Práctica 3

Implementación

Gestión de un hotel

UNIVERSIDAD DE GRANADA

Grupo: D1 Five Stars

Juan Andrés Mauricio Martín
Daniel Alconchel Vázquez
Luis Crespo Orti
Laura Lázaro Soraluce
Ximo Sanz Tornero

1. Reparto de subsistemas.

Gestión de actividades - Luis Crespo Orti.
Gestión de clientes - Laura Lázaro Soraluce.
Gestión de reservas - Ximo Sanz Tornero.
Gestión de personal - Juan Andrés Mauricio Martín.
Gestión de inventario - Daniel Alconchel Vázquez.

2. Descripción del sistema.

El sistema se encargará de la gestión de un negocio hotelero. Este se encuentra dividido en 5 subsistemas: gestión de actividades, gestión de clientes, gestión de reservas de habitaciones, gestión de personal y gestión de inventario del hotel. Una particularidad importante de nuestro sistema es que los clientes empleados en las reservas del hotel son distintos de los de las actividades.

Gestión de actividades

En primer lugar, vamos a crear un subsistema dentro del sistema del hotel desde el cual podemos gestionar todas las actividades que se llevan a cabo dentro del mismo.

Con el término "**actividad**" nos referimos a todo evento que se desarrolle en el ámbito del hotel, y que generalmente estará ligado a los siguientes campos:

- Un **nombre** indicativo de la naturaleza de la actividad: cadena de 50 caracteres.
- Un **lugar**/localización donde se lleva a cabo: cadena de 50 caracteres.
- Un **tipo**/rango de actividad que dictará el nivel de exclusividad de la misma, y servirá para saber qué tipo de cliente puede acceder a ella: entero entre el 0 y el 5.
- El **horario** de la actividad, que incluirá una **fecha** (dato de tipo *date*), una **hora de inicio** (dato de tipo *time*) y una **hora de finalización** (dato de tipo *time*).
- Un **número máximo de personas** que pueden participar en la actividad: dato de tipo entero positivo.
- El **documento de identidad (DNI, NIE…) del encargado** o coordinador de la actividad, el cual maneja todo lo relacionado con ella y estará a disposición de los clientes durante todo su desarrollo. Dicho encargado debe figurar como empleado, siendo gestionado desde el subsistema correspondiente. Este documento será una cadena de hasta 12 caracteres.
- Una **lista de documentos de identidad de clientes** del hotel que participarán en la misma. Puede tratarse de un DNI, NIE, pasaporte u otro. Cada uno de ellos es una cadena de hasta 12 caracteres. Los clientes se han de gestionar desde el subsistema correspondiente.
- Un **identificador de actividad** que se genera de forma automática al crear un nuevo registro de actividad y debe ser único. Será un dato de tipo numérico y autoincremental.

Algunos ejemplos son el uso del gimnasio, el spa o el restaurante; o eventos más concretos como una conferencia/reunión puntual o la proyección de una película.

· Para **registrar una nueva actividad**, el administrador correspondiente proporcionará la información anterior, siendo obligatorios únicamente el nombre de la actividad, el tipo de actividad, y la fecha y tramo horario en que se desarrolla (el resto de datos se pueden asignar luego). En cuanto al número máximo de personas que participan, si no se indica explícitamente, se fija por defecto a un valor predeterminado. El identificador único de la actividad se generará automáticamente, siendo distinto a todos los que hay en el momento. El sistema almacena los datos o da error.
· Para mostrar el **listado de inscritos en una actividad,** el administrador consulta según un identificador de actividad válido y obtiene una lista de clientes con sus respectivos nombres, apellidos, emails y teléfonos de contacto. El sistema devuelve los datos solicitados o bien da error si la lista está vacía.
· Para actualizar el **horario de una actividad**, el administrador introduce un día del calendario, una hora de inicio y una hora de finalización (la segunda debe ser posterior a la primera), junto con un identificador de actividad válido. Esta modificación da lugar a un correo que se envía a todos los inscritos a modo de notificación. El sistema confirma el cambio o devuelve un error.
· De manera similar, para actualizar el **lugar de una actividad,** el administrador introduce la localización correspondiente junto con un identificador de actividad válido, y al guardar los cambios se envía un email a los usuarios apuntados a la actividad. El sistema confirma el cambio o devuelve un error.
· Para asignar un **encargado a una actividad**, el administrador selecciona una actividad por identificador válido e indica el documento de un empleado existente. El sistema acepta la inclusión de dicho encargado o produce un error.
· Por último, para **eliminar una actividad**, basta con que el administrador proporcione el ID de la misma y desaparecerá el registro asociado en el sistema. O bien se acepta la operación o da error.

