Documentación de Base de Datos

Fecha: 22/07/2025

Este documento describe el diseño y funcionamiento de la base de datos desarrollada para el sistema de gestión.

# 1. Introducción general

# Mercury Solutions S.A. es una empresa dedicada a la comercialización y distribución de productos de consumo masivo. Ante el crecimiento de sus operaciones y la necesidad de optimizar la gestión de sus procesos internos, la dirección ha solicitado el desarrollo de un Sistema de Información Integrado, basado en una única base de datos relacional.

# Este sistema tiene como objetivo centralizar la información y asegurar la integridad, disponibilidad y consistencia de los datos que atraviesan las diferentes áreas funcionales de la empresa: Compras, Ventas, Logística & Stock y Recursos Humanos. La unificación de estos módulos permitirá mejorar la toma de decisiones, reducir redundancias y aumentar la eficiencia operativa.

# El presente trabajo aborda el diseño e implementación de dicho sistema, siguiendo buenas prácticas de modelado de datos, normalización, uso de restricciones de integridad y construcción de vistas y disparadores que garanticen la coherencia de la información almacenada.

# 2. Diccionario de datos (por tabla)

A continuación se detalla la estructura de cada tabla de la base de datos.

|  |  |
| --- | --- |
| **DICCIONARIO DE DATOS** | |
| **Tabla** | **Columnas (PK = Primary Key, FK = Foreign Key, R = Required(Not Null) )** |
| paises | pais\_id int PK R AUTO\_INCREMENT, pais\_id int R AUTO\_INCREMENT, nombre varchar(50) R |
| provincias | provincia\_id int PK R AUTO\_INCREMENT, pais\_id int FK R, nombre varchar(50) R |
| ciudades | ciudad\_id int PK R AUTO\_INCREMENT, provincia\_id int FK R, nombre varchar(50) R |
| codigos\_postales | codigo\_postal\_id int PK R AUTO\_INCREMENT, ciudad\_id int FK R, codigo int R |
| direcciones | direccion\_id int PK R AUTO\_INCREMENT, calle varchar(50) R, numero int R, piso int , depto int , ciudad\_id int FK R, provincia\_id int FK R, pais\_id int FK R, codigo\_postal\_id int FK |
| tipo\_departamentos | tipo\_departamento\_id int PK R AUTO\_INCREMENT, nombre varchar(50) R, descripcion varchar(100) |
| sucursales | sucursal\_id int PK R AUTO\_INCREMENT, nombre varchar(50) R, direccion\_id int FK |
| departamentos | departamento\_id int PK R AUTO\_INCREMENT, sucursal\_id int FK, tipo\_departamento\_id int FK, nombre varchar(50) R |
| tipo\_empleados | tipo\_empleado\_id int PK R AUTO\_INCREMENT, nombre varchar(50) R, descripcion varchar(100) |
| empleados | empleado\_id int PK R AUTO\_INCREMENT, nombre varchar(50) R, apellido varchar(50) R, dni varchar(15) , cuil varchar(15) , fecha\_nacimiento datetime , fecha\_alta datetime , email varchar(75) , telefono varchar(20) , tipo\_empleado\_id int FK, departamento\_id int FK, sucursal\_id int FK, direccion\_id int FK |
| clientes | cliente\_id int PK R AUTO\_INCREMENT, razon\_social varchar(50) R, cuil varchar(15) , email varchar(50) , telefono varchar(20) , fecha\_alta date , credito decimal , direccion\_id int FK |
| rubros | rubro\_id int PK R AUTO\_INCREMENT, nombre varchar(50) R, descripcion varchar(100) |
| proveedores | proveedor\_id int PK R AUTO\_INCREMENT, razon\_social varchar(50) R, cuit varchar(15) , email varchar(50) , telefono varchar(20) , direccion\_id int FK, fecha\_alta date , credito int , rubro\_id int FK |
| categoria\_productos | categoria\_id int PK R AUTO\_INCREMENT, nombre varchar(50) R, descripcion varchar(100) |
| unidades\_medida | unidad\_medida\_id int PK R AUTO\_INCREMENT, unidad varchar(10) |
