Proyecto BD empresa: Hotel

Gastón Bermúdez Rochietti

1º DAW

FASE 1: ANÁLISIS

Breve descripción de la empresa, indicando las principales tareas y actividades que se quieren registrar en la base de datos. Descripción de cada elemento de la base de datos (entidades, relaciones, atributos y cardinalidades). Realizar el esquema entidad-relación de la empresa

- Se realizará una base de datos para la gestión de un hotel, donde se registrarán datos sobre los clientes, las compras y los empleados, así como las facturas y albaranes, tanto los que se generan (para cobrar) como los que se reciben (para pagar).
- Entidad EMPLEADO:
    - Atributos
        - ID (PK)
        - Nombre
        - Apellido 1
        - Apellido 2
        - DNI/NIE (UQ)
        - NIE antiguo (opcional)
        - Número de seguridad social (UQ)
        - Domicilio
        - Código postal
        - Localidad
        - Teléfono (opcional) (tabla multivaluada con teléfono y descripción (empresa, personal, casa, etc),
        - Correo electrónico de empresa (opcional)
        - Puesto
    - Registra a todos los empleados
    - Relaciones:
        - SUPERVISA (1:N): reflexiva para recoger las jerarquías del personal. Un empleado puede supervisar a (0,n) empleados y es supervisado por (0,1) empleados.
        - DIRIGE (1:1): Con Departamento. Un empleado puede dirigir (0,1) departamentos y un departamento es dirigido por (1,2) empleados (el jefe y, opcionalmente, el segundo jefe).
        - PERTENECE (N:1): Con Departamento. Un empleado puede pertenecer a (1,1) departamentos y a un departamento pertenecen (1,n) empleados.
- Entidad DEPARTAMENTO:
    - Atributos
        - ID (PK)
        - Nombre departamento (UQ)
        - Presupuesto
        - Teléfonos (multivaluado: tabla normalizada con ID, descripción y número/extensión).

    - Registrará los distintos departamentos del hotel. Cada departamento tiene varios teléfonos asociados. Por ejemplo, el departamento BARES tiene asociados los teléfonos del bar piscina, restaurante, oficina y comedor.
    - Relaciones:
        - DIRIGE (1:1): Con Empleado. Un empleado puede dirigir (0,1) departamentos y un departamento es dirigido por (1,2) empleados (el jefe y, opcionalmente, el segundo jefe).
        - PERTENECE (N:1): Con Empleado. Un empleado puede pertenecer a (1,1) departamentos y a un departamento pertenecen (1,n) empleados.
        - REALIZA (1:1:N): Con Pedido y Proveedor. Un departamento puede realizar (0,n) pedidos a (1,1) proveedores. Un pedido es realizado por (1,1) departamentos.
        - INVENTARÍA (1:N): Con Artículo. Un artículo es inventariado por (1,1) departamentos y un departamento puede inventariar (0,n) artículos.
            - Si varios departamentos tienen un producto igual o similar, tendrían artículos distintos. Por ejemplo, el agua del restaurante y el agua de recepción son artículos distintos con usos distintos, aunque para el proveedor sea el mismo producto.
- Entidad PEDIDO:
    - Atributos
        - ID (PK)
        - Fecha/hora
    - Registrará los pedidos realizados por cada departamento.
    - Relaciones:
        - REALIZA (1:1:N): Con Departamento y Proveedor. Un departamento puede realizar (0,n) pedidos a (1,1) proveedores. Un pedido es realizado por (1,1) departamentos.
        - PEDIDO_AGRUPA (1:N): Con Línea de pedido. Un pedido agrupa (1,n) líneas y cada línea es agrupada en (1,1) pedido.
        - CORRESPONDE_A (N:M): Con Albarán. A cada pedido le corresponden (0,n) albaranes y a cada albarán le corresponden (1,n) pedidos.
            - Un pedido puede existir sin el albarán durante el tiempo que tarda en recibirse, pero no se puede recibir un albarán sin un pedido asociado.
            - Si se ha comprado algo sin haberlo pedido por el sistema, se tendrá que realizar el pedido correspondiente para poder registrar el albarán.
- Entidad LÍNEA DE PEDIDO (entidad débil de PEDIDO):
    - Atributos
        - Número de línea (PK)
        - Cantidad
    - Registra los artículos que se piden en cada pedido.
    - Relaciones:

        - PEDIDO_AGRUPA (1:N): Con Pedido. Un pedido agrupa (1,n) líneas y cada línea es agrupada en (1,1) pedido.
            - No se puede almacenar un pedido sin al menos una línea.
        - PEDIDO_RECOGE (1:N): Con Artículo. Una línea recoge (1,1) artículos y cada artículo es recogido en (0,n) líneas.
            - No se puede almacenar una línea sin artículo.
- Entidad ALBARÁN:
    - Atributos:
        - ID (PK)
        - Número externo (código por el que el proveedor identifica el documento)
        - Fecha albarán
        - Fecha recepción
        - Base imponible (calculado según las líneas de albarán)
        - Total impuestos (calculado según las líneas de albarán)
        - Importe total (calculado según la base imponible y el total de impuestos)
    - Relaciones:
        - CORRESPONDE_A (N:M): Con Pedido. A cada pedido le corresponden (0,n) albaranes y a cada albarán le corresponden (1,n) pedidos.
            - Un pedido puede existir sin el albarán durante el tiempo que tarda en recibirse, pero no se puede recibir un albarán sin un pedido asociado.
            - Si se ha comprado algo sin haberlo pedido por el sistema, se tendrá que realizar el pedido correspondiente para poder registrar el albarán.
        - ALBARÁN_AGRUPA (1:N): Con Línea de Albarán. Un albarán agrupa (1,n) líneas y cada línea es agrupada en (1,1) albarán.
            - No se puede almacenar un albarán sin al menos una línea.
        - ENTREGA (1:N): Con Proveedor. Cada proveedor puede entregar (0,n) albaranes y cada albarán es entregado por (1,1) proveedor.
            - Un proveedor podría trabajar solo con facturas y no con albaranes.
        - REÚNE (1:N): Con Factura de Pago. Cada factura de pago reúne (0,n) albaranes y cada albarán se recoge en (0,1) factura de pago.
            - Una factura podría estar asociada a un servicio y no a un producto, por lo que no tendría un albarán asociado.
            - Un albarán no se asociará a su factura correspondiente hasta que se reciba la factura a final de mes, por eso se permite la existencia de albaranes sin factura aunque, teóricamente, todos los albaranes necesariamente acabarán asociados a (1,1) facturas.
- Entidad LÍNEA DE ALBARÁN (entidad débil de ALBARÁN):
    - Atributos
        - Número de línea (PK)
        - Cantidad

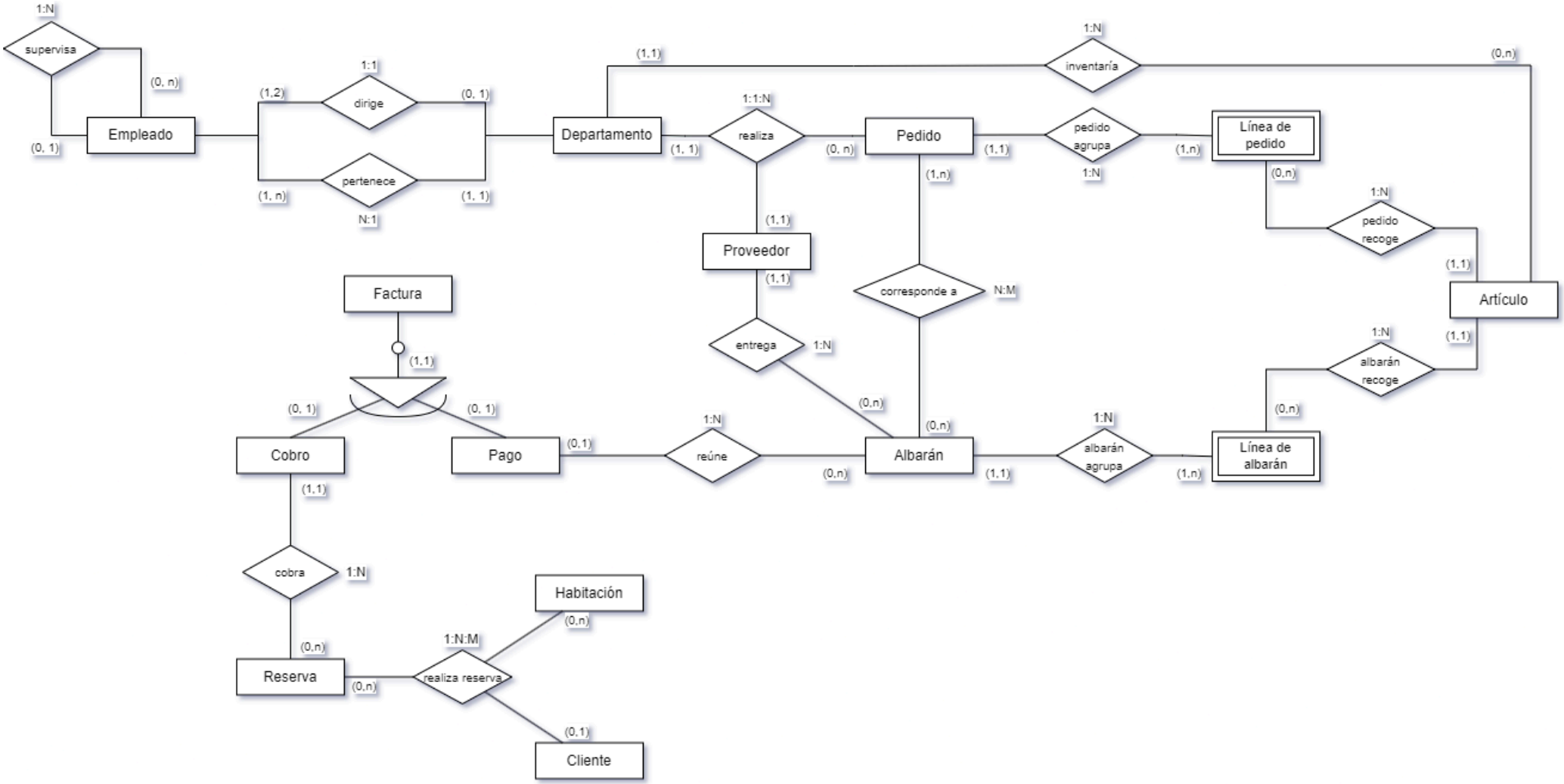
        - Precio de compra
        - Impuesto
        - Descuento (opcional)
        - Total (calculado con un trigger after insert / update).
    - Registra los artículos que se reciben con cada albarán.
    - Relaciones:
        - ALBARÁN_AGRUPA (1:N): Con Albarán. Un albarán agrupa (1,n) líneas y cada línea es agrupada en (1,1) albarán.
            - No se puede almacenar un albarán sin al menos una línea.
        - ALBARÁN_RECOGE (1:N): Con Artículo. Una línea recoge (1,1) artículos y cada artículo es recogido en (0,n) líneas.
            - No se puede almacenar una línea sin artículo.
- Entidad ARTÍCULO:
    - Atributos
        - ID (PK)
        - Nombre artículo (UQ)
    - Registra todos los artículos que puede comprar cada departamento y que se registran en el inventario del hotel.
    - Relaciones:
        - INVENTARÍA (1:N): Con Artículo. Un artículo es inventariado por (1,1) departamentos y un departamento puede inventariar (0,n) artículos.
            - Si varios departamentos tienen un producto igual o similar, tendrían artículos distintos. Por ejemplo, el agua del restaurante y el agua de recepción son artículos distintos con usos distintos, aunque para el proveedor sea el mismo producto.
        - PEDIDO_RECOGE (1:N): Con Artículo. Una línea recoge (1,1) artículos y cada artículo es recogido en (0,n) líneas.
            - No se puede almacenar una línea sin artículo.
        - ALBARÁN_RECOGE (1:N): Con Artículo. Una línea recoge (1,1) artículos y cada artículo es recogido en (0,n) líneas.
            - No se puede almacenar una línea sin artículo.
    - Incluye una vista HISTÓRICO DE PRECIOS del artículo, con datos provenientes de los albaranes con los que se ha asociado el artículo:
        - fecha de compra
        - precio
        - impuesto
        - proveedor
        - albarán en el que se registró el precio
- Entidad PROVEEDOR:
    - Atributos:
        - ID (PK)
        - Razón social
        - NIF (UQ)
        - Domicilio fiscal

