, **atributos, relaciones** y **reglas de integridad** necesarias para soportar los procesos principales del sistema:

- Gestión de **usuarios y roles** (control de acceso – RBAC).
- Gestión de **clientes y empleados** (perfiles).
- Gestión de **vehículos** asociados a clientes.

- Gestión del **catálogo de servicios** del taller.
- Gestión de **citas en agenda** (reserva, modificación, cancelación, asignación de mecánicos).
- Trazabilidad por **historial de estados** de la cita..
- Gestión de **orden de trabajo** (ejecución real del trabajo).
- Registro de **servicios realizados** y **piezas utilizadas**.
- Gestión de **facturación y pagos**.
- Registro de **notificaciones** enviadas a usuarios.
- Definición de **horarios y festivos** para cálculo de disponibilidad

Este modelo se alinea con el alcance y requisitos definidos en UT01 (Plan de Proyecto) y UT01.1 (Documento de alcance), y con los procesos descritos en UT01.2 (Casos de uso).

2. Supuestos de diseño

1. Un usuario puede tener uno o varios roles (RBAC).
2. Un usuario puede tener perfil de cliente y/o empleado (perfiles separados de USERS).
3. Un cliente puede registrar varios vehículos, pero un vehículo pertenece a un único cliente.
4. Una cita se registra para un vehículo y un servicio principal.
5. La asignación de mecánico a una cita es opcional (puede realizarse posteriormente).
6. Una cita puede generar como máximo una orden de trabajo.
7. Una orden de trabajo puede generar como máximo una factura.
8. Una orden de trabajo contiene líneas de servicios realizados y piezas usadas.
9. La disponibilidad del taller se calcula combinando:
    - horario semanal (business_hours),
    - festivos/cierres (holidays),
    - citas existentes (appointments).

3. Modelo de datos (tablas) y propósito

A continuación se describen las tablas principales del sistema, su propósito y relaciones.

3.1 Seguridad y usuarios (RBAC)

USERS

Propósito: autenticación y datos comunes (email, contraseña, nombre, estado).

- **PK:** `id_user`.
- **Unicidad:** `email`
- **LOPDGDD (dato mínimo):** `last_login_at` (fecha/hora del último inicio de sesión).

ROLES

Propósito: catálogo de roles (cliente, recepcionista, mecánico, jefe, admin).

- **PK:** `id_role`.
- **Unicidad:** `nombre`.

USER_ROLES

Propósito: tabla puente N..N entre usuarios y roles.

- **PK compuesta:** `(id_user, id_role)`.

Cardinalidad:

- `USERS (1) — (N) USER_ROLES`
- `ROLES (1) — (N) USER_ROLES`

3.2 Perfiles

CUSTOMERS

Propósito: extensión de usuario cuando actúa como cliente (NIF, dirección, etc.).

- **PK:** `id_customer`.
- **FK única:** `id_user` (FK + UNIQUE)

EMPLOYEES

Propósito: extensión de usuario cuando actúa como empleado.

- **PK:** `id_employee`.
- **FK única:** `id_user` (FK + UNIQUE)..

Cardinalidad:

- `USERS (1) — (0..1) CUSTOMERS`
- `USERS (1) — (0..1) EMPLOYEES`

(Un usuario puede ser cliente, empleado, ambos o ninguno, según el caso.)

3.3 Dominio del taller

VEHICLES

Propósito: vehículos asociados a un cliente.

- **PK:** `id_vehicle`.
- **FK:** `id_customer`.
- **Unicidad:** `matricula`.

SERVICES

Propósito: catálogo de servicios del taller (precio base y minutos estimados).

- **PK:** `id_service`.
- **Unicidad**: `codigo`.

Cardinalidad:

- `CUSTOMERS (1) — (N) VEHICLES`
- `SERVICES (1) — (N) APPOINTMENTS` (ver citas)

3.4 Disponibilidad (horarios y festivos)

BUSINESS_HOURS

Propósito: horario semanal del taller por día.

- **PK:** `id_bh`.
- **Unicidad:** `dia_semana` (1 registro por día).

HOLIDAYS

Propósito: festivos/cierres excepcionales.

- **PK:** `id_holiday`.
- **Unicidad:** `fecha`.