| productos | producto\_id int PK R AUTO\_INCREMENT, rubro\_id int FK, categoria\_id int FK, nombre varchar(50) R, descripcion varchar(100) , unidad\_medida\_id int FK R, codigo\_barra varchar(25) , precio\_unitario decimal R |
| producto\_proveedor | producto\_id int FK R, producto\_id int PK R, proveedor\_id int FK R, proveedor\_id int PK R, precio\_compra decimal , fecha\_alta datetime |
| rankings | rubro\_id int FK R, rubro\_id int PK R, proveedor\_id int FK R, proveedor\_id int PK R, ranking int , fecha\_ranking datetime , comentarios varchar(100) |
| almacenes | almacen\_id int PK R AUTO\_INCREMENT, sucursal\_id int FK, nombre varchar(50) R, direccion\_id int FK |
| stocks | stock\_id int PK R AUTO\_INCREMENT, producto\_id int FK R, almacen\_id int FK R, cantidad int , stock\_minimo int , stock\_maximo int |
| estados\_compras | estado\_id int PK R AUTO\_INCREMENT, estado varchar(15) R |
| condiciones\_pagos | condicion\_pago\_id int PK R AUTO\_INCREMENT, condicion\_pago varchar(50) R |
| orden\_compras | orden\_compra\_id int PK R AUTO\_INCREMENT, proveedor\_id int FK, empleado\_id int FK, sucursal\_id int FK, departamento\_id int FK, fecha\_orden datetime R, estado\_id int FK, total decimal , condicion\_pago\_id int FK |
| detalle\_orden\_compras | detalle\_orden\_id int PK R AUTO\_INCREMENT, orden\_compra\_id int FK R, producto\_id int FK R, cantidad int R, precio\_unitario decimal R, total\_linea decimal STORED GENERATED |
| estado\_remitos | estado\_id int PK R AUTO\_INCREMENT, estado varchar(20) R |
| remitos | remito\_id int PK R AUTO\_INCREMENT, proveedor\_id int FK, sucursal\_id int FK, orden\_compra\_id int FK, fecha\_remito datetime R, estado\_id int FK |
| detalle\_remitos | detalle\_remito\_id int PK R AUTO\_INCREMENT, remito\_id int FK R, producto\_id int FK R, cantidad int R |
| facturas\_compras | factura\_compra\_id int PK R AUTO\_INCREMENT, orden\_compra\_id int FK, proveedor\_id int FK, fecha\_factura datetime R, tipo\_factura char(1) , numero\_factura int , total decimal |
| detalle\_facturas\_compras | detalle\_factura\_compra\_id int PK R AUTO\_INCREMENT, factura\_compra\_id int FK R, producto\_id int FK R, cantidad int R, precio\_unitario decimal R, total\_linea decimal STORED GENERATED |
| estado\_cotizaciones | estado\_id int PK R AUTO\_INCREMENT, estado varchar(20) R |
| condiciones\_pago\_cotizaciones | condicion\_pago\_id int PK R AUTO\_INCREMENT, condicion\_pago varchar(50) R |
| cotizaciones | cotizacion\_id int PK R AUTO\_INCREMENT, cliente\_id int FK, empleado\_id int FK, fecha\_cotizacion datetime R, validez\_hasta datetime , estado\_id int FK, precio\_total decimal , condicion\_pago\_id int FK |
| detalle\_cotizaciones | detalle\_cotizacion\_id int PK R AUTO\_INCREMENT, cotizacion\_id int FK R, producto\_id int FK R, cantidad int R, precio\_unitario decimal R, total\_linea decimal STORED GENERATED |
| cotizacion\_orden | cotizacion\_id int FK R, cotizacion\_id int PK R, orden\_compra\_id int FK R, orden\_compra\_id int PK R, ganadora tinyint |
| estado\_facturas | estado\_id int PK R AUTO\_INCREMENT, estado varchar(20) R |
| facturas | factura\_id int PK R AUTO\_INCREMENT, cliente\_id int FK, empleado\_id int FK, sucursal\_id int FK, fecha\_factura datetime R, tipo\_factura char(1) R, numero\_factura int , estado\_id int FK, total decimal , condicion\_pago\_id int FK |
| detalle\_facturas | detalle\_factura\_id int PK R AUTO\_INCREMENT, factura\_id int FK R, producto\_id int FK R, cantidad int R, descuento decimal , impuesto decimal , precio\_unitario decimal R, total\_linea decimal STORED GENERATED |
| remitos\_clientes | remito\_cliente\_id int PK R AUTO\_INCREMENT, factura\_id int FK, cliente\_id int FK, fecha\_remito datetime R |