        - Código postal
        - Localidad
        - Provincia
        - País
        - Teléfono/s (opcional)
        - Correo/s electrónico/s (opcional)
    - Registra todos los proveedores a los que los departamentos pueden hacer pedidos.
    - Relaciones:
        - REALIZA (1:1:N): Con Departamento y Pedido. Un departamento puede realizar (0,n) pedidos a (1,1) proveedores. Un pedido es realizado por (1,1) departamentos.
- Entidad FACTURA (supertipo):
    - Atributos generales:
        - ID (PK)
        - Fecha de factura
        - Impuestos
        - Base imponible
    - Generalización total y exclusiva.
    - Existen dos tipos de factura: las de cobro, generadas por el hotel, y las de pago, recibidas de los proveedores.
- Entidad FACTURA DE PAGO (subtipo)
    - Atributos específicos
        - Número externo (código por el que el proveedor identifica el documento)
    - Son las facturas que se pagarán a los proveedores.
    - Relaciones:
        - REÚNE (1:N): Con Albarán. Cada factura de pago reúne (0,n) albaranes y cada albarán se recoge en (0,1) factura de pago.
            - Una factura podría estar asociada a un servicio y no a un producto, por lo que no tendría un albarán asociado.
            - Un albarán no se asociará a su factura correspondiente hasta que se reciba la factura a final de mes, por eso se permite la existencia de albaranes sin factura aunque, teóricamente, todos los albaranes necesariamente acabarán asociados a (1,1) facturas.
- Entidad FACTURA DE COBRO (subtipo)
    - No tienen atributos específicos.
    - Las facturas de cobro se relacionan con los servicios ofrecidos por el hotel, principalmente la reserva de habitaciones.
    - Relaciones:
        - COBRA (1:N): Con Reserva. Una reserva se cobra en (1,1) facturas, generada automáticamente al realizar la reserva. En una factura se cobran (0,n) reservas, dejando la posibilidad de emitir facturas por otras

razones no relacionadas con las reservas (restaurante, spa, servicios extra…).

- Entidad HABITACIÓN:
  - Atributos:
    - Número (PK)
    - Tipo (suite, normal, doble, ático).
    - Vistas al mar (boolean)
    - Estado (enum: “Pendiente”, “Limpia”)
    - Ocupada (boolean)
    - Habitacion
  - Relaciones:
    - REALIZA_RESERVA (1:N:M): Ternaria con Reserva y Cliente. Una habitación puede estar asociada a (0,n) reservas. Toda reserva tiene que estar asociada a (1,n) habitaciones y a (1,1) clientes.
- Entidad RESERVA:
  - Atributos:
    - ID (PK)
    - Fecha
  - Relaciones:
    - REALIZA_RESERVA (1:N:M): Ternaria con Habitación y Cliente. Una habitación puede estar asociada a (0,n) reservas. Toda reserva tiene que estar asociada a (1,n) habitaciones y a (1,1) clientes.
- Entidad CLIENTE:
  - Atributos:
    - Número (PK)
    - NIF/NIE (UQ)
    - Nombre
    - Primer apellido
    - Segundo apellido
    - Domicilio
    - Teléfono
    - Correo electrónico (opcional)
  - Relaciones:
    - REALIZA_RESERVA (1:N:M): Ternaria con Reserva y Cliente. Una habitación puede estar asociada a (0,n) reservas. Toda reserva tiene que estar asociada a (1,n) habitaciones y a (1,1) clientes.

Diagrama entidad-relación



FASE 2: DISEÑO

Descripción de las tablas

- empleado (**ID** (PK), Nombre, Apellido1, Apellido2, NIF_NIE, NIE_Antiguo, Seg_Social, Domicilio, Localidad (FK), Puesto, Superior (FK), Departamento (FK))
- departamento (**ID** (PK), Nombre, Presupuesto, Jefe1 (FK), Jefe2 (FK))
- articulo (**ID** (PK), Nombre_articulo, Departamento (FK))
- proveedor (**ID** (PK), Razon_social, NIF, Domicilio_fiscal, Localidad (FK))
- pedido (**ID** (PK), FechaHora)
- linea_pedido (**ID** (PK), CantidadSMALL, **Pedido** (PK, FK), Articulo (FK))
- albaran (**ID** (PK), Numero_externo, Fecha_albaran, Fecha_recepcion, Base_imponible, Total_impuestos, Importe_total, Proveedor (FK), Factura (FK))
- linea_albaran (**ID** (PK), CantidadSMALL, Precio_compra, Impuesto, Descuento, Importe, **Albaran** (PK, FK), Articulo (FK))
- factura_pago (**ID** (PK), Fecha_factura, Base_imponible, Total_impuestos, Numero_externo)
- factura_cobro (**ID** (PK), Fecha_factura, Base_imponible, Total_impuestos)
- habitacion (**ID** (PK), Tipo, Vista_mar, Precio_base)
- reserva (**ID** (PK), Fecha, Factura)
- cliente (**ID** (PK), NIF_NIE, Nombre, Apellido1, Apellido2, Domicilio)

TABLAS DE RELACIONES

- realiza_pedido (**Pedido** (FK, PK), Departamento (FK), Proveedor (FK))
- corresponde (**Pedido** (FK, PK), **Albaran** (FK, PK))
- realiza_reserva (**Reserva** (FK, PK), **Habitacion** (FK, PK), Cliente (FK))

NORMALIZACIÓN

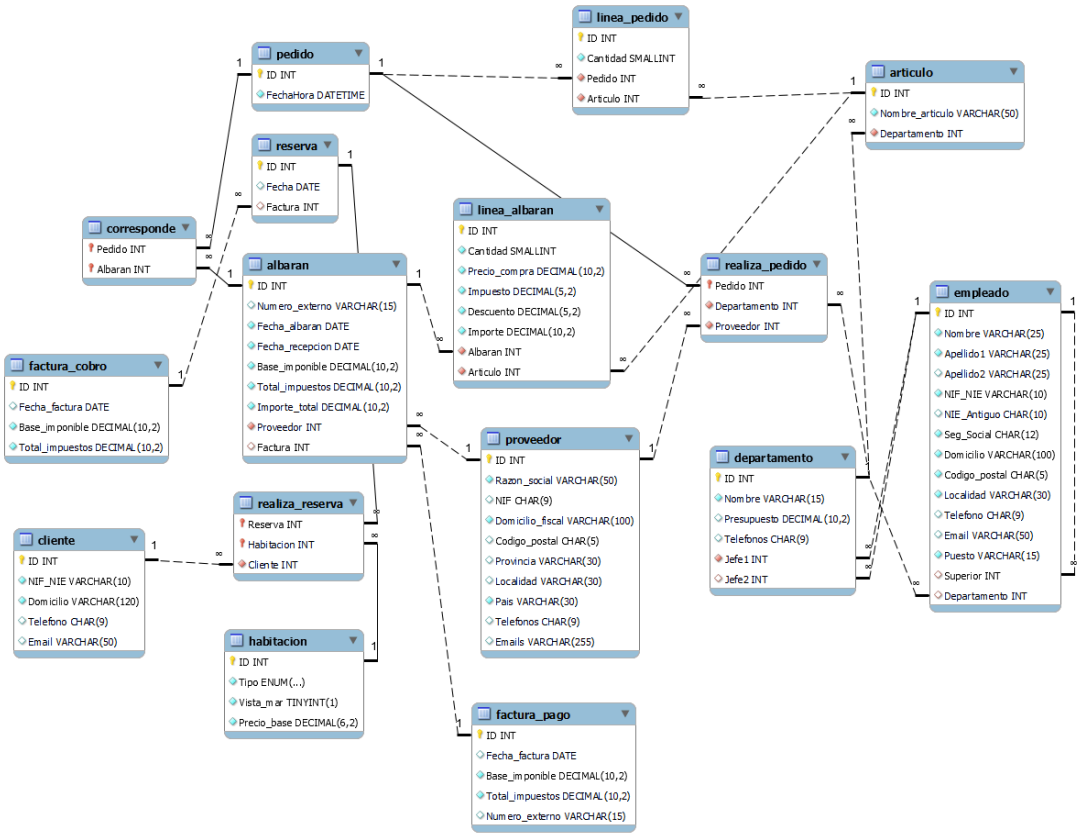
- empleado_telefono (**Empleado** (FK, PK), Telefono, Prefijo, **Descripcion** (PK))
- empleado_email (**Empleado** (FK, PK), Email, **Descripcion** (PK))
- departamento_telefono (**Departamento** (FK, PK), Telefono, Extension, **Descripcion** (PK))
- proveedor_telefono (**Proveedor** (FK, PK), Telefono, Prefijo, Extension, **Descripcion** (PK))
- proveedor_email (**Proveedor** (FK, PK), Email, **Descripcion** (PK))
- cliente_telefono (**Cliente** (FK, PK), Telefono, Prefijo, **Descripcion** (PK))
- cliente_email (**Cliente** (FK, PK), Email, **Descripcion** (PK))
- localidad (**Localidad** (PK), Codigo_postal, Provincia, Pais)