3.5 Agenda (citas)

APPOINTMENTS

Propósito: citas del taller (agenda).

- **PK**: `id_appointment`.
- **FK:** `id_customer`, `id_vehicle`, `id_service`.
- **FK opcional:** `id_employee_asignado` (mecánico).
- **Campos clave:** `inicio`, `fin`, `estado`.

APPOINTMENT_STATUS_HISTORY

Propósito: historial/auditoría de cambios de estado de una cita.

- **PK:** `id_hist`.
- **FK:** `id_appointment`, `changed_by_user`.
- **Campos clave:** estado anterior/nuevo, `fecha_cambio`, nota.

Cardinalidad:

- `CUSTOMERS (1) — (N) APPOINTMENTS`
- `VEHICLES (1) — (N) APPOINTMENTS`
- `SERVICES (1) — (N) APPOINTMENTS`
- `EMPLOYEES (1) — (0..N) APPOINTMENTS` (asignación opcional)
- `APPOINTMENTS (1) — (N) APPOINTMENT_STATUS_HISTORY`

3.6 Ejecución real (orden de trabajo)

WORK_ORDERS

Propósito: orden de trabajo asociada a una cita.

- **PK:** `id_work_order`.
- **FK única:** `id_appointment` (FK + UNIQUE).
- **Campos clave:** `estado`, `diagnostico`, fechas.

WORK_ORDER_SERVICES

Propósito: líneas de servicios realizados en una OT.

- **PK:** `id_wos`.
- **FK:** `id_work_order`.
- **FK opcional:** `id_service` (si corresponde a un servicio del catálogo).

PARTS

Propósito: catálogo de piezas.

- **PK:** `id_part`.
- **Unicidad:** `sku`.
- **Campos clave:** precio unitario y stock.

WORK_ORDER_PARTS

Propósito: líneas de piezas usadas en una OT.

- **PK:** `id_wop`.
- **FK:** `id_work_order`, `id_part`.

Cardinalidad:

- `APPOINTMENTS (1) — (0..1) WORK_ORDERS`
- `WORK_ORDERS (1) — (N) WORK_ORDER_SERVICES`
- `WORK_ORDERS (1) — (N) WORK_ORDER_PARTS`
- `PARTS (1) — (N) WORK_ORDER_PARTS`

3.7 Facturación y pagos

INVOICES

Propósito: factura asociada a una OT.

- **PK:** `id_invoice`.
- **FK única:** `id_work_order` (FK + UNIQUE).
- **Unicidad:** `numero_factura`.

PAYMENTS

Propósito: pagos asociados a una factura (permite pagos parciales).

- **PK:** `id_payment`.
- **FK:** `id_invoice`.

Cardinalidad:

- `WORK_ORDERS (1) — (0..1) INVOICES`
- `INVOICES (1) — (N) PAYMENTS`

3.8 Notificaciones

NOTIFICATIONS

Propósito: registro de comunicaciones (email/sms).

- **PK:** `id_notification`.
- **FK:** `id_user`.
- **FK opcional:** `id_appointment`.
- **Campos:** canal, estado, fechas, `error_text`.

Cardinalidad:

- `USERS (1) — (N) NOTIFICATIONS`
- `APPOINTMENTS (1) — (0..N) NOTIFICATIONS`

4. Reglas de negocio e integridad

4.1 Restricciones de unicidad

- `USERS.email` único.
- `VEHICLES.matricula` único.
- `SERVICES.codigo` único.
- `PARTS.sku` único.
- `INVOICES.numero_factura` único.

4.2 Integridad referencial (FKs)

- No se permite crear una cita sin **cliente/vehículo/servicio** válidos.
- Para entidades principales con histórico, se recomienda **baja lógica** mediante `activo` (en lugar de borrado físico).

4.3 Reglas sobre fechas

- En `APPOINTMENTS`: `fin > inicio`.
- En `BUSINESS_HOURS`: `dia_semana` entre 1 y 7.

4.4 Control de solapes en citas

- Regla: un mismo mecánico no puede tener dos citas con rangos de tiempo solapados.
- Esta regla se controla preferentemente en la lógica de aplicación.
- Opcionalmente, para máxima integridad, se puede reforzar mediante **triggers** en MariaDB.