| detalle\_remitos\_clientes | detalle\_remito\_cliente\_id int PK R AUTO\_INCREMENT, remito\_cliente\_id int FK R, producto\_id int FK R, cantidad int R |
| recibos | recibo\_id int PK R AUTO\_INCREMENT, cliente\_id int FK, empleado\_id int FK, fecha\_recibo datetime R, total decimal |
| detalle\_recibos | detalle\_recibo\_id int PK R AUTO\_INCREMENT, recibo\_id int FK R, factura\_id int FK R, monto\_cancelado decimal R |
| medios\_pagos | medio\_pago\_id int PK R AUTO\_INCREMENT, nombre varchar(50) , descripcion varchar(100) |
| pagos | pago\_id int PK R AUTO\_INCREMENT, recibo\_id int FK R, medio\_pago\_id int FK R, fecha\_pago datetime R, monto decimal R, detalles varchar(100) |
| personas | persona\_id int PK R AUTO\_INCREMENT, nombre varchar(50) R, apellido varchar(50) R, dni varchar(15) , fecha\_nacimiento datetime , sexo char(1) , email varchar(60) , telefono varchar(20) , direccion\_id int FK |
| grupo\_familiares | grupo\_familiar\_id int PK R AUTO\_INCREMENT, empleado\_id int FK R, descripcion varchar(100) |
| parentezcos | parentezco\_id int PK R AUTO\_INCREMENT, parentezco varchar(20) |
| grupo\_familiares\_personas | grupo\_familiar\_id int FK R, grupo\_familiar\_id int PK R, persona\_id int FK R, persona\_id int PK R, parentezco\_id int FK |
| empleado\_departamento\_historial | empleado\_id int FK R, empleado\_id int PK R, departamento\_id int FK R, departamento\_id int PK R, fecha\_desde datetime R, fecha\_hasta datetime |
| cursos | curso\_id int PK R AUTO\_INCREMENT, codigo varchar(10) , nombre varchar(50) R, descripcion varchar(100) , horas\_totales int |
| empleado\_curso | empleado\_id int FK R, empleado\_id int PK R, curso\_id int FK R, curso\_id int PK R, fecha\_inicio datetime R, fecha\_fin datetime , resultado int |
| diagnosticos | diagnostico\_id int PK R AUTO\_INCREMENT, codigo varchar(10) , descripcion varchar(100) |
| lugares\_accidentes | lugar\_id int PK R AUTO\_INCREMENT, lugar varchar(50) |
| accidentes\_laborales | accidente\_id int PK R AUTO\_INCREMENT, empleado\_id int FK R, fecha\_accidente datetime R, descripcion varchar(100) , gravedad int , lugar\_id int FK, dias\_incapacidad int , diagnostico\_id int FK |
| tratamientos | tratamiento\_id int PK R AUTO\_INCREMENT, diagnostico\_id int FK, empleado\_id int FK, fecha\_inicio datetime , fecha\_fin datetime , descripcion varchar(100) |
| revisiones\_medicas | revision\_id int PK R AUTO\_INCREMENT, empleado\_id int FK R, fecha\_revision datetime R, observaciones varchar(100) |
| unidades\_medicas | unidad\_id int PK R AUTO\_INCREMENT, unidad varchar(10) |
| parametros\_medicos | parametro\_id int PK R AUTO\_INCREMENT, revision\_id int FK R, nombre varchar(50) R, valor int , unidad\_id int FK |
| conceptos | concepto\_id int PK R AUTO\_INCREMENT, codigo varchar(10) , nombre varchar(50) R, naturaleza tinyint R, formula varchar(50) |
| liquidaciones\_sueldo | liquidacion\_id int PK R AUTO\_INCREMENT, empleado\_id int FK R, codigo varchar(10) R, periodo int R, fecha\_pago datetime R, total\_bruto decimal , total\_deducciones decimal , total\_neto decimal , observaciones varchar(100) |
| detalle\_liquidaciones | detalle\_liquidacion\_id int PK R AUTO\_INCREMENT, liquidacion\_id int FK R, concepto\_id int FK R, cantidad int R, monto\_unitario decimal R, monto\_total decimal STORED GENERATED |
| cliente\_rubro | cliente\_id int FK R, cliente\_id int PK R, rubro\_id int FK R, rubro\_id int PK R |
| ranking\_cliente | cliente\_id int FK R, cliente\_id int PK R, ranking int , comentarios varchar(100) , fecha\_ranking date |