Normalización

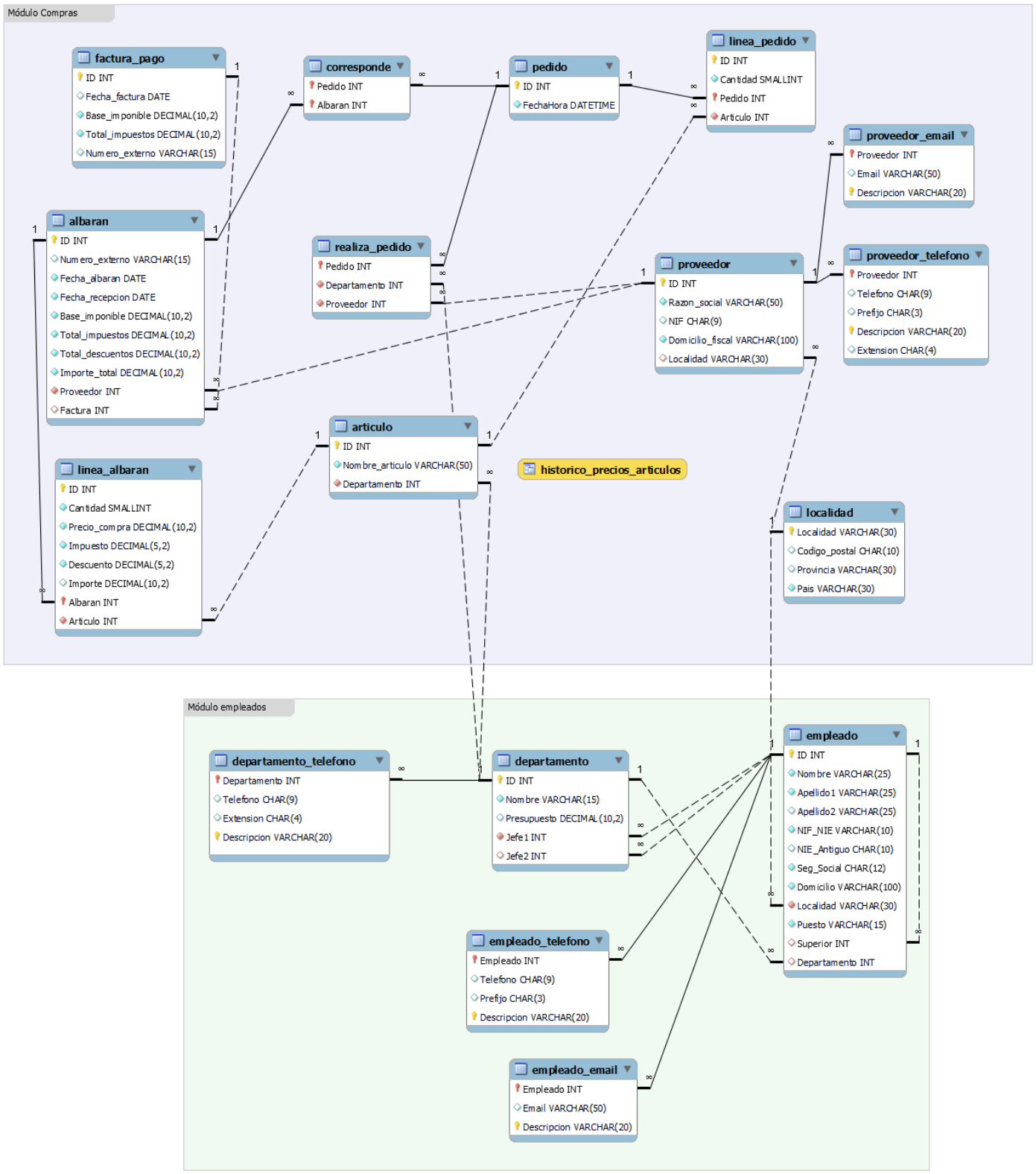
TABLA	TABLAS 1FN	TABLAS 2FN	TABLAS 3FN
empleado	Telefono Email	-	-localidad (codigo_postal, país y provincia son dependientes de localidad)
departamento	Telefono	-	-
proveedor	Telefono Email	-	-localidad (codigo_postal, país y provincia son dependientes de localidad)
cliente	Telefono Email	-	-

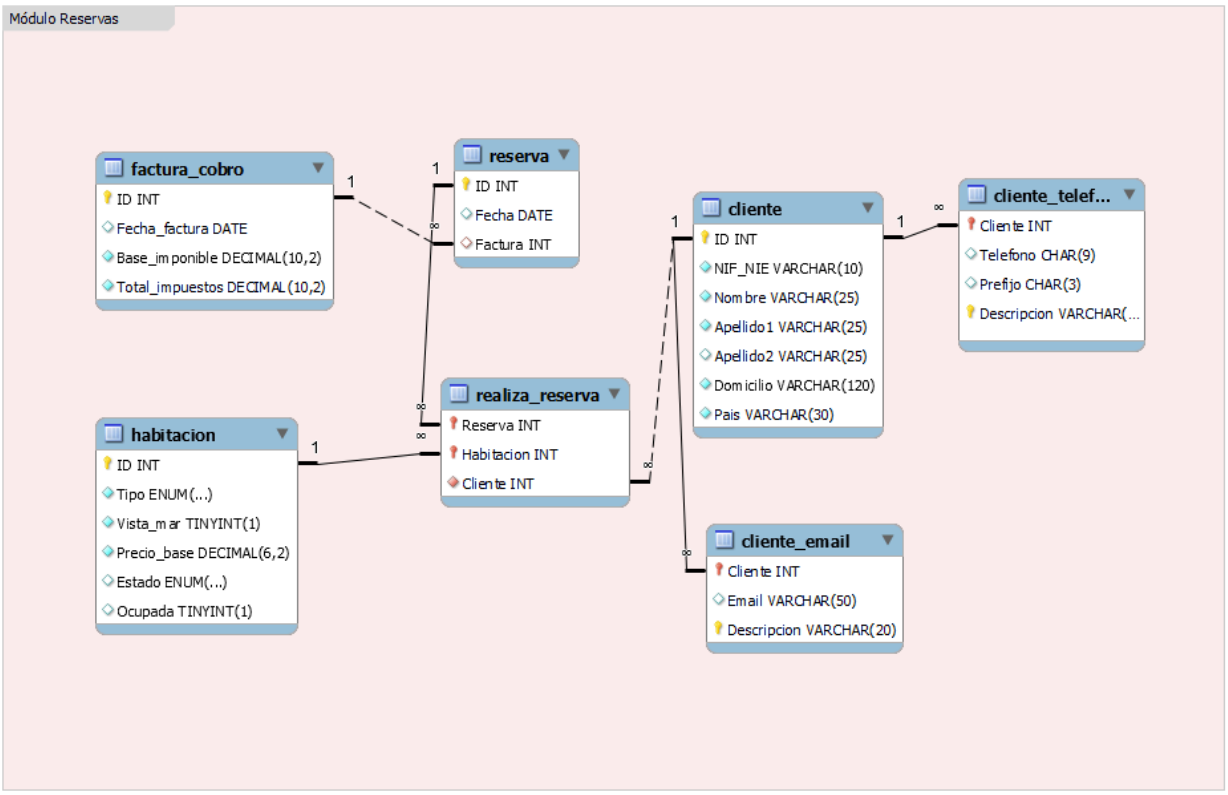
Diagramas relacionales

Sin normalizar



Normalizado





Integridad referencial

CLAVE AJENA	BORRADO/MODIFICACIÓN	JUSTIFICACIÓN
Tabla empleado FK departamento	ON DELETE RESTRICT ON UPDATE CASCADE	- Eliminar un departamento requeriría recolocar a los empleados en otros departamentos, no eliminarlos. - Al actualizar un departamento, interesa que se actualice también en la tabla empleado.
Tabla departamento FK Jefe1 / Jefe2	ON DELETE RESTRICT ON UPDATE CASCADE	- No interesa eliminar un departamento si se elimina su jefe. - Si se actualiza el jefe, sí interesa actualizar

		la tabla departamento.
Tabla empleado FK empleado (Superior)	ON DELETE RESTRICT ON UPDATE CASCADE	- Al eliminar un empleado no interesa eliminar a los empleados que tuviera a su cargo como jefe. - Sí interesa, sin embargo, actualizar las entradas de los empleados en caso de modificar el jefe.
Tabla artículo FK Departamento	ON DELETE CASCADE ON UPDATE CASCADE	- Al eliminar un departamento, los artículos asociados al mismo no sirven de nada y quedarían inutilizados en la base de datos, por lo que interesa eliminarlos. - De la misma manera, si se actualiza el departamento, también interesa actualizarlo en la tabla artículo.
Tablas linea_pedido y linea_albaran FK Pedido/Albaran FK Artículo	ON DELETE CASCADE ON UPDATE CASCADE	- Al eliminar tanto un pedido como un artículo, no tiene sentido mantener ni las líneas de pedido ni las de albarán con las que fueron asociados, por lo que interesa eliminar también esas líneas. - En caso de ser modificados, también interesa modificar las líneas de pedido / albarán.