La idea es que, en fases posteriores, para cada uno de estos requisitos funcionales se construyan *menús desplegables con barra de búsqueda* con el objetivo de facilitar la entrada de información. Es por ello que se indican los requisitos de datos correspondientes. Los menús solo mostrarán las opciones que se puedan seleccionar según los requisitos semánticos, es por ello que aquellos datos que se han de leer para verificarlos se incluyen en los RDR.

Las actividades se podrán buscar por identificador y nombre; los empleados y los clientes por documento, nombre y apellidos. Los datos mostrados en los menús desplegables también se indican en los RDS. En ocasiones, los registros que se muestran vendrán filtrados por algún parámetro.

Gestión de clientes

En cuanto a la gestión de los clientes, queremos crear un subsistema de información que gestione todo lo relacionado con los usuarios del hotel, ya sean clientes que se hospedan

en él, o que simplemente hacen uso de alguno de sus servicios (canchas deportivas, actividades, restaurante…). De cada usuario almacenaremos:

- Su **ID**: cadena de hasta 12 caracteres. Único para cada cliente
- Su **nombre**: cadena de hasta 20 caracteres
- Sus **apellidos**: cadena de hasta 40 caracteres
- Su **número de teléfono**: cadena hasta 15 caracteres
- Su **correo electrónico**: cadena de hasta 50 caracteres (En formato: x@x.x - donde "x" son combinaciones de 1 o más caracteres cualesquiera)
- El **tipo** de usuario que son: entero entre 0 y 5
- El **importe** que deben: entero mayor o igual que 0
- La **lista de actividades** que tienen reservadas (de tipo ID Actividad)

Para **dar de alta** a un nuevo cliente, el usuario debe proporcionar el ID del cliente, su nombre completo, y su correo y número de teléfono, así como qué tipo de cliente son. El sistema almacenará estos datos, confirmando la inserción o dando un error.
Para **dar de baja** a un cliente, el usuario debe proporcionar el ID del cliente. El importe que deben ha de estar a cero para que se pueda llevar a cabo la baja. Se confirmará el borrado o se dará un error.
Para **cambiar el tipo** de cliente, el usuario debe proporcionar el ID del cliente y el nuevo tipo al que se quiere cambiar. El sistema confirmará la alteración o mostrará un error.
Para **reservar** una actividad, se debe proporcionar el nombre de la actividad, así como su hora y fecha, y el ID del usuario que lo quiere reservar. El sistema sólo permitirá la reserva si el usuario es del tipo permitido en el servicio. Confirmará la reserva o dará un error.
Para **liquidar los pagos** de un cliente, el usuario debe proporcionar el ID del usuario y un método de pago con suficiente crédito para cubrir el importe que debe. Se confirmará la liquidación o dará un error.
Para **consultar los servicios a los que tiene acceso** un cliente, el usuario debe proporcionar o bien el ID del usuario (si quiere consultar las actividades a las que tiene acceso un cliente específico), o bien un tipo de cliente (si quiere consultar a las que tiene acceso cualquier cliente de dicho tipo). El sistema devolverá un listado con los servicios disponibles o dará error.

El agente externo en todos los requisitos funcionales es el Usuario, que engloba tanto a los trabajadores, como administradores y clientes.

Gestión de reservas de habitaciones

Para almacenar información sobre las distintas reservas del hotel, almacenaremos un número autoincremental que sirve como identificador de la reserva, una cadena de caracteres de longitud 12 que sirve como identificador del cliente, al que llamaremos ID, tales como DNI, NIE o Pasaporte. También guardamos información del teléfono del cliente en una cadena de 15 caracteres, el correo electrónico en una cadena de 50, método de pago en una de 24 caracteres y datos asociados a la reserva como las fechas en las que es reservada con un tipo date, el número de habitación que es un número entero de 4 dígitos y el tipo de habitación que se usa, siendo esta de 1 dígito. El tipo de habitación nos informa del número de personas para las que está diseñada la habitación y si es una suite o una normal. No contemplamos que el DNI/Pasaporte sea único para cada reserva ya que una

misma persona podría reservar dos habitaciones distintas. Cabe destacar que los clientes que reservan habitaciones y los que hacen actividades son distintos. El subsistema podrá realizar las siguiente funcionalidades:

- ***Hacer una reserva:*** un administrativo debe proporcionar los datos necesarios para realizar la reserva, los citados anteriormente. El número de habitación será proporcionado por él mismo mirando las reservas anteriores y asegurándose de usar una habitación libre del tipo de habitación demandado en las fechas que se indiquen. Siendo este último quien confirme o niegue la reserva en caso de un posible error, como puede ser que no queden habitaciones de dicho tipo disponibles. Cabe destacar que el identificador es único para cada reserva.

- ***Cancelar una reserva:*** el cliente deberá proporcionar al administrativo el identificador de la reserva. Dado que el identificador exista, se deberán eliminar todos los atributos vinculados a dicha reserva. En caso de que sea incorrecto se mostrará un mensaje por pantalla informando del error. Si la reserva es eliminada correctamente también se mostrará por pantalla.

- ***Entrada del cliente:*** el administrativo recibe un cliente al hotel, (entrega de llaves), para ello comprueba la reserva del mismo y confirma si los datos son correctos. Nuevamente, el sistema indicará si la información es correcta o no según la validez de todos los datos anteriores, contrastándolos con las fechas actuales. En el caso de que no se encuentre el código identificador de la reserva se mostrará un mensaje de error. Si todo es correcto se muestra el número de habitación a ocupar y se marca un booleano asociado al check-in a true.

- ***Salida del cliente:*** el administrativo recoge las llaves de un cliente que abandona el hotel. Para ello, al igual que en la entrada usa el código identificador de la reserva y el número de habitación para comprobar que los datos son correctos y se informa del resultado de dicha acción. Con el fin de saber si una reserva ha sido utilizada se marca un booleano check out a true.

- ***Reserva de inventario para una reserva:*** el cliente puede reservar inventario libre con la cantidad de dicho inventario que desee, como pueden ser unas raquetas de pádel, y estas quedan asociadas a su reserva en el hotel. Para ello se hará una llamada al subsistema de “Gestión de inventario”, se comprueba que la diferencia entre la cantidad total del inventario y la suma la cantidad de dicho objeto asociado al resto de reservas con la cantidad demandada es positiva. Todo esto es comprobado por un administrativo y él mismo informa de si es posible la reserva del inventario. En caso de proporcionar un código de reserva erróneo al igual que en todos los otros casos, se devuelve un mensaje de error.

Gestión de personal

Con respecto a la gestión del personal deseamos crear un sistema para almacenar información acerca de los distintos empleados del hotel. De cada trabajador almacenaremos su **nombre y apellidos**, almacenados en dos cadenas de 20 y 40

caracteres respectivamente, una cadena de caracteres de longitud 12 que sirve como identificador, al que llamaremos **ID**, tales como DNI, NIE o Pasaporte. Para finalizar la información personal añadiremos la **fecha de nacimiento** como un tipo Date. Además, almacenaremos el **tipo de trabajo que realiza** (administración, mantenimiento, hostelería, etc.) en una serie de hasta 20 caracteres. Finalmente, también incluiremos la **fecha de finalización de contrato**, nuevamente como tipo Date. El subsistema podrá realizar las siguiente funcionalidades:

- ***Dar de alta a un nuevo empleado:*** el administrador debe proporcionar el nombre completo del trabajador, su número identificador, la edad, el área de trabajo y las fechas iniciales y finales de su contrato. Dichos datos serán almacenados por el sistema. Siendo este último quien confirme o niegue la inserción en caso de un posible error. Cabe destacar que el identificador es único para cada empleado, por lo que no se podrá registrar a un empleado cuyo identificador coincida con uno anterior.

- ***Dar de baja a un empleado:*** el administrador deberá proporcionar el identificador del trabajador. En caso de que el ID exista (si no existiese no se da de baja ningún trabajador), se deberán eliminar todos los atributos vinculados a dicho empleado. Será importante comprobar que al dar de baja un trabajador, en caso de estar asociado a una o varias actividades, sean eliminados en cascada. Además, igual que en el caso anterior, el sistema deberá avisar con un mensaje de afirmación o rechazo en caso de haberlo realizado con éxito o no.