# 3. Diagrama entidad-relación (DER)

El DER representa gráficamente las entidades principales y sus relaciones.

Puede verse en la siguiente página: [Diccionario y diagrama BDD](https://dbdocs.io/lucas.millan/Tfi_bdd)

# 4. Vistas

Las vistas son consultas predefinidas que simplifican el acceso a la información. Cada vista se documenta con su definición SQL y propósito.

CREATE VIEW vw\_direcciones AS

SELECT

    dir.direccion\_id,

    ciu.nombre as ciudad,

    prov.nombre as provincia,

    pais.nombre as pais,

    CONCAT(dir.calle, ' ', dir.numero,

         IF(dir.piso IS NOT NULL, CONCAT(', Piso ', dir.piso), ''),

         IF(dir.depto IS NOT NULL, CONCAT(', Depto ', dir.depto), ''),

         IF(ciu.nombre IS NOT NULL, CONCAT(', ', ciu.nombre), ''),

         IF(prov.nombre IS NOT NULL, CONCAT(', ', prov.nombre), ''),

         IF(pais.nombre IS NOT NULL, CONCAT(', ', pais.nombre), '')

  ) AS direccion\_completa

FROM direcciones dir

JOIN ciudades ciu ON dir.ciudad\_id=ciu.ciudad\_id

JOIN provincias prov ON dir.provincia\_id=prov.provincia\_id

JOIN paises pais ON dir.pais\_id=pais.pais\_id;

**vw\_direcciones** buscar hacer todos los joins necesarios al momento de consultar direcciones con id. Esto debido a que direcciones es una tabla muy consultada y siempre hay que hacer joins para ver toda su información.

Además se agrega direccion\_completa como la concatenación de toda la información de dirección con el fin de que sea más amigable la dirección al usuario que consulte.

CREATE VIEW vw\_empleados AS

SELECT

    em.nombre,

    em.apellido,

    em.dni,

    em.cuil,

    em.fecha\_nacimiento,

    em.fecha\_alta,

    em.email,

    em.telefono,

    tipoem.nombre as tipo\_empleado,

    dep.nombre as departamento,

    suc.nombre as sucursal,

    dir.ciudad,

    dir.provincia,

    dir.pais,

    dir.direccion\_completa

FROM empleados em

JOIN tipo\_empleados tipoem ON em.tipo\_empleado\_id=tipoem.tipo\_empleado\_id

JOIN departamentos dep ON em.departamento\_id=dep.departamento\_id

JOIN sucursales suc ON em.sucursal\_id=suc.sucursal\_id

JOIN vw\_direcciones dir ON em.direccion\_id=dir.direccion\_id;

**vw\_empleados** permite consultar toda la información de los empleados desde una sola vista, con mucha facilidad para filtrar por cualquier dato, sumado a que tiene todos sus joins ya generados.

CREATE VIEW vw\_facturas AS

SELECT

    fac.numero\_factura,

    fac.tipo\_factura,

    fac.fecha\_factura,

    fac.total,

    est.estado,

    cl.cuil as cuil\_cliente,

    cl.razon\_social,

    em.empleado\_id as empleado\_n°,

    em.nombre as nombre\_empleado,

    suc.nombre as sucursal,

    fac.factura\_id

FROM facturas fac

JOIN estado\_facturas est ON fac.estado\_id=est.estado\_id

JOIN clientes cl ON fac.cliente\_id=cl.cliente\_id

JOIN empleados em ON fac.empleado\_id=em.empleado\_id

JOIN sucursales suc ON fac.sucursal\_id=suc.sucursal\_id;

**vw\_facturas** busca mostrar la información básica de la factura, omitiendo detalles de compra como productos, descripciones e impuestos, mostrando solo total y datos del cliente y empleado vendedor.

