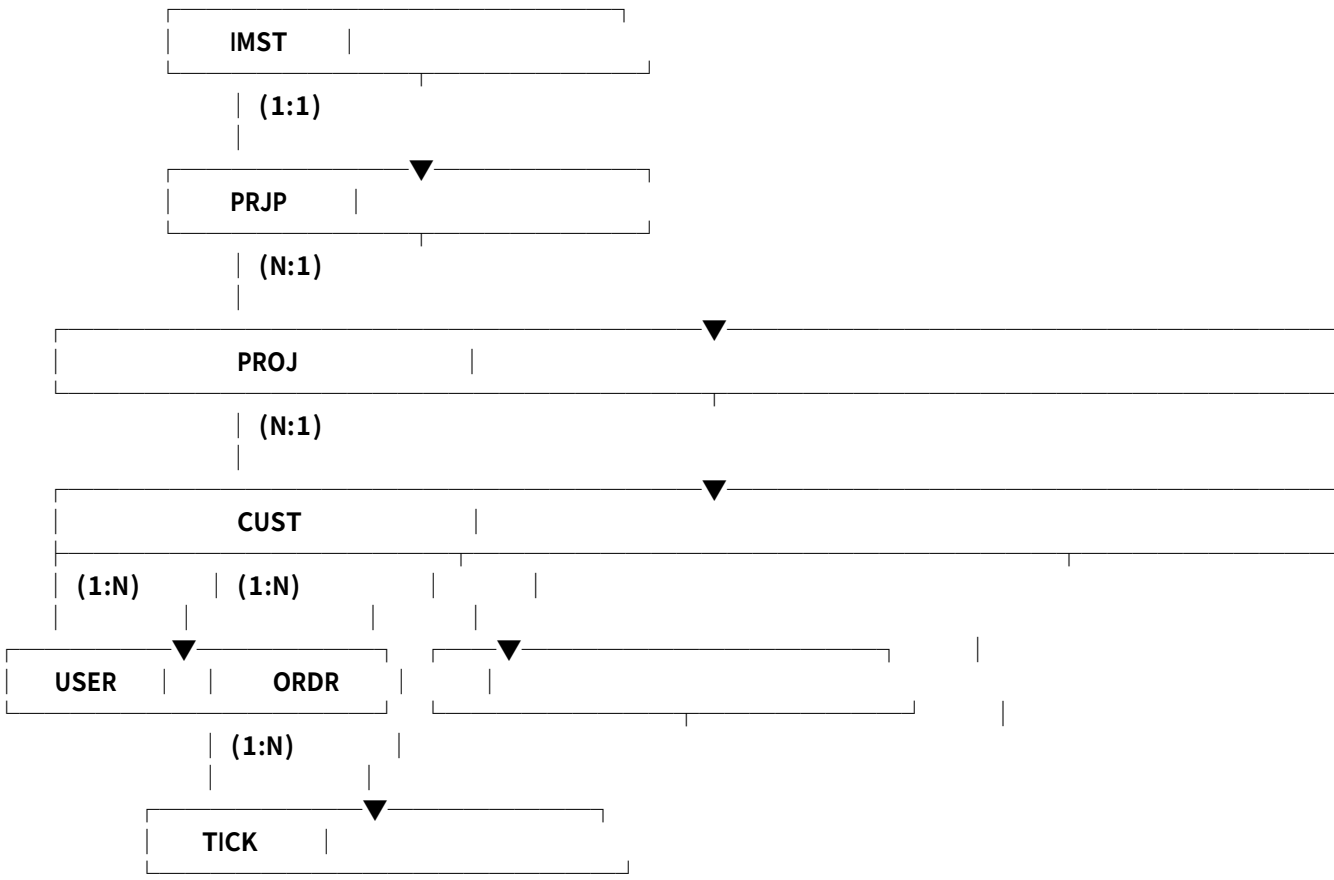
Documento de Arquitectura de Datos – Proyecto ObraTrack

Este documento corresponde a una muestra del avance realizado en el proyecto **ObraTrack**, actualmente en desarrollo por **Echnelapp** para la empresa **Transex**. Incluye el modelo entidad–relación actualizado (versión seleccionada), descripción detallada de las tablas principales y ejemplos de mapeo en TypeORM.

Diagrama ER (Versión 3 — Robusta)



Descripción detallada de las tablas

Tabla: cust (Customer)

La tabla **cust** representa a los clientes corporativos que usan el sistema. Es la entidad raíz para proyectos, órdenes y usuarios (cada usuario pertenece a exactamente un cliente).