Tabla albaran FK factura	ON DELETE RESTRICT ON UPDATE CASCADE	- Al eliminar una factura, no interesa que se eliminen los albaranes con los que se asoció, ya que podrían asociarse a otra factura. - Al actualizar la factura, sin embargo, sí que conviene actualizar sus datos en la tabla albarán.
Tabla albaran FK proveedor	ON DELETE CASCADE ON UPDATE CASCADE	- Al eliminar un proveedor, no hay razón para mantener sus albaranes. Sin embargo, no hay razón tampoco para eliminar un proveedor. - Al actualizar el proveedor, sí que conviene actualizar sus datos en la tabla albarán.
Tabla reserva FK factura	ON DELETE SET NULL ON UPDATE CASCADE	- Al eliminar una factura, no interesa eliminar la reserva asociada, solo desvincularla de la factura. Por eso se opta por dejar la entrada como NULL. - Al actualizar la factura, sin embargo, sí que interesa actualizarla en la reserva.

Tabla realiza_pedido FK Pedido	ON DELETE CASCADE ON UPDATE CASCADE	- Al eliminar un pedido interesa eliminar también su relación con el departamento y el proveedor. - Al actualizar un pedido también interesa actualizarlo en la relación.
Tabla realiza_pedido FK Departamento FK Proveedor	ON DELETE RESTRICT ON UPDATE CASCADE	- Al eliminar un departamento o un proveedor, no interesa perder la información de los mismos en los pedidos realizados. - Si se actualizan el departamento o el proveedor sí que interesa actualizarlos en la relación con los pedidos.
Tabla corresponde FK Pedido FK Albaran	ON DELETE CASCADE ON UPDATE CASCADE	- Al eliminar un pedido o un albarán ya no interesa mantener su información en la relación con el otro. - Al actualizarlos, también interesa actualizarlos en la tabla de relación.
Tabla realiza_reserva FK Reserva	ON DELETE CASCADE ON UPDATE CASCADE	- Al eliminar una reserva, no interesa mantenerla en la relación con habitación y cliente. - Al modificarla, también interesa modificarla en la relación.

Tabla realiza_reserva FK Cliente FK Habitacion	ON DELETE RESTRICT ON UPDATE CASCADE	- Al eliminar un cliente o una habitación, interesa mantener la información de las reservas que se hicieron. - Al modificar un cliente o habitación, interesa modificarlos también en la tabla de relación.
Tablas empleado_telefono, empleado_email, departamento_telefono, proveedor_telefono, proveedor_email, cliente_telefono, cliente_email FK Empleado FK Proveedor FK Cliente FK Departamento	ON DELETE CASCADE ON UPDATE CASCADE	- Al eliminar una entrada de empleado, departamento, cliente o proveedor no es necesario mantener los datos de contacto como emails o teléfonos, por lo que se eliminarían también. - Al actualizar los datos de los empleados, departamentos, proveedores o clientes también interesa actualizarlos en las tablas de sus datos de contacto.
Tablas empleado y proveedor FK Localidad	ON DELETE RESTRICT ON UPDATE CASCADE	- Al eliminar una localidad no es conveniente eliminar su información de las tablas empleado y proveedor donde se usan. - Al actualizar una localidad sí que interesa actualizarla en las tablas empleado y proveedor.

FASE 3: IMPLEMENTACIÓN

(consultar scripts adjuntos)

FASE 4: POLÍTICA DE ACCESO

Política de acceso

Para organizar las políticas de acceso a la base de datos, se han configurado diversos perfiles, basados en los distintos tipos de trabajadores que necesitarán acceso a la base de datos a través de una aplicación de gestión. Para la distribución de permisos se ha tenido en cuenta la protección de la estabilidad de la base de datos así como la protección de los datos personales de empleados y clientes. Los perfiles son los siguientes:

- Administrador del sistema:
    - Es el personal informático que se encarga del mantenimiento de la base de datos y de la aplicación de acceso.
    - Este perfil necesita un acceso completo, no solo a la base de datos del hotel, sino también a las bases de datos del sistema, como mysql o information_schema, con permisos completos.
    - Es el único perfil que puede crear usuarios de acceso a la base de datos.
- Dirección
    - El personal directivo del hotel. Tiene acceso a toda la base de datos hotel con permisos completos.
    - Es el único perfil, además del administrador del sistema, que puede otorgar permisos a otros usuarios (grant_usage), aunque no puede crear usuarios.
- Administración
    - Se encarga de la gestión de facturas, tanto de cobro como de pago, para la posterior gestión por parte del personal de contabilidad y finanzas, quien tiene acceso a las cuentas.
    - Para ello, necesita acceso a todas las tablas relacionadas con las compras, así como a la información de las reservas.
    - No tiene acceso a datos personales de clientes ni de empleados.
- Jefatura
    - Reúne a todos los jefes de los departamentos, quienes se encargan de la gestión de empleados de su departamento y de las compras.
    - Tiene acceso a todas las tablas relacionadas con las compras y a los empleados.
    - El alta de empleados corresponde a la dirección, pero el jefe sí puede modificar ciertos datos de los empleados.
- Contabilidad y finanzas:
    - Se encarga de los pagos a proveedores y la gestión de cuentas financieras.
    - En esta base de datos, solo tiene acceso a las facturas, tanto de cobro como de pago, y a las ID de los albaranes, por estar relacionados con las facturas.
- Recepción
    - El personal de recepción se encarga de la gestión de reservas y clientes. Al registrar una reserva, la recepción puede dar de alta un cliente y relacionarlo con una habitación, generando al mismo tiempo una factura de forma automática.

    - Necesita acceso a todas las tablas del módulo reservas: habitación, reserva, cliente, factura_cobro, realiza_reserva, cliente_email y cliente_telefono.
- Gobernanza
    - El personal de gobernanza se encarga de coordinar el departamento de limpieza y es el responsable de las habitaciones.
    - Para realizar su labor, necesita acceso de consulta a toda la tabla habitacion, además de acceso de modificación a la columna estado, para indicar al resto del equipo cuándo una habitación está pendiente de limpiar o lista para el acceso de clientes.
- Mantenimiento
    - El departamento de mantenimiento se encarga del estado del hotel, incluyendo las habitaciones, para lo que trabaja junto con gobernanza para saber cuándo acceder a una habitación.
    - Necesita acceso de consulta a la tabla habitación y de modificación de la columna estado.
- Restaurante
    - El departamento de restaurante se encarga del servicio en restaurante y bares y de la gestión del *room service*.
    - Necesita acceso a las tablas habitaciones, reserva, realiza_reserva y cliente para poder consultar qué habitaciones necesita visitar y qué habitaciones tienen acceso al comedor o al restaurante, además de controlar que no accedan personas sin reserva.
- Cocina
    - Para el personal de cocina se ha implementado una solución para que puedan consultar el stock de cada producto en los almacenes, además de marcar cuándo es necesario comprar más producto.
    - Necesita acceso de consulta a la tabla artículo y a la tabla departamento.

Asignación de privilegios

Para la asignación de privilegios, se han configurado roles basados en los diferentes perfiles de trabajadores. Además, con el fin de optimizar y facilitar la distribución de los permisos, se han se han creado dos “subroles” que agrupan varios privilegios comunes a varios roles:

- Gestión de compras
    - Agrupa el acceso a todas las tablas relacionadas con las compras, incluyendo los artículos, albaranes, pedidos y proveedores.
    - No incluye el permiso de creación de nuevos artículos, ya que es una tarea limitada al personal de administración.
    - Asignado a los roles jefatura y administracion.
- Gestión de empleados
    - Agrupa el acceso a las tablas relacionadas con los empleados y sus datos personales.
    - No incluye el permiso de alta de nuevos empleados, ya que es una tarea limitada a la directiva del hotel.

    - Asignado al rol jefatura.

En la siguiente tabla se recogen todos los roles junto con todos los permisos y accesos, a nivel de tabla y de columna. Se han categorizado además en distintos niveles de acceso, de mayor a menor cantidad de privilegios:

- Nivel de acceso 0
    - Administración del sistema
- Nivel de acceso 1
    - Dirección
- Nivel de acceso 2
    - Administración
    - Jefatura
- Nivel de acceso 3
    - Recepción
    - Contabilidad
- Nivel de acceso 4
    - Gobernanza
    - Cocina
    - Mantenimiento
    - Restaurante

Nivel de acceso	**Rol**	**Tabla**	**Columna**	**Permisos CRUD**
0	ROL_admin	TODAS (todas las BBDD)	TODAS	CRUD
1	ROL_direccion	TODAS (bd.hotel)	TODAS	CRUD
2	ROL_administracion	Departamento	id, nombre, jefe1, jefe2	R
2	ROL_administracion	Departamento_telefono	TODAS	R
2	ROL_administracion	empleado	id, nombre, apellido1, apellido2, puesto, superior, departamento	R
2	ROL_administracion	factura_pago	TODAS	CRUD
2	ROL_administracion	proveedor	TODAS	RU
2	ROL_administracion	proveedor_email	TODAS	CRUD