4.5 Trazabilidad

Se aplican dos niveles de trazabilidad:

1) **Historial de estados de cita** (`APPOINTMENT_STATUS_HISTORY`):
    - Registra quién cambia el estado, estado anterior/nuevo, fecha y nota opcional.

2) **Trazabilidad transversal (todas las tablas)** mediante campos de auditoría:
    - `created_by_user` (FK a `USERS.id_user`, permite NULL)
    - `created_at`
    - `updated_by_user` (FK a `USERS.id_user`, permite NULL)
    - `updated_at`

Además, para cumplir el requisito mínimo indicado en el feedback:

- `USERS.last_login_at` guarda la fecha/hora del último inicio de sesión.

4.6 Facturación consistente

- Los totales de factura se derivan de los detalles de la OT:
    - líneas de servicios realizados,
    - piezas usadas,
    - impuestos (IVA).
- PAYMENTS permite reflejar cobros parciales y el método de pago.

5. Justificación de decisiones de diseño

1. **Separación USERS / CUSTOMERS / EMPLOYEES**

   Permite una autenticación única y evita duplicar credenciales, manteniendo datos específicos en perfiles.
2. **RBAC con ROLES + USER_ROLES**

   Facilita la ampliación de permisos sin rediseñar el modelo (ej. nuevos roles).
3. **Separación APPOINTMENTS (planificación) y WORK_ORDERS (ejecución)**

   La cita es la reserva en agenda; la OT representa lo que realmente se hace en el taller, permitiendo:
    - histórico del vehículo,
    - trazabilidad,
    - facturación correcta.
4. **Detalle de OT (servicios/piezas) mediante tablas de líneas**

   Permite registrar con precisión el trabajo realizado y calcular facturación de forma transparente.

5. **Notificaciones persistentes**

   NOTIFICATIONS aporta evidencia de comunicaciones (confirmaciones, recordatorios, cancelaciones).

6. **Disponibilidad con horario + festivos**

   Con BUSINESS_HOURS y HOLIDAYS se soporta el cálculo de disponibilidad real del taller.

6. Cardinalidades y mapeo ER → SQL (MariaDB)

6.1 Tabla de cardinalidades (relaciones del modelo)

RELACIÓN	CARDINALIDAD	EXPLICACIÓN
USERS — USER_ROLES	1 — N	Un usuario puede tener varios roles asignados.
ROLES — USER_ROLES	1 — N	Un rol puede estar asignado a muchos usuarios.
USERS — CUSTOMERS	1 — 0..1	Un usuario puede ser cliente (o no). Si lo es, solo puede tener un perfil de cliente.
USERS — EMPLOYEES	1 — 0..1	Un usuario puede ser empleado (o no). Si lo es, solo puede tener un perfil de empleado.
CUSTOMERS — VEHICLES	1 — N	Un cliente puede registrar varios vehículos.
CUSTOMERS — APPOINTMENTS	1 — N	Un cliente puede tener muchas citas.
VEHICLES — APPOINTMENTS	1 — N	Un vehículo puede tener muchas citas a lo largo del tiempo.
SERVICES — APPOINTMENTS	1 — N	Un servicio puede estar asociado a muchas citas.
EMPLOYEES — APPOINTMENTS	1 — 0..N	Un empleado (mecánico) puede tener muchas citas asignadas; una cita puede no tener mecánico.
APPOINTMENTS — APPOINTMENT_STATUS _HISTORY	1 — N	Una cita genera múltiples cambios de estado (auditoría).
APPOINTMENTS — WORK_ORDERS	1 — 0..1	Una cita puede no iniciarse; si se inicia, genera como máximo una OT.

WORK_ORDERS — WORK_ORDER_SERVICES	**1 — N**	**Una OT contiene varias líneas de servicios realizados.**
WORK_ORDERS — WORK_ORDER_PARTS	**1 — N**	**Una OT contiene varias líneas de piezas utilizadas.**
PARTS — WORK_ORDER_PARTS	**1 — N**	**Una pieza puede usarse en muchas OTs (líneas).**
WORK_ORDERS — INVOICES	**1 — 0..1**	**Una OT puede no estar facturada; si lo está, genera una factura.**
INVOICES — PAYMENTS	**1 — N**	**Una factura puede tener uno o varios pagos (parciales).**
USERS — NOTIFICATIONS	**1 — N**	**Un usuario puede recibir múltiples notificaciones.**
APPOINTMENTS — NOTIFICATIONS	**1 — 0..N**	**Una cita puede generar notificaciones (confirmación, recordatorio, etc.). La notificación puede existir sin cita asociada.**

6.2 Mapeo ER → SQL: cómo se implementan las cardinalidades en MariaDB

Este apartado explica cómo se traduce un diagrama E/R (entidad–relación) al modelo relacional en SQL.