CREATE VIEW libro\_haberes AS

SELECT

    liq.codigo as liquidacion\_nro,

    em.empleado\_id as nro\_empleado,

    em.dni as dni\_empleado,

    CONCAT(em.apellido, ', ', em.nombre) as nombre\_empleado,

    liq.periodo,

    liq.fecha\_pago,

    liq.total\_bruto,

    liq.total\_deducciones,

    liq.total\_neto

FROM liquidaciones\_sueldo liq

JOIN empleados em ON liq.empleado\_id=em.empleado\_id;

**libro\_haberes** permite ver las liquidaciones de sueldo de los empleados de forma completa, dando toda la información sobre estas y haciendo los joins con empleados para ver todo de forma más sencilla y permitiendo hacer consultas simples.

CREATE VIEW historial\_lugar\_laboral\_empleados AS

SELECT

    em.dni as dni\_empleado,

    CONCAT(em.apellido, ', ', em.nombre) as nombre\_empleado,

    suc.nombre as sucursal,

    dep.nombre as departamento,

    DATE\_FORMAT(his.fecha\_desde, '%d/%m/%Y') as desde,

    CASE

    WHEN his.fecha\_hasta IS NULL THEN 'ACTUAL'

    ELSE DATE\_FORMAT(his.fecha\_hasta, '%d/%m/%Y')

    END AS hasta

FROM empleado\_departamento\_historial his

JOIN empleados em ON his.empleado\_id = em.empleado\_id

JOIN departamentos dep ON his.departamento\_id = dep.departamento\_id

JOIN sucursales suc ON dep.sucursal\_id = suc.sucursal\_id;

**historial\_lugar\_laboral\_empleados** muestra la información de los lugares laborales de los empleados para la consulta sobre su historial laboral en la empresa y saber a que sectores perteneció.

CREATE VIEW vw\_empleados\_cursos AS

SELECT

    cur.nombre as nombre\_curso,

    cur.codigo as codigo\_curso,

    cur.descripcion as descripcion\_curso,

    em.dni as documento\_empleado,

    CONCAT(em.apellido, ', ', em.nombre) as nombre\_empleado,

    DATE\_FORMAT(asi.fecha\_inicio, '%d/%m/%Y') as cursa\_desde,

    CASE

    WHEN asi.fecha\_fin IS NULL THEN 'CURSA'

    ELSE DATE\_FORMAT(asi.fecha\_fin, '%d/%m/%Y')

    END AS cursa\_hasta,

    asi.resultado as nota\_final,

    -- datos para comparaciones

    asi.fecha\_inicio as fecha\_inicio\_real,

    asi.fecha\_fin as fecha\_fin\_real

FROM empleado\_curso asi

JOIN cursos cur ON asi.curso\_id = cur.curso\_id

JOIN empleados em ON asi.empleado\_id = em.empleado\_id;

**vw\_empleados\_cursos** vista de empleados y los cursos realizados.

CREATE VIEW vw\_stock\_depositos AS

SELECT

    st.producto\_id,

    pro.nombre as nombre\_producto,

    st.almacen\_id,

    alm.nombre as nombre\_deposito,

    dir.ciudad,

    dir.provincia,

    dir.pais,

    dir.direccion\_completa,

    st.cantidad as stock\_deposito

FROM stocks st

JOIN productos pro ON st.producto\_id=pro.producto\_id

JOIN almacenes alm ON st.almacen\_id=alm.almacen\_id

JOIN vw\_direcciones dir ON alm.direccion\_id=dir.direccion\_id;

**vw\_stock\_depositos** muestra los depósitos y los productos que estos contienen junto con su stock.

CREATE VIEW vw\_revisiones\_medicas AS

SELECT

    rev.revision\_id,

    em.empleado\_id,

    em.dni as dni\_empleado,

    CONCAT(em.apellido, ', ', em.nombre) as nombre\_empleado,

    rev.fecha\_revision,

    rev.observaciones,

    pm.nombre as parametro\_medico,

    un.unidad,

    pm.valor

FROM revisiones\_medicas rev

JOIN parametros\_medicos pm ON rev.revision\_id = pm.revision\_id

JOIN unidades\_medicas un ON pm.unidad\_id = un.unidad\_id

JOIN empleados em ON rev.empleado\_id = em.empleado\_id;

**vw\_revisiones\_medicas** hace un join de las revisiones medicas con los empleados y todo lo necesario para que el personal de RRHH tenga que hacer consultas sencillas sobre esto.