Nivel de acceso	Rol	Tabla	Columna	Permisos CRUD
2	ROL_administracion	proveedor_telefono	TODAS	CRUD
2	ROL_administracion	articulo	TODAS	CRUD
2	ROL_administracion	factura_cobro	TODAS	CRUD
2	ROL_administracion	reserva	TODAS	R
2	ROL_administracion	habitacion	id, tipo, vista_mar	RU
2	ROL_administracion	habitacion	precio_base	R
2	ROL_administracion	Realiza_reserva	TODAS	R
2	ROL_administracion	cliente	id	R
2	ROL_administracion	ROL: gestion_compras	-	-
2	ROL_jefatura	ROL: gestion_empleados	-	-
2	ROL_jefatura	ROL: gestion_compras	-	-
3	ROL_recepcion	Habitacion	id, tipo, vista_mar, precio_base	R
3	ROL_recepcion	Habitacion	estado, ocupada	RU
3	ROL_recepcion	Reserva	TODAS	CRUD
3	ROL_recepcion	Cliente	TODAS	CRUD
3	ROL_recepcion	cliente_telefono	TODAS	CRUD
3	ROL_recepcion	cliente_email	TODAS	CRUD
3	ROL_recepcion	Realiza_reserva	TODAS	CRUD
3	ROL_recepcion	factura_cobro	TODAS	CRUD
3	ROL_recepcion	Departamento	id, nombre, jefe1, jefe2	R

Nivel de acceso	Rol	Tabla	Columna	Permisos CRUD
3	ROL_recepcion	Departamento_telefono	TODAS	R
3	ROL_recepcion	empleado	id, nombre, apellido1, apellido2	R
3	ROL_contabilidad	factura_cobro	TODAS	CRUD
3	ROL_contabilidad	factura_pago	TODAS	CRUD
3	ROL_contabilidad	albaran	id	R
3	ROL_contabilidad	Departamento	TODAS	RU
3	ROL_contabilidad	Departamento_telefono	TODAS	R
4	ROL_gobernanza	Habitacion	id, tipo, vista_mar	R
4	ROL_gobernanza	Habitacion	estado, ocupada	RU
4	ROL_gobernanza	reserva	ID, fecha	R
4	ROL_gobernanza	realiza_reserva	reserva, habitacion	R
4	ROL_mantenimiento	Habitacion	id, tipo, vista_mar, ocupada	R
4	ROL_mantenimiento	Habitacion	estado	RU
4	ROL_restaurante	Habitacion	id, ocupada	R
4	ROL_restaurante	reserva	id	R
4	ROL_restaurante	cliente	id, nombre, apellido1, apellido2	R
4	ROL_restaurante	realiza_reserva	reserva, habitacion, cliente	R
4	ROL_cocina	departamento	id, nombre	R

Nivel de acceso	Rol	Tabla	Columna	Permisos CRUD
4	ROL_cocina	articulo	TODAS	R
SUBROL	ROL_gestion_compras	corresponde	TODAS	CRUD
SUBROL	ROL_gestion_compras	albaran	TODAS	CRUD
SUBROL	ROL_gestion_compras	pedido	TODAS	CRUD
SUBROL	ROL_gestion_compras	linea_pedido	TODAS	CRUD
SUBROL	ROL_gestion_compras	linea_albaran	TODAS	CRUD
SUBROL	ROL_gestion_compras	proveedor	TODAS	R
SUBROL	ROL_gestion_compras	realiza_pedido	TODAS	CRUD
SUBROL	ROL_gestion_compras	articulo	TODAS	R
SUBROL	ROL_gestion_compras	proveedor_email	TODAS	R
SUBROL	ROL_gestion_compras	proveedor_telefono	TODAS	R
SUBROL	ROL_gestion_compras	localidad	TODAS	R
SUBROL	ROL_gestion_empleados	empleado	id, nombre, apellido1, apellido2, nif_nie, superior, departamento	R
SUBROL	ROL_gestion_empleados	empleado	puesto	RU

Nivel de acceso	Rol	Tabla	Columna	Permisos CRUD
SUBROL	ROL_gestion_empleados	empleado_telefono	TODAS	RU
SUBROL	ROL_gestion_empleados	empleado_email	TODAS	RU
SUBROL	ROL_gestion_empleados	Departamento	TODAS	R
SUBROL	ROL_gestion_empleados	Departamento_telefono	TODAS	R

Revisión de permisos

Dado que existen dos roles y, potencialmente, varios usuarios con permisos “grant_usage”, es importante, para la integridad y la seguridad de la base de datos, tener un control de los permisos que se otorgan. Para ello, se proponen una serie de medidas que permitirán que los permisos no se salgan de control:

1) No reutilizar usuarios

Es importante que cada vez que un trabajador deje su puesto, su usuario se dé de baja y, en caso de ser sustituido por otro trabajador, se genere un usuario distinto o, como mínimo, se revisen sus permisos y se generen nuevas credenciales de autenticación.

2) Revisiones periódicas

Incluso aunque no haya cambios de personal, es recomendable hacer revisiones periódicas de los permisos que tienen todos los usuarios, comprobando que los permisos que tienen realmente son necesarios para su desempeño laboral. Según el diseño de cuentas de usuario descrito aquí, idealmente cada usuario tendrá solo los permisos que le otorga su rol. En caso de ser otorgados permisos extra de manera excepcional, estos se otorgarán solo de manera temporal. Si fuera necesario otorgar permanentemente otros permisos, se deberá consultar con el administrador de la base de datos para modificar los roles preestablecidos.

Las revisiones se realizarán cada 6 meses o cada vez que haya cambios de personal con acceso a la base de datos.

3) Datos especialmente protegidos o sensibles

En todo caso, la gestión de los permisos se hará teniendo siempre en cuenta qué datos son delicados y tendrán un tratamiento especial. Los usuarios que puedan consultar este tipo de datos tendrán un mayor control y revisiones, y solo se otorgarán los permisos cuando sea estrictamente necesario. Estos datos incluyen, pero no se limitan a:

- Datos personales, tanto de trabajadores como de clientes
    - Nombres y apellidos
    - Números de identidad
    - Documentación escaneada
    - Domicilios o lugares de procedencia
    - Datos de contacto como correos electrónicos o teléfonos.
- Datos financieros de la empresa
    - Cuentas
    - Balances
    - Informes
    - Contratos de servicios
- Documentación laboral

    - Contratos
    - Nóminas
    - Otros documentos personales

Script SQL

En el script adjunto se crean, en orden:

- En primer lugar, los roles, y se otorgan los privilegios correspondientes, empezando por los de mayores privilegios:
    - El rol_admin y el rol_direccion, por ser los que más privilegios tienen.
    - Los “subroles” rol_gestion_compras y rol_gestion_empleados.
    - rol_administracion
    - rol_jefatura
    - rol_recepcion
    - rol_contabilidad
    - rol_gobernanza
    - rol_mantenimiento
    - rol_restaurante
    - rol_cocina
- A continuación, se crea un usuario para cada rol (create user), se le otorga el rol correspondiente (grant role) y se le activa el rol por defecto (set default role). Para el rol jefatura, se han creado cinco usuarios distintos, uno por cada departamento: gobernanza, cocina, mantenimiento, restaurante y recepción.
- Por último, se ejecuta un “flush privileges” para que los privilegios puedan asignarse.

FASE 5: ADMINISTRACIÓN

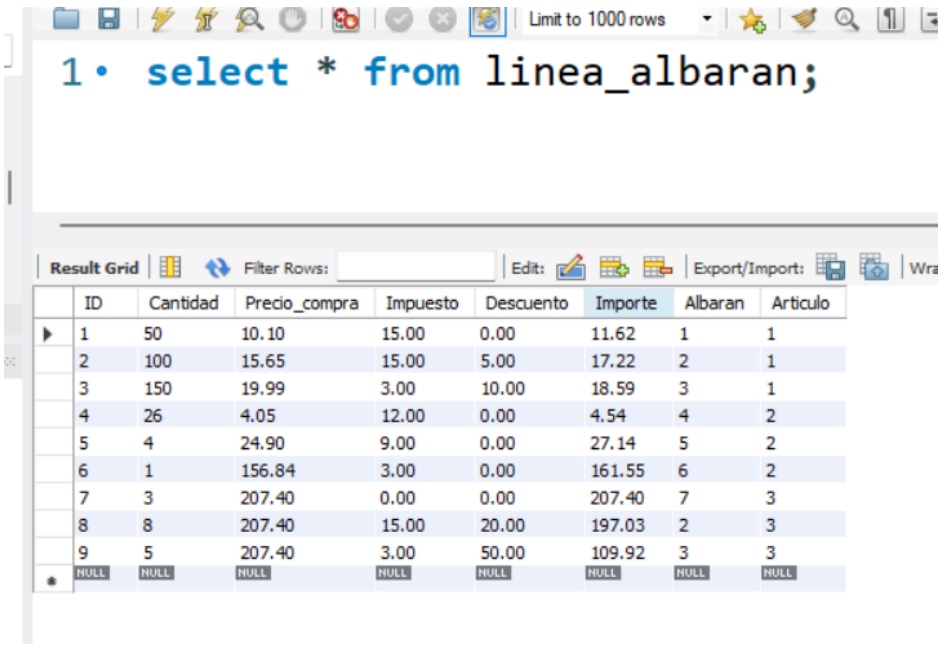
Triggers `albaran_calcular_importe` y `albaran_actualizar_importe`