A) Relación 1 — N (uno a muchos)

Cómo se representa en SQL:

- La tabla del "lado N" incluye una **FK NOT NULL** apuntando al "lado 1".
- Se añade un índice en la FK para rendimiento.

Ejemplo en el proyecto:

- CUSTOMERS (1) — (N) VEHICLES
- vehicles.id_customer es **FK NOT NULL** hacia customers.id_customer .

Patrón SQL:

- Tabla padre: customers(id_customer PK)
- Tabla hija: vehicles(id_vehicle PK, id_customer FK NOT NULL)

B) Relación 1 — 0..1 (uno a uno opcional)

Qué significa:

- En el lado "0..1" puede que no exista registro relacionado.
- Si existe, solo puede existir uno.

Cómo se representa en SQL:

- La tabla "hija" contiene FK al padre.
- Esa FK debe tener una restricción **UNIQUE**, para que no se repita.

Ejemplo en el proyecto:

- USERS (1) — (0..1) CUSTOMERS
- customers.id_user es FK a users.id_user y además es **UNIQUE**.

Patrón SQL:

- Tabla padre: users(id_user PK)
- Tabla hija: customers(id_customer PK, id_user FK UNIQUE)

Esto garantiza:

- Un usuario puede tener 0 o 1 fila en customers.
- Nunca 2 (porque id_user es UNIQUE).

C) Relación 0..1 — N (opcional en la FK)

Qué significa:

- Una entidad puede o no estar asociada a la otra (FK opcional).

Cómo se representa en SQL:

- Se usa una FK NULLABLE (admite NULL).

Ejemplo en el proyecto:

- EMPLOYEES (1) — (0..N) APPOINTMENTS (asignación)
  - appointments.id_employee_asignado es FK NULL, porque una cita puede existir sin mecánico asignado.

Patrón SQL:

- appointments.id_employee_asignado NULL REFERENCES employees(id_employee)

D) Relación N — N (muchos a muchos)

Cómo se representa en SQL:

- Se crea una tabla intermedia (tabla puente) con:
  - dos FKs (a ambas tablas),
  - una PK compuesta (FK1 , FK2) o un id propio + UNIQUE.

Ejemplo en el proyecto:

- USERS (N) — (N) ROLES
  - Se implementa con user_roles(id_user, id_role) .

Patrón SQL:

- user_roles(id_user FK, id_role FK, PRIMARY KEY(id_user,id_role))

E) Relación 1 — 1 (obligatoria)

En este proyecto no se usa como obligatoria "pura", porque normalmente en un sistema real hay fases (cita puede existir sin OT, OT sin factura, etc.).

Si fuese obligatoria, se suele resolver de 2 formas:

1. Compartiendo PK (la hija usa el mismo id que el padre)
2. FK NOT NULL + UNIQUE y control de inserción (más común si se crea en momentos distintos)

7. Índices recomendados y justificación (rendimiento y consultas habituales)

En un sistema de gestión de citas y taller, la mayoría de consultas reales se repiten siempre:

- Ver agenda por fecha (día/semana/mes).
- Ver agenda de un mecánico.
- Filtrar citas por estado (pendiente/confirmada/en curso…).
- Consultar históricos (cambios de estado, facturas y pagos).
- Buscar por claves únicas (email, matrícula, código de servicio, SKU, número de factura).

Los índices elegidos en el DDL están orientados a estas consultas y a evitar "full table scans" a medida que crece el volumen de datos.