CREATE VIEW vw\_personas AS

SELECT per.\*, dir.direccion\_completa

FROM personas per

JOIN vw\_direcciones dir ON per.direccion\_id = dir.direccion\_id;

**vw\_personas** se encarga de unir la tabla de personas con la direccion, para mostrar toda la información de forma completa.

CREATE VIEW vw\_clientes AS

SELECT

    cl.cliente\_id,

    cl.cuil,

    cl.razon\_social,

    cl.telefono,

    cl.email,

    cl.credito,

    cl.fecha\_alta,

    dir.ciudad,

    dir.provincia,

    dir.pais,

    dir.direccion\_completa,

    ru.nombre AS rubro,

    rk.ranking,

    rk.comentarios,

    rk.fecha\_ranking

FROM clientes cl

JOIN vw\_direcciones dir ON cl.direccion\_id = dir.direccion\_id

JOIN cliente\_rubro cr ON cl.cliente\_id = cr.cliente\_id

JOIN rubros ru ON cr.rubro\_id = ru.rubro\_id

LEFT JOIN ranking\_cliente rk ON cl.cliente\_id = rk.cliente\_id;

**vw\_clientes** unifica las tablas de clientes, direcciones, rubros y ranking para mostrar toda la información al momento de hacer facturación.

Con la unión a ranking permite además ordenar los clientes por su ranking y filtrar a partir de cualquier dato.

# 5. Triggers y procedimientos almacenados

Los triggers automatizan acciones ante eventos específicos. Se detalla su comportamiento, activadores y tablas afectadas.

“Todos los triggers tienen su version en INSERT, UPDATE y DELETE.”

Trigger: actualizar\_total\_factura  
  
Descripción: Este conjunto de triggers recalcula automáticamente el campo ‘total’ de la tabla facturas cada vez que se insertan, actualizan o eliminan ítems en la tabla detalle\_facturas.  
  
Objetivo: mantener actualizado el monto total de la factura en base a la suma de sus líneas.  
  
Trigger: actualizar\_total\_cotizacion  
  
Descripción: Actualiza el campo ‘precio\_total’ de la tabla cotizaciones cada vez que se modifica su detalle.  
  
Utilidad: asegura que el total de la cotización refleje fielmente la suma de todos los ítems que la componen.

Trigger: actualizar\_total\_orden\_compras  
  
Descripción: Recalcula automáticamente el campo ‘total’ de una orden de compra cada vez que se agrega, modifica o elimina un ítem en su detalle.  
  
Objetivo: mantener el total de cada recibo coherente con los pagos efectuados.  
  
Trigger: actualizar\_total\_recibos  
  
Descripción: Actuliza el campo ‘total’ de la tabla recibos al modificar los montos cancelados registrados en su detalle.  
  
Objetivo: mantener el total de cada recibo coherente con los pagos efectuados.  
  
Trigger: actualizar\_total\_facturas\_compras  
  
Descripción: Actualiza el campo ‘total’ de la tabla facturas\_compras cada vez que se modifica su detalle.  
  
Finalidad: reflejar de forma precisa el costo total de una factura de compra según sus líneas.  
  
Trigger: recalcular\_totales\_liquidacion  
  
Descripción: Este grupo de triggers recalcula los campos ‘total\_bruto’ , ‘total\_deducciones’ y ‘total\_neto’ de la tabla liquidaciones\_sueldo.

* Suma los conceptos de tipo remunerativo (bruto) y los conceptos de deducciones.
* Luego calcula el total\_neto restando deducciones al bruto.

Utilidad: garantiza la integridad y exactitud de los importes liquidados, considerando tanto ingresos como descuentos.

# 6. Roles

# Rol ventas: El rol Ventas está orientado a gestionar todo el ciclo comercial con el cliente, desde la elaboración de cotizaciones hasta el seguimiento del cobro. Las funciones asignadas incluyen:

Cotizaciones: Creación y administración de propuestas comerciales para clientes potenciales.

Selección de cotización ganadora: Elección de la oferta aceptada por el cliente entre múltiples alternativas presentadas.

Facturación: Generación de facturas de venta a partir de cotizaciones aceptadas o remitos entregados.

Remitos al cliente: Emisión y control de documentos que respaldan la entrega física de productos.

Cobranzas: Registro y control de los pagos recibidos de los clientes, vinculados a facturas emitidas.