- ***Modificar los datos de un empleado:*** el administrador debe proporcionar el ID del trabajador, así como todos los datos que vaya a modificar. El sistema podrá aceptar valores nulos para la no modificación de ese atributo específico. Nuevamente, el sistema indicará si el cambio se ha realizado o no según la validez de todos los datos anteriores. En el caso de que no encuentre el ID, no se producirá ningún cambio.

- ***Mostrar a los empleados del mismo área:*** el administrador debe proporcionar el área específica y el sistema devolverá la lista de empleados cuyo campo área es el indicado.

- ***Mostrar las actividades asociadas a un mismo trabajador:*** el administrador debe proporcionar el ID de un trabajador. Será el sistema quien tenga que hacer una llamada al subsistema “Gestión de Actividades” del cual tomaremos todos los atributos que se estudian. Todo esto para que más tarde podamos encontrar todas las actividades en las que aparece el trabajador especificado anteriormente.

Gestión de inventario

Finalmente, crearemos un sistema de información que registre el inventario del hotel (utensilios de cocina, utensilios de deportes, sábanas…). De cada objeto almacenaremos el ID (identificador/nombre del objeto), el cúal será el identificador del objeto (lo permite identificar de forma unívoca), que será una serie de hasta 20 caracteres; la cantidad que

disponemos de ese objeto, que será un valor numérico; categoría del objeto (si es de cocina, habitación, deportes…), que será una serie de hasta 30 caracteres y el nombre de la distribuidora que nos proporciona dichos objetos, que será una serie de hasta 40 caracteres. Las funcionalidades del subsistema serán las siguientes:

- ***Dar de alta* un nuevo objeto**: El administrador debe proporcionar el identificador (nombre) del mismo, el nombre de la distribuidora asociada, la categoría del objeto y la cantidad inicial de la que se dispone. Estos datos serán almacenados por el sistema, el cuál, debe confirmar la inserción o dar un error. Hay que tener en cuenta que cada ID es único, por lo que no se puede registrar un nuevo objeto que tenga un ID ya registrado en el sistema.

- **D*ar de baja* un objeto**, el administrador deberá proporcionar el identificador del mismo, y se eliminará del sistema el objeto con todos sus atributos asociados. Debe confirmarse el borrado o dar un error (Si se proporciona un ID no registrado en el sistema, no se da de baja ningún objeto). Además, al borrar un objeto, debe comprobar si está alquilado por algún usuario del hotel, mirando el subsistema "Gestión de reserva de habitaciones".

- **M*ostrar un listado* de los objetos por categorías** con todos los datos correspondientes. El administrador o trabajador proporciona la categoría de los objetos y se mostrarán todos los objetos registrados en el sistema con sus atributos correspondientes que pertenezcan a dicha categoría.

- ***Mostrar los datos*** de un determinado objeto, para el cuál el administrador o trabajador proporciona el identificador del objeto y el sistema nos proporciona los datos asociados al mismo (en caso de que el identificador esté registrado en el sistema) o error (si el identificador no aparece en el sistema).

- ***Cambiar los datos*** de un objeto, para lo cual, el administrador proporciona el id del objeto, así como la nueva cantidad y/o nueva distribuidora del objeto asociado y el sistema confirma el cambio o da un error (Si no encuentra el ID en el sistema no puede cambiar nada)

- **M*ostrar objetos con disponibilidad nula***, donde el sistema retorna una lista de los objetos cuya cantidad disponible asociada es igual a 0.

3. Listado de requisitos funcionales y semánticos.

Gestión de actividades

He ajustado algunos RDW para que solo incluyan los datos estrictamente necesarios.

RF 1.1: Registro de nueva actividad.

Entrada: Agente externo: administrador. Acción: solicitar inserción de nueva actividad. Requisito de datos de entrada: RDE 1.1.
BD: Requisito de datos de escritura: RDW 1.1. Requisito de datos de lectura: RDR 1.1.
Salida: Agente externo: administrador y clientes que sean inscritos en ella. Acción: confirmación del resultado, menús desplegables de empleados y clientes, y envío de correo a los clientes incluidos en la lista de inscritos. Requisito de datos de salida: RDS 1.1.