7.1 Índices clave por tabla (qué aceleran y por qué)

A) USERS

- uk_users_email (email)
  - **Para qué sirve:** login y búsquedas por email.
  - **Por qué es importante:** el email se usa como identificador de inicio de sesión y debe ser único y rápido de localizar

B) VEHICLE

- uk_vehicles_matricula (matricula)
  - **Para qué sirve:** localizar un vehículo por matrícula (caso muy habitual en taller).
  - **Por qué es importante:** evita duplicados y acelera búsquedas directas.
- idx_vehicles_customer (id_customer)
  - **Para qué sirve:** listar vehículos de un cliente.
  - **Consulta típica:** "mostrar mis coches" o "vehículos del cliente".

C) SERVICES

- uk_services_codigo (codigo)
  - **Para qué sirve:** localizar un servicio por código (catálogo).
- idx_services_nombre (nombre)
  - **Para qué sirve:** búsquedas por texto (autocomplete / listado filtrado).
- idx_services_activo (activo)
  - **Para qué sirve:** mostrar solo servicios activos en el front

D) APPOINTMENTS (tabla más crítica por volumen)

- idx_appt_inicio (inicio)
  - **Para qué sirve:** agenda por rango de fechas.
  - **Consulta típica:** "citas del día/semana".
- idx_appt_estado (estado)
  - **Para qué sirve:** filtros por estado en listados.
  - **Consulta típica:** "pendientes de confirmar", "citas finalizadas".

- idx_appt_employee (id_employee_asignado)
  - **Para qué sirve:** listar citas de un mecánico.
  - **Consulta típica:** "agenda del mecánico X".
- idx_appt_employee_inicio (id_employee_asignado, inicio) **(muy importante)**
  - **Para qué sirve:** agenda del mecánico ordenada/filtrada por fecha.
  - **Consulta típica:** "citas del mecánico X entre dos fechas".
  - **Por qué se usa índice compuesto:** optimiza consultas que filtran por mecánico y rango temporal a la vez.
- idx_appt_customer (id_customer)
  - **Para qué sirve:** ver el historial de citas de un cliente.
- idx_appt_vehicle (id_vehicle)
  - **Para qué sirve:** histórico por vehículo (muy típico para revisiones y mantenimiento).
- idx_appt_service (id_service)
  - **Para qué sirve:** estadísticas y listados por tipo de servicio.

E) APPOINTMENT_STATUS_HISTORY

- idx_hist_appt (id_appointment)
  - **Para qué sirve:** obtener el historial de estados de una cita concreta.
- idx_hist_user (changed_by_user)
  - **Para qué sirve:** auditoría por usuario (quién cambió estados).
- idx_hist_fecha (fecha_cambio)
  - **Para qué sirve:** ordenar/buscar cambios recientes.

F) WORK_ORDERS + líneas de detalle

- uk_work_orders_appt (id_appointment)
  - **Para qué sirve:** garantizar 1 OT por cita (0..1) y acceso rápido OT ← cita.
- idx_wos_wo (id_work_order)
  - **Para qué sirve:** listar líneas de servicios de una OT.
- idx_wop_wo (id_work_order)
  - **Para qué sirve:** listar líneas de piezas de una OT.
- idx_wop_part (id_part)
  - **Para qué sirve:** consultar en cuántas OTs se ha usado una pieza.

G) INVOICES y PAYMENTS

- uk_invoices_numero (numero_factura)
  - **Para qué sirve:** localizar factura por número (búsqueda directa y control de duplicados).
- uk_invoices_wo (id_work_order)
  - **Para qué sirve:** garantizar 0..1 factura por OT.
- idx_payments_invoice (id_invoice)
  - **Para qué sirve:** obtener pagos de una factura.
- idx_payments_paid_at (paid_at)
  - **Para qué sirve:** listados por fecha (cierres de caja, informes).

H) NOTIFICATIONS

- idx_notif_user (id_user)
  - **Para qué sirve:** historial de notificaciones del usuario.
- idx_notif_appt (id_appointment)
  - **Para qué sirve:** notificaciones asociadas a una cita (confirmación/recordatorio).
- idx_notif_estado (estado)
  - **Para qué sirve:** sistema de reintentos (pendientes/error).
- idx_notif_created_at (created_at)
  - **Para qué sirve:** obtener notificaciones recientes.