- El objetivo general de este trigger es calcular automáticamente los importes de cada línea de albarán, teniendo en cuenta los valores introducidos manualmente: el precio unitario, la cantidad de producto, el descuento aplicado y los impuestos. Una vez calculado el importe de la línea, el mismo trigger también actualiza la tabla albarán para actualizar los totales de descuentos, impuestos y total a pagar.
- Se ejecuta cada vez que se inserta o se actualiza un registro en la tabla linea_albaran, para lo que se han creado dos triggers distintos; before insert y before update.
- Esto se logra con dos sentencias:
    - Una “SET new.importe” con la que se guarda en la columna importe el cálculo del precio, sumando el precio de compra al impuesto y restando el descuento, todo ello multiplicado por la cantidad de producto.
    - Una “UPDATE albaran WHERE id = new.albaran” donde se actualiza la entrada del albarán al que corresponde la nueva línea (al que se accede mediante un WHERE con el ID del albarán de la línea). Se actualizan los campos base_imponible, total_impuestos, total_descuentos e importe_total. En el trigger BEFORE UPDATE a cada valor se le resta el valor anterior (old.valor).

```
DELIMITER $$
DROP TRIGGER if exists hotel.albaran_calcular_importe$$
CREATE TRIGGER hotel.albaran_calcular_importe
before insert on linea_albaran
for each row
begin
set new.importe = new.cantidad * (new.precio_compra - ((new.descuento/100) *
new.precio_compra) + ((new.impuesto/100) * new.precio_compra));
UPDATE albaran
SET
    base_imponible = base_imponible + new.cantidad * new.precio_compra,
    total_impuestos = total_impuestos + new.cantidad * ((new.impuesto / 100) *
new.precio_compra),
    total_descuentos = total_descuentos + new.cantidad * ((new.descuento / 100) *
new.precio_compra),
    importe_total = importe_total + new.importe
WHERE
    id = new.albaran;
end$$

DROP TRIGGER if exists hotel.albaran_actualizar_importe$$
CREATE TRIGGER hotel.albaran_actualizar_importe
before update on linea_albaran
for each row
```

```
begin
set new.importe = new.cantidad * (new.precio_compra - ((new.descuento/100) *
new.precio_compra) + ((new.impuesto/100) * new.precio_compra));
UPDATE albaran
SET
    base_imponible = base_imponible + new.cantidad * new.precio_compra - old.cantidad *
old.precio_compra,
    total_impuestos = total_impuestos + new.cantidad * ((new.impuesto / 100) *
new.precio_compra) - old.cantidad * ((old.impuesto / 100) * old.precio_compra),
    total_descuentos = total_descuentos + new.cantidad * ((new.descuento / 100) *
new.precio_compra) - old.cantidad * ((old.descuento / 100) * old.precio_compra),
    importe_total = importe_total + new.importe - old.importe
WHERE
    id = new.albaran;
end$$
```

- Tests:
    - Precondiciones: hay 9 líneas de albaranes y 7 albaranes distintos. Se creará una línea nueva en el albarán 8, cuyos valores están todos a 0.



ID	Cantidad	Precio_compra	Impuesto	Descuento	Importe	Albaran	Articulo
1	50	10.10	15.00	0.00	11.62	1	1
2	100	15.65	15.00	5.00	17.22	2	1
3	150	19.99	3.00	10.00	18.59	3	1
4	26	4.05	12.00	0.00	4.54	4	2
5	4	24.90	9.00	0.00	27.14	5	2
6	1	156.84	3.00	0.00	161.55	6	2
7	3	207.40	0.00	0.00	207.40	7	3
8	8	207.40	15.00	20.00	197.03	2	3
9	5	207.40	3.00	50.00	109.92	3	3
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
1 • select * from albaran;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

ID	Numero_externo	Fecha_albaran	Fecha_recepcion	Base_imponible	Total_impuestos	Total_descuentos	Importe_total	Proveedor	Factura
1	A123	2023-01-05	2023-01-06	10.10	1.52	0.00	11.62	1	1
2	B456	2023-01-10	2023-01-11	223.05	33.46	42.26	214.25	2	2
3	C789	2023-02-09	2023-01-16	227.39	6.82	105.70	128.51	3	3
4	D489	2023-04-11	2023-01-16	4.05	0.49	0.00	4.54	1	1
5	E945	2023-02-17	2023-01-16	24.90	2.24	0.00	27.14	2	2
6	F138	2023-09-05	2023-01-16	156.84	4.71	0.00	161.55	3	3
7	G749	2022-12-18	2023-01-16	207.40	0.00	0.00	207.40	3	3
8	A123	2024-06-13	2024-06-13	0.00	0.00	0.00	0.00	4	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

- Ejecución: se inserta una línea nueva donde se detallan manualmente la cantidad, el precio de compra, el porcentaje de impuesto aplicado y el porcentaje del descuento, así como el albarán al que corresponde y el artículo (seleccionado de la lista de artículos).

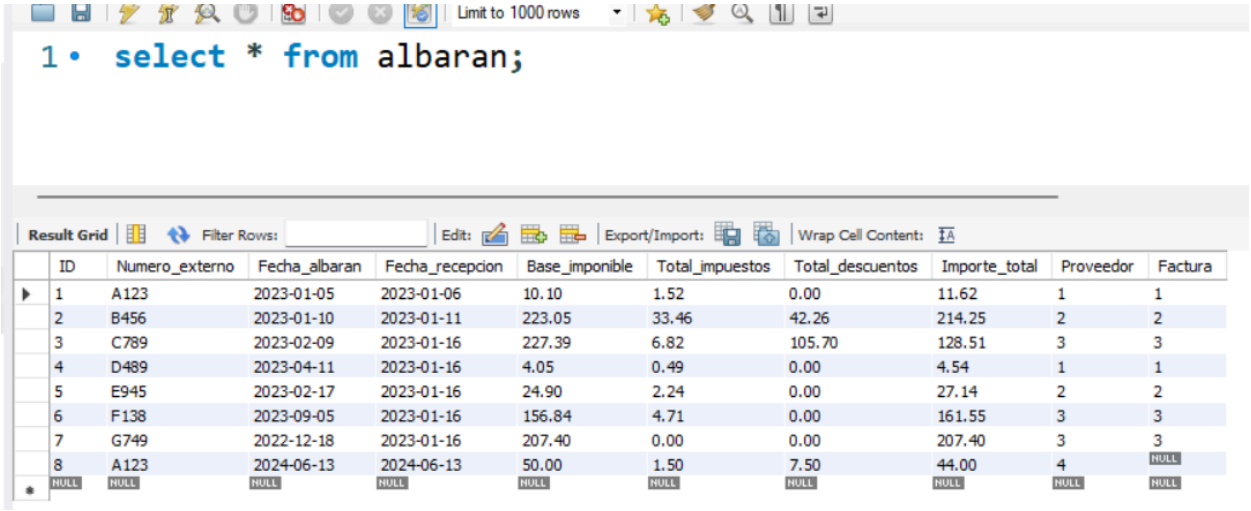
```
1 • insert into linea_albaran
2   (cantidad, precio_compra, impuesto, descuento, albaran, articulo) values
3   (5, 10.00, 3.00, 15.00, 8,
4   (select id from articulo where nombre_articulo like '%toalla%'));
```

- Postcondiciones: La columna importe de la línea y las columnas base_imponible, total_impuestos, total_descuentos e importe_total del albarán se han calculado automáticamente gracias al trigger.

```
1 • select * from linea_albaran;
```

Result Grid | Filter Rows: | Edit: | Export/Import:

ID	Cantidad	Precio_compra	Impuesto	Descuento	Importe	Albaran	Articulo
1	50	10.10	15.00	0.00	11.62	1	1
2	100	15.65	15.00	5.00	17.22	2	1
3	150	19.99	3.00	10.00	18.59	3	1
4	26	4.05	12.00	0.00	4.54	4	2
5	4	24.90	9.00	0.00	27.14	5	2
6	1	156.84	3.00	0.00	161.55	6	2
7	3	207.40	0.00	0.00	207.40	7	3
8	8	207.40	15.00	20.00	197.03	2	3
9	5	207.40	3.00	50.00	109.92	3	3
10	5	10.00	3.00	15.00	44.00	8	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL



```
1 • select * from albaran;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

ID	Numero_externo	Fecha_albaran	Fecha_recepcion	Base_imponible	Total_impuestos	Total_descuentos	Importe_total	Proveedor	Factura
1	A123	2023-01-05	2023-01-06	10.10	1.52	0.00	11.62	1	1
2	B456	2023-01-10	2023-01-11	223.05	33.46	42.26	214.25	2	2
3	C789	2023-02-09	2023-01-16	227.39	6.82	105.70	128.51	3	3
4	D489	2023-04-11	2023-01-16	4.05	0.49	0.00	4.54	1	1
5	E945	2023-02-17	2023-01-16	24.90	2.24	0.00	27.14	2	2
6	F138	2023-09-05	2023-01-16	156.84	4.71	0.00	161.55	3	3
7	G749	2022-12-18	2023-01-16	207.40	0.00	0.00	207.40	3	3
8	A123	2024-06-13	2024-06-13	50.00	1.50	7.50	44.00	4	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Procedimiento: `emails_clientes()`

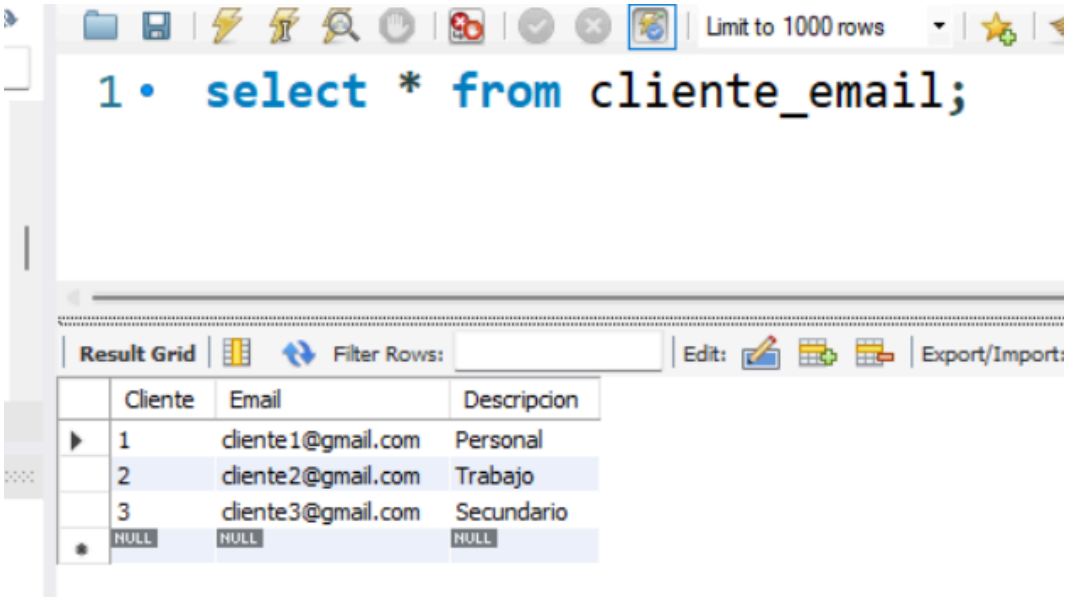
- Devuelve los emails de todos los clientes concatenados en una cadena con un texto legible, junto con el nombre completo del cliente y la descripción del email.
- Para ello, se crea un cursor que accede a la tabla cliente_email y almacena las columnas cliente, email y descripcion en las variables vcliente, vemail y vdescripcion, respectivamente.
- Se crea, además, una variable flag para detener el bucle y un handler que activa la flag cuando al no encontrar más registros.
- El formato de la cadena devuelta es “Email ‘*descripcion*’ del cliente *Nombre Apellido Apellido*: *email*”. Por ejemplo: “Email ‘personal’ del cliente Gastón Bermúdez Rochietti: gaston@email.com”.
- A cada correo le corresponde una frase, y todas van separadas por tres barras ///. La línea “set cadena = substring(cadena, 6, length(cadena))” elimina las tres primeras barras que se generan al principio de la cadena final.

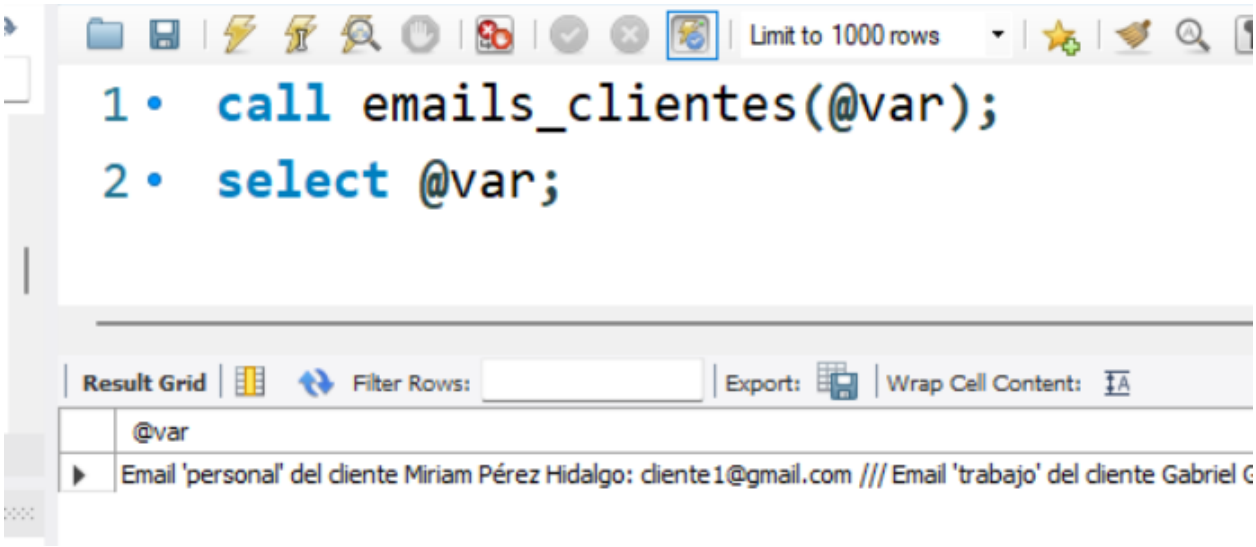
```
DROP PROCEDURE IF EXISTS hotel.emails_clientes$$
CREATE PROCEDURE hotel.emails_clientes(OUT cadena LONGTEXT)
BEGIN
    DECLARE flag BOOLEAN DEFAULT FALSE;
    DECLARE vemail VARCHAR(50);
    DECLARE vdescripcion VARCHAR(50);
    DECLARE vcliente INT;

    DECLARE cursor1 CURSOR FOR SELECT cliente, email, descripcion FROM cliente_email;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET flag = TRUE;

    SET cadena = "";
    OPEN cursor1;
```

```
    bucle: LOOP
        FETCH cursor1 INTO vcliente, vemail, vdescripcion;
        IF flag = TRUE THEN
            LEAVE bucle;
        END IF;
        SET cadena = concat(cadena, " /// Email \'",  lower(vdescripcion), "\' del cliente
", (select concat_ws(" ", nombre, apellido1, apellido2) from cliente where id = vcliente),
": ", vemail);
    END LOOP bucle;
    CLOSE cursor1;
    SET cadena = substring(cadena, 6, length(cadena));
END$$
```

- Tests: se llama al procedimiento, almacenando el resultado en la variable @var, y luego se muestra la variable en pantalla. El resultado es:

```
Email 'personal' del cliente Miriam Pérez Hidalgo: cliente1@gmail.com /// Email 'trabajo'
del cliente Gabriel Gómez Rodríguez: cliente2@gmail.com /// Email 'secundario' del cliente
José Álvarez Bermúdez: cliente3@gmail.com
```





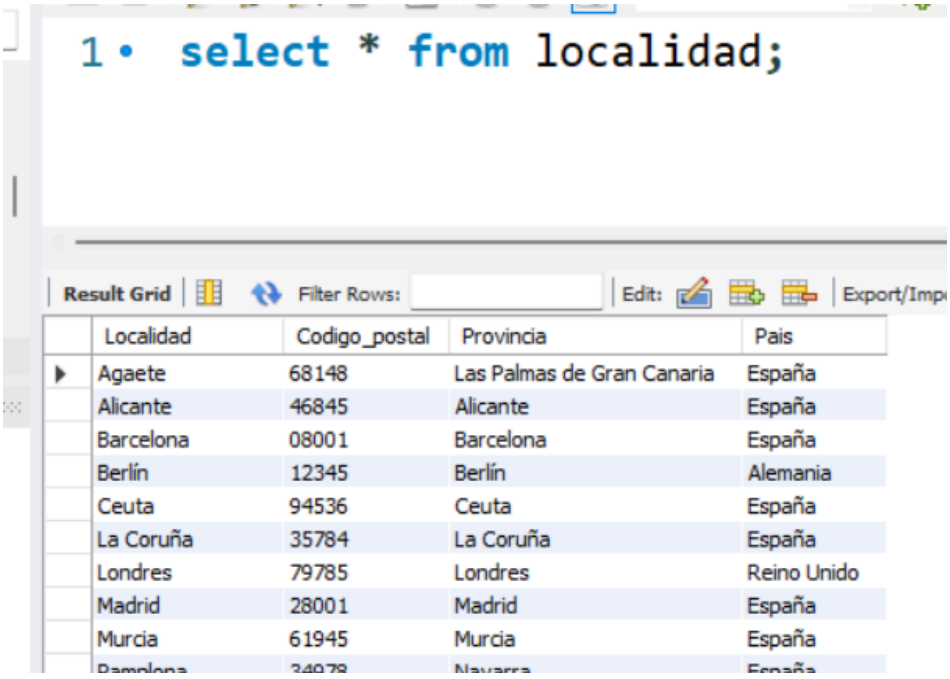
Funciones: `pais()` y `contrasenia()`

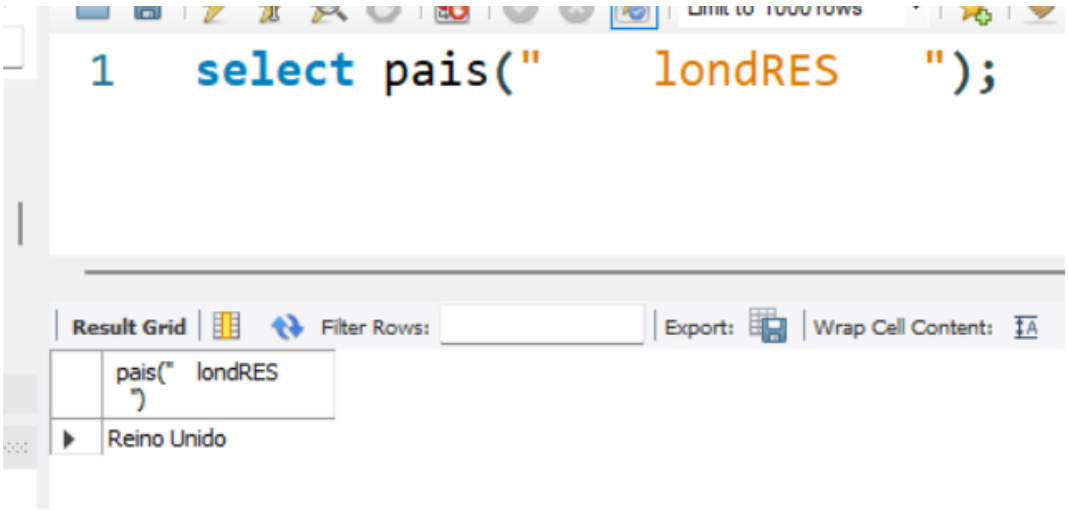
- La función `pais()` toma como parámetro un nombre de localidad o un código postal y devuelve el país donde se encuentra dicha localidad.
- Para ello, en primer lugar se crea una variable vpais, que será la que se devuelva como resultado de la función, y se recortan los espacios en blanco al comienzo y al final de la cadena de entrada con la función trim().
- A continuación, se analiza la variable de entrada, comprobando el primer carácter mediante la funcion left() y expresiones regulares.
- Se realiza un condicional: si el primer carácter es un número se trata como un código postal, por lo que se almacena en la variable vpais el resultado de la consulta a la columna localidad de la tabla localidad con un where que compara la variable de entrada con la columna codigo_postal. De lo contrario, se trata como el nombre de una localidad, por lo que el where se compara con la columna localidad, no sin antes adaptar la cadena y ponerle mayúscula inicial, para hacer coincidir el fomato con el de las localidades almacenadas, con las funciones concat(), lower(), upper(), left() y substring().
- Con la función ifnull() se logra personalizar el contenido en caso de que la consulta devuelva un NULL. En este caso se ha optado por la cadena “Localidad no registrada (NULL)”.
- Por último, se devuelve la variable vpais.

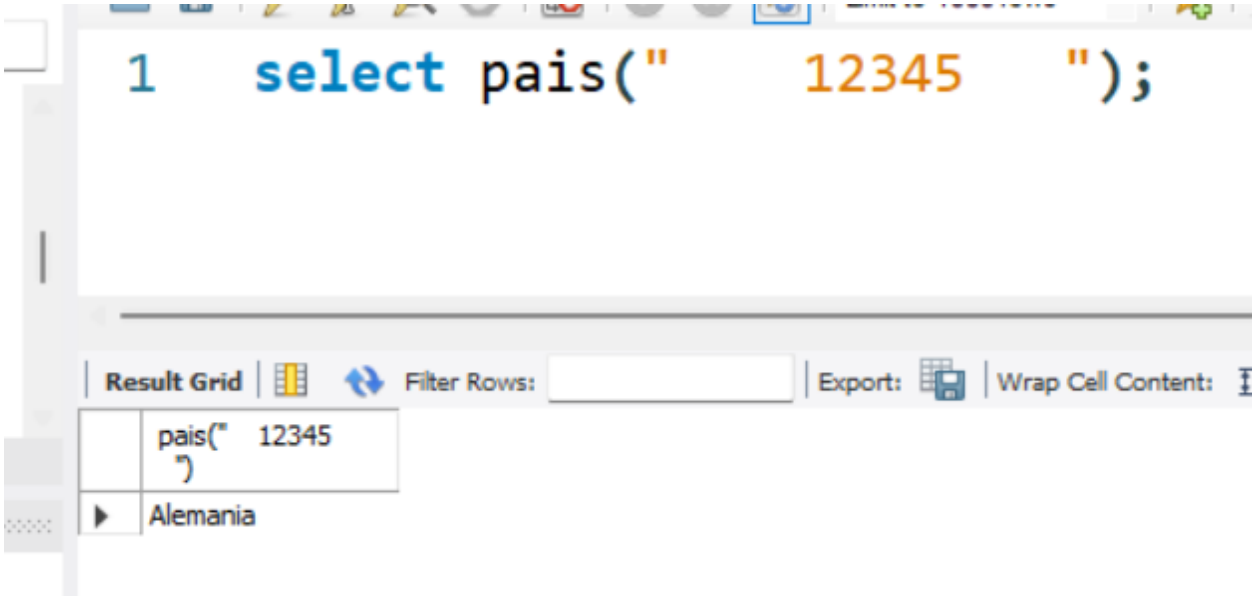
```
drop function if exists hotel.pais$$
create function hotel.pais(vlocalidad varchar(30)) returns varchar(30) not deterministic
begin
	declare vpais varchar(30) default "";
	set vlocalidad = trim(vlocalidad);
    if (left(vlocalidad, 1) rlike '[0-9]') then
```