Columnas propuestas: - **id** (PK, serial/autoincrement)
- **cust_code** (VARCHAR, UNIQUE) — código identificador del cliente
- **name** (VARCHAR) — nombre completo
- **sort_name** (VARCHAR) — nombre corto o abreviado
- **addr_line_1**, **addr_line_2**, **addr_city**, **addr_state**, **addr_cntry**, **addr_zip**
- **phone**, **email**
- **created_at**, **updated_at** (timestamps)

Ejemplo TypeORM (entity): @Entity() export class Cust { @PrimaryGeneratedColumn() id: number; @Column({ unique: true }) cust_code: string; @Column({ nullable: true }) name: string; /* ...otras columnas de dirección y contacto... */ @CreateDateColumn() created_at: Date; @UpdateDateColumn() updated_at: Date; @OneToMany(() => User, (user) => user.cust) users: User[]; @OneToMany(() => Proj, (proj) => proj.cust) projects: Proj[]; @OneToMany(() => Ordr, (ordr) => ordr.cust) orders: Ordr[]; }

Tabla: user (Usuario del sistema)

La tabla **user** almacena credenciales y metadatos de usuarios. Cada registro **user** pertenece a exactamente un **cust** (cust_id es NOT NULL), asegurando el requisito de un solo cliente por usuario.

Columnas propuestas: - **id** (PK)
- **username** (VARCHAR, UNIQUE)
- **password_hash** (VARCHAR)
- **email** (VARCHAR)
- **role** (VARCHAR / ENUM) — e.g., 'admin', 'manager', 'viewer'
- **cust_id** (FK → cust.id) — NOT NULL
- **last_login** (timestamp, opcional)
- **created_at**, **updated_at**

Ejemplo TypeORM (entity): @Entity() export class User { @PrimaryGeneratedColumn() id: number; @Column({ unique: true }) username: string; @Column() password_hash: string; @Column({ nullable: true }) email: string; @Column({ default: 'viewer' }) role: string; @ManyToOne(() => Cust, (cust) => cust.users, { nullable: false }) @JoinColumn({ name: 'cust_id' }) cust: Cust; @CreateDateColumn() created_at: Date; @UpdateDateColumn() updated_at: Date; }

Tabla: proj (Proyecto)

La tabla **proj** agrupa proyectos vinculados a un cliente (cust). Un proyecto puede tener múltiples registro **prjp** (productos/enlaces) asociadas.

Columnas propuestas: - **id** (PK)
- **proj_code** (VARCHAR, opcional, UNIQUE por cliente)
- **cust_id** (FK → cust.id)

- **name**, **description**
- **start_date**, **end_date**
- **status**
- **created_at**, **updated_at**

@Entity() export class Proj { @PrimaryGeneratedColumn() id: number; @Column() name: string; @ManyToOne(() => Cust, (cust) => cust.projects, { nullable: false }) @JoinColumn({ name: 'cust_id' }) cust: Cust; @OneToMany(() => Prjp, (prjp) => prjp.proj) prjps: Prjp[]; }

Tabla: prjp (Proyecto-Producto / Enlace)

La tabla **prjp** modela ítems o productos asociados a un **proj**. Cada prjp puede estar enlazado a una entrada de inventario/estado (**imst**).

Columnas propuestas: - **id** (PK)
- **proj_id** (FK → proj.id)
- **product_code**, **quantity**, **notes**
- **imst_id** (FK → imst.id, opcional)
- **created_at**, **updated_at**

@Entity() export class Prjp { @PrimaryGeneratedColumn() id: number; @ManyToOne(() => Proj, (proj) => proj.prjps, { nullable: false }) proj: Proj; @ManyToOne(() => Imst, (imst) => imst.prjps, { nullable: true }) imst: Imst; }

Tabla: imst (Inventario / Estado)

La tabla **imst** representa el inventario o estado asociado a un prjp (relación 1:1 en diseño). Puede contener detalles técnicos del producto/ítem.

Columnas propuestas: - **id** (PK)
- **sku**, **description**
- **stock**, **location**
- **created_at**, **updated_at**

@Entity() export class Imst { @PrimaryGeneratedColumn() id: number; @Column({ nullable: true }) sku: string; @OneToOne(() => Prjp, (prjp) => prjp.imst) prjp: Prjp; }

Tabla: ordr (Orden)

La tabla **ordr** contiene órdenes de trabajo o pedidos vinculados a un cliente. Una orden puede tener múltiples tickets.

Columnas propuestas: - **id** (PK)
- **ordr_code** (VARCHAR, opcional)
- **cust_id** (FK → cust.id)
- **status**, **priority**
- **due_date**
- **created_at**, **updated_at**

@Entity() export class Ordr { @PrimaryGeneratedColumn() id: number; @ManyToOne(() => Cust, (cust) => cust.orders, { nullable: false }) cust: Cust; @OneToMany(() => Tick, (tick) => tick.ordr) ticks:

Tick[]; }

Tabla: tick (Ticket)

La tabla **tick** modela tickets asociados a órdenes. Útil para seguimiento de tareas, incidencias o actividades.

Columnas propuestas: - **id** (PK)
- **ordr_id** (FK → ordr.id)
- **title**, **description**
- **status**, **assigned_to** (FK → user.id, opcional)
- **created_at**, **updated_at**

@Entity() export class Tick { @PrimaryGeneratedColumn() id: number; @ManyToOne(() => Ordr, (ordr) => ordr.ticks, { nullable: false }) ordr: Ordr; @Column() title: string; @Column({ nullable: true }) description: string; }

Notas: Las relaciones FK deben tener índices para rendimiento. Considerar restricciones de integridad (ON DELETE/UPDATE) según la política de negocio. Los ejemplos TypeORM son plantillas y pueden ajustarse a convenciones del proyecto.