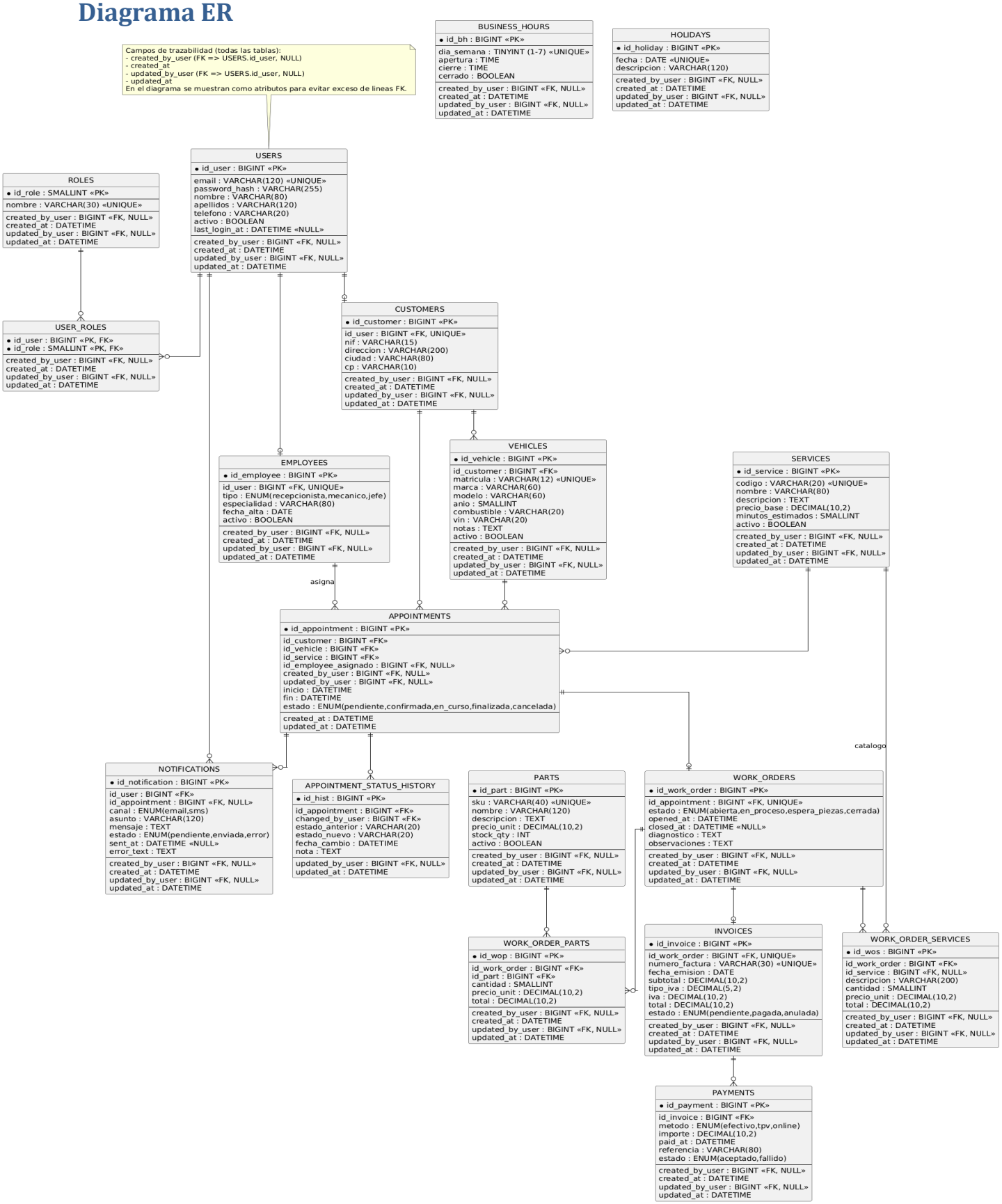
7.2 Consultas típicas del sistema (y qué índice las soporta)

Consulta Típica	Índice Principal
Agenda del día / semana (rango fechas)	`appointments(inicio)`
Agenda de un mecánico entre fechas	`appointments(id_employee_asignado, inicio)`
Citas pendientes / confirmadas	`appointments(estado)`
Historial de un vehículo	`appointments(id_vehicle)`
OT de una cita	`work_orders(id_appointment UNIQUE)`
Factura desde una OT	`invoices(id_work_order UNIQUE)`
Pagos de una factura	`payments(id_invoice)`
Notificaciones pendientes	`notifications(estado)`