```
            set vpais = ifnull((select pais from localidad where codigo_postal =
vlocalidad), "Localidad no registrada (NULL)");
       else
            set vlocalidad = concat(upper(left(vlocalidad, 1)),
lower(substring(vlocalidad, 2, length(vlocalidad))));
        set vpais = ifnull((select pais from localidad where localidad = vlocalidad),
"Localidad no registrada (NULL)");
       end if;
    return vpais;
end$$
```

- Tests: Al lanzar la función con el parámetro “ londRES “, se devuelve la cadena “Reino Unido”. Al lanzarla con el parámetro “ 12345 ” se devuelve “Alemania”.



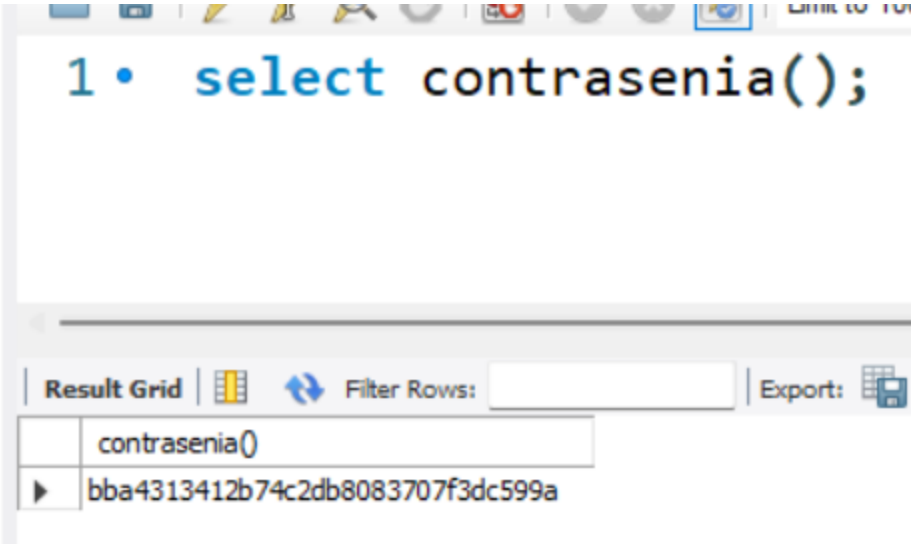




- La función `contrasenia()` crea una cadena de 32 caracteres aleatorios, apta para ser utilizada como contraseña, y la codifica mediante el algoritmo md5.
- Para ello, declara una variable vpass donde se almacenará la contraseña. Luego, entra un un bucle que, en primer lugar, guarda en vpass la contraseña generada (gracias a las funciones uuid(), que genera caracteres aleatorios y a replace(), que retira los caracteres '-' que genera uuid(). A continuación, se comprueba que esta cadena no exista en la columna authentication_string de la tabla mysql.user (codificándola antes con la función `sha2()`), y si no existe, sale del bucle. Una vez fuera del bucle, devuelve la variable vpass.
- Al acceder a la tabla de sistema user, esta función solo puede ser utilizada por un usuario administrador.

```
drop function if exists hotel.contrasenia$$
create function hotel.contrasenia() returns varchar(32) not deterministic
begin
	declare vpass varchar(32);
    bucle: loop
		set vpass = md5(replace(uuid(), "-", ""));
		if (exists (select authentication_string from mysql.user where
authentication_string = vpass)) = false then
			leave bucle;
		end if;
	end loop;
    return vpass;
end$$
```

- Test:



Evento `eliminar_pedidos_antiguos`

- Una vez al mes, revisa todos los registros de pedidos, albaranes y facturas de pago y elimina todas las entradas que tengan una antigüedad mayor a 5 años.
- Para ello, se lanzan tres sentencias “delete from”, sobre las tablas pedido, albaran y factura_pago, respectivamente.
- La sentencia contiene un where que comprueba que la diferencia en años entre la fecha actual y las columnas fechahora, fecha_albaran o fecha_factura (según la tabla) sea mayor que 5. Para ello se utilizan las funciones timestampdiff() y current_timestamp().

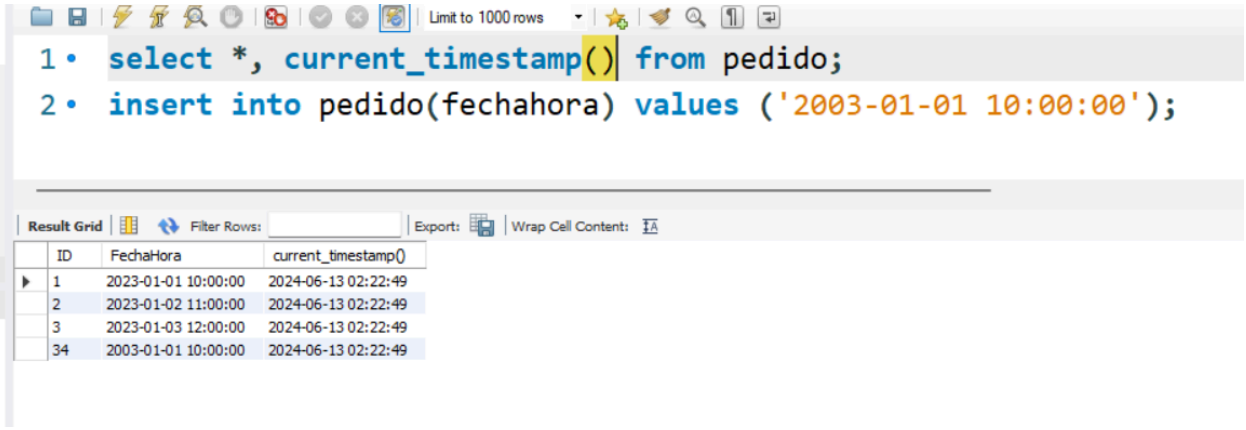
```
delimiter ;
set global event_scheduler = on;

delimiter $$
drop event if exists hotel.eliminar_pedidos_antiguos $$
create event hotel.eliminar_pedidos_antiguos
on schedule every 1 month starts current_timestamp enable do
begin
	delete from pedido where timestampdiff(year, fechahora, current_timestamp()) > 5;
	delete from albaran where timestampdiff(year, fecha_albaran, current_timestamp()) >
5;
	delete from factura_pago where timestampdiff(year, fecha_factura,
current_timestamp()) > 5;
end$$
```

- Tests:
  - Para que funcione el evento, es necesario activar en primer lugar la variable global event_scheduler. Para ello, se ejecuta la sentencia “set global event_scheduler = on;”.

- Para probar el evento, se ha cambiado la frecuencia de ejecución a cada 5 segundos.

```
delimiter $$
drop event if exists hotel.eliminar_pedidos_antiguos $$
create event hotel.eliminar_pedidos_antiguos
on schedule every 5 second starts current_timestamp enable do
begin
    delete from pedido where timestampdiff(year, fechahora, current_timestamp()) > 5;
    delete from albaran where timestampdiff(year, fecha_albaran, current_timestamp()) > 5;
    delete from factura_pago where timestampdiff(year, fecha_factura, current_timestamp()) > 5;
end$$
```

  - Se añade una fila a la tabla pedido con una fecha antigua y a las 02:22:49 se muestra el contenido de la tabla, donde se ve la nueva fila añadida.



- Pasados más de 5 segundos, se vuelve a mostrar la tabla y se ha eliminado la fila antigua.