Este rol tiene acceso exclusivo a las entidades y reportes vinculados al proceso de ventas, sin acceso a funciones administrativas ni financieras externas a su dominio.

**Rol stock:**  
  
El rol Stock se encarga del control de inventarios y movimientos de productos dentro de los distintos almacenes. Sus principales responsabilidades incluyen:  
  
Existencias por producto/almacén: Consulta y gestión de los niveles de stock disponibles por cada producto en los distintos depósitos.  
  
Disparadores de ajuste automático: Aplicación de reglas predefinidas para realizar ajustes automáticos de inventario ante eventos como diferencias de inventario físico o vencimientos.  
  
Reportes de valorización: Generación de informes que reflejan el valor económico del inventario, permitiendo análisis contables y de gestión.  
  
Este rol garantiza la disponibilidad de productos, la precisión del inventario y la integridad del sistema logístico interno.

**Rol compras:**  
  
El rol Compras es responsable de gestionar las adquisiciones de productos y servicios necesarios para el funcionamiento de la empresa. Sus principales funciones son:

Órdenes de compra multidepartamento: Emisión y seguimiento de órdenes de compra, permitiendo solicitudes desde distintos sectores de la empresa.

Recepción (remito): Registro de los productos entregados por proveedores mediante remitos.

Factura de compra: Ingreso de las facturas enviadas por proveedores tras la entrega de productos o servicios.

Conciliación OC–Remito–Factura: Validación cruzada de los documentos de compra para asegurar la correspondencia entre lo ordenado, recibido y facturado.

Este rol se centra en asegurar la trazabilidad de las compras, optimizar la relación con proveedores y mantener la documentación coherente y actualizada.  
  
**Rol rrhh:**  
  
El rol RRHH administra toda la información relacionada con el personal de la empresa. Está enfocado en la gestión integral del recurso humano, y sus funciones abarcan:  
  
Legajos: Administración de la información personal, contractual y administrativa de los empleados.  
  
Grupo familiar: Registro de los vínculos familiares del personal, útiles para beneficios sociales y seguros.  
  
Historial de sector: Seguimiento de los cambios de puesto, área o departamento a lo largo del tiempo.  
  
Capacitación: Planificación y registro de las actividades de formación y desarrollo del personal.  
  
Salud ocupacional: Gestión de controles médicos, evaluaciones y condiciones de salud relacionadas con el trabajo.  
  
Liquidación de sueldos: Generación y control de las liquidaciones salariales mensuales, incluyendo conceptos fijos y variables.  
  
Este rol mantiene la trazabilidad, legalidad y eficiencia en la gestión de capital humano de la organización.

# 7. Observaciones y futuras mejoras

* **Integridad referencial y validaciones:**  
  El diseño actual implementa claves foráneas y restricciones básicas, pero podrían agregarse más validaciones a nivel de aplicación y base de datos para asegurar la coherencia de los datos (por ejemplo, validaciones de rangos, formatos de email, etc.).
* **Auditoría y trazabilidad:**  
  Actualmente solo se registran fechas de creación y actualización. Futuras versiones podrían incluir tablas de auditoría para registrar cambios críticos (alta, baja, modificaciones) y usuarios responsables.
* **Gestión de usuarios y permisos:**  
  Se definieron roles y privilegios básicos. Se recomienda implementar un sistema de gestión de usuarios más granular, con autenticación y autorización a nivel de aplicación.
* **Escalabilidad y performance:**  
  El modelo es adecuado para un entorno académico o PyME, pero para grandes volúmenes de datos podrían optimizarse índices, particiones y consultas complejas.
* **Automatización y procesos:**  
  Se utilizan triggers para actualizar totales, pero podrían agregarse más automatizaciones (por ejemplo, alertas de stock mínimo, vencimientos de pagos, etc.).
* **Manejo de datos históricos:**  
  Algunas tablas (como empleados y departamentos) podrían beneficiarse de un historial más detallado de cambios y movimientos internos.
* **Internacionalización y localización:**  
  El sistema está orientado a Argentina, pero podría adaptarse para soportar múltiples países, monedas e idiomas.
* **Documentación y pruebas:**  
  Se recomienda ampliar la documentación técnica y funcional, así como implementar pruebas automatizadas para asegurar la calidad y robustez del sistema.