8. Campos transversales de auditoría (todas las tablas)

Campo	Tipo (recomendado)	Reglas
`created_by_user`	BIGINT UNSIGNED	FK a `USERS.id_user`, **NULL permitido**
`created_at`	DATETIME	NOT NULL, DEFAULT CURRENT_TIMESTAMP
`updated_by_user	BIGINT UNSIGNED	FK a `USERS.id_user`, **NULL permitido**
`updated_at`	DATETIME	NOT NULL, DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP

Diagrama ER


Campos de trazabilidad (todas las tablas):
- created_by_user (FK => USERS.id_user, NULL)
- created_at
- updated_by_user (FK => USERS.id_user, NULL)
- updated_at
En el diagrama se muestran como atributos para evitar exceso de lineas FK.
BUSINESS_HOURS
• id_bh : BIGINT «PK»
dia_semana : TINYINT (1-7) «UNIQUE»
apertura : TIME
cierre : TIME
cerrado : BOOLEAN
created_by_user : BIGINT «FK, NULL»
created_at : DATETIME
updated_by_user : BIGINT «FK, NULL»
updated_at : DATETIME
HOLIDAYS
• id_holiday : BIGINT «PK»
fecha : DATE «UNIQUE»
descripcion : VARCHAR(120)
created_by_user : BIGINT «FK, NULL»
created_at : DATETIME
updated_by_user : BIGINT «FK, NULL»
updated_at : DATETIME
USERS
• id_user : BIGINT «PK»
email : VARCHAR(120) «UNIQUE»
password_hash : VARCHAR(255)
nombre : VARCHAR(80)
apellidos : VARCHAR(120)
telefono : VARCHAR(20)
activo : BOOLEAN
last_login_at : DATETIME «NULL»
created_by_user : BIGINT «FK, NULL»
created_at : DATETIME
updated_by_user : BIGINT «FK, NULL»
updated_at : DATETIME
ROLES
• id_role : SMALLINT «PK»
nombre : VARCHAR(30) «UNIQUE»
created_by_user : BIGINT «FK, NULL»
created_at : DATETIME
updated_by_user : BIGINT «FK, NULL»
updated_at : DATETIME
USER_ROLES
• id_user : BIGINT «PK, FK»
• id_role : SMALLINT «PK, FK»
created_by_user : BIGINT «FK, NULL»
created_at : DATETIME
updated_by_user : BIGINT «FK, NULL»
updated_at : DATETIME
CUSTOMERS
• id_customer : BIGINT «PK»
id_user : BIGINT «FK, UNIQUE»
nif : VARCHAR(15)
direccion : VARCHAR(200)
ciudad : VARCHAR(80)
cp : VARCHAR(10)
created_by_user : BIGINT «FK, NULL»
created_at : DATETIME
updated_by_user : BIGINT «FK, NULL»
updated_at : DATETIME
EMPLOYEES
• id_employee : BIGINT «PK»
id_user : BIGINT «FK, UNIQUE»
tipo : ENUM(recepcionista,mecanico,jefe)
especialidad : VARCHAR(80)
fecha_alta : DATE
activo : BOOLEAN
created_by_user : BIGINT «FK, NULL»
created_at : DATETIME
updated_by_user : BIGINT «FK, NULL»
updated_at : DATETIME
VEHICLES
• id_vehicle : BIGINT «PK»
id_customer : BIGINT «FK»
matricula : VARCHAR(12) «UNIQUE»
marca : VARCHAR(60)
modelo : VARCHAR(60)
anio : SMALLINT
combustible : VARCHAR(20)
vin : VARCHAR(20)
notas : TEXT
activo : BOOLEAN
created_by_user : BIGINT «FK, NULL»
created_at : DATETIME
updated_by_user : BIGINT «FK, NULL»
updated_at : DATETIME
SERVICES
• id_service : BIGINT «PK»
codigo : VARCHAR(20) «UNIQUE»
nombre : VARCHAR(80)
descripcion : TEXT
precio_base : DECIMAL(10,2)
minutos_estimados : SMALLINT
activo : BOOLEAN
created_by_user : BIGINT «FK, NULL»
created_at : DATETIME
updated_by_user : BIGINT «FK, NULL»
updated_at : DATETIME
asigna
APPOINTMENTS
• id_appointment : BIGINT «PK»
id_customer : BIGINT «FK»
id_vehicle : BIGINT «FK»
id_service : BIGINT «FK»
id_employee_asignado : BIGINT «FK, NULL»
created_by_user : BIGINT «FK, NULL»
updated_by_user : BIGINT «FK, NULL»
inicio : DATETIME
fin : DATETIME
estado : ENUM(pendiente,confirmada,en_curso,finalizada,cancelada)
created_at : DATETIME
updated_at : DATETIME
catalogo
NOTIFICATIONS
• id_notification : BIGINT «PK»
id_user : BIGINT «FK»
id_appointment : BIGINT «FK, NULL»
canal : ENUM(email,sms)
asunto : VARCHAR(120)
mensaje : TEXT
estado : ENUM(pendiente,enviada,error)
sent_at : DATETIME «NULL»
error_text : TEXT
created_by_user : BIGINT «FK, NULL»
created_at : DATETIME
updated_by_user : BIGINT «FK, NULL»
updated_at : DATETIME
APPOINTMENT_STATUS_HISTORY
• id_hist : BIGINT «PK»
id_appointment : BIGINT «FK»
changed_by_user : BIGINT «FK»
estado_anterior : VARCHAR(20)
estado_nuevo : VARCHAR(20)
fecha_cambio : DATETIME
nota : TEXT
updated_by_user : BIGINT «FK, NULL»
updated_at : DATETIME
PARTS
• id_part : BIGINT «PK»
sku : VARCHAR(40) «UNIQUE»
nombre : VARCHAR(120)
descripcion : TEXT
precio_unit : DECIMAL(10,2)
stock_qty : INT
activo : BOOLEAN
created_by_user : BIGINT «FK, NULL»
created_at : DATETIME
updated_by_user : BIGINT «FK, NULL»
updated_at : DATETIME
WORK_ORDERS
• id_work_order : BIGINT «PK»
id_appointment : BIGINT «FK, UNIQUE»
estado : ENUM(abierta,en_proceso,espera_piezas,cerrada)
opened_at : DATETIME
closed_at : DATETIME «NULL»
diagnostico : TEXT
observaciones : TEXT
created_by_user : BIGINT «FK, NULL»
created_at : DATETIME
updated_by_user : BIGINT «FK, NULL»
updated_at : DATETIME
WORK_ORDER_PARTS
• id_wop : BIGINT «PK»
id_work_order : BIGINT «FK»
id_part : BIGINT «FK»
cantidad : SMALLINT
precio_unit : DECIMAL(10,2)
total : DECIMAL(10,2)
created_by_user : BIGINT «FK, NULL»
created_at : DATETIME
updated_by_user : BIGINT «FK, NULL»
updated_at : DATETIME
INVOICES
• id_invoice : BIGINT «PK»
id_work_order : BIGINT «FK, UNIQUE»
numero_factura : VARCHAR(30) «UNIQUE»
fecha_emision : DATE
subtotal : DECIMAL(10,2)
tipo_iva : DECIMAL(5,2)
iva : DECIMAL(10,2)
total : DECIMAL(10,2)
estado : ENUM(pendiente,pagada,anulada)
created_by_user : BIGINT «FK, NULL»
created_at : DATETIME
updated_by_user : BIGINT «FK, NULL»
updated_at : DATETIME
WORK_ORDER_SERVICES
• id_wos : BIGINT «PK»
id_work_order : BIGINT «FK»
id_service : BIGINT «FK, NULL»
descripcion : VARCHAR(200)
cantidad : SMALLINT
precio_unit : DECIMAL(10,2)
total : DECIMAL(10,2)
created_by_user : BIGINT «FK, NULL»
created_at : DATETIME
updated_by_user : BIGINT «FK, NULL»
updated_at : DATETIME
PAYMENTS
• id_payment : BIGINT «PK»
id_invoice : BIGINT «FK»
metodo : ENUM(efectivo,tpv,online)
importe : DECIMAL(10,2)
paid_at : DATETIME
referencia : VARCHAR(80)
estado : ENUM(aceptado,fallido)
created_by_user : BIGINT «FK, NULL»
created_at : DATETIME
updated_by_user : BIGINT «FK, NULL»
updated_at : DATETIME