RDE 1.1: Datos de entrada para el registro de actividad.
- Relativo a la actividad:
  - ID de actividad: único.
  - Nombre: cadena de caracteres (50).
  - Lugar: cadena de caracteres (50).
  - Tipo de actividad: entero (0-5).
  - Fecha: dato de tipo *date*.
  - Horario de inicio: dato de tipo *time*.
  - Horario de finalización: dato de tipo *time*.
  - Número máximo de personas: entero.
- Relativo al encargado (opcional):
  - Documento: cadena de caracteres (12).
  - Nombre: cadena de caracteres (20).
  - Apellidos: cadena de caracteres (40).
- Relativo a el/los cliente(s) inscritos (opcional):
  - Documento: cadena de caracteres (12).
  - Nombre: cadena de caracteres (20).
  - Apellidos: cadena de caracteres (40).

RDW 1.1: Datos almacenados para nueva actividad.
- ID de actividad: único.
- Nombre: cadena de caracteres (50).
- Lugar: cadena de caracteres (50).
- Tipo de actividad: entero (0-5).
- Fecha: dato de tipo *date*.
- Horario de inicio: dato de tipo *time*.
- Horario de finalización: dato de tipo *time*.
- Número máximo de personas: entero.
- Documento del encargado: cadena de caracteres (12).
- Lista de documentos de clientes: vector/lista de cadenas de caracteres (12).

Si se incluye como entrada un documento de empleado, entonces a este se le añade el ID de la actividad en su lista de actividades asignadas.

Si se incluye como entrada una lista de documentos de clientes, entonces a cada uno de ellos se le añade el ID de la actividad a su lista de actividades reservadas.

RDR 1.1: Datos que se leen de la BD.
- Relativo a las actividades:
  - ID de actividad: único.
  - Fecha: dato de tipo *date*.

Horario de inicio: dato de tipo *time*.
Horario de finalización: dato de tipo *time*.
Documento del encargado: cadena de caracteres (12).

Relativo a los empleados:

Documento: cadena de caracteres (12).
Nombre: cadena de caracteres (20).
Apellidos: cadena de caracteres (40).
Fecha fin: dato de tipo *date*.

Relativo a los clientes:

Documento: cadena de caracteres (12).
Nombre: Cadena de caracteres (20).
Apellidos: Cadena de caracteres (40).
Correo electrónico: Cadena de caracteres (50).
Tipo de cliente: entero (0-5).

RDS 1.1: Datos que se muestran como salida.

Relativo a las actividades:

Actividad ID: único.
Nombre: cadena de caracteres (50).
Lugar: cadena de caracteres (50).
Fecha: dato de tipo *date*.
Horario de inicio: dato de tipo *time*.
Horario de finalización: dato de tipo *time*.

Relativo a los empleados:

Documento: cadena de caracteres (12).
Nombre: Cadena de caracteres (20).
Apellidos: Cadena de caracteres (40).

Relativo a los clientes:

Documento: cadena de caracteres (12).
Nombre: cadena de caracteres (20).
Apellidos: cadena de caracteres (40).
Correo electrónico: Cadena de caracteres (50).

Observaciones:

- Los RDR(s) se incluyen para mostrar menús de selección desplegables con todas las opciones disponibles y válidas, tanto para los encargados como para los clientes. En el caso de los clientes, solo se muestran aquellos que tienen acceso a la actividad según su tipo (suponiendo que ya hemos rellenado el campo de tipo de actividad); y en el caso de los encargados, solo se muestran aquellos cuya fecha de fin de contrato es anterior a la fecha de la actividad seleccionada (si es que ya la hemos indicado). Se incluyen RDS(s) para este propósito.
- Dichos menús tendrán barras de búsqueda donde se puede buscar por documento, nombre o apellidos (estos datos se incluyen, por tanto, como RDE(s)).
- En principio, suponemos que dos actividades pueden realizarse en el mismo lugar y misma fecha-horario (por ejemplo, una sesión de yoga y de crossfit pueden desarrollarse simultáneamente en el gimnasio). Tampoco vamos a considerar un

problema que dos actividades tengan el mismo nombre. Es por eso que no vamos a añadir un requisito semántico al respecto.

- Incluimos como RDR la fecha, tramo horario y documento del encargado de cada actividad, para comprobar si podemos asignar dicho encargado teniendo en cuenta si está disponible (tanto a nivel de horas diarias trabajadas como disponibilidad en ese horario concreto).
- Si se adjunta una lista de clientes, estos deben ser avisados por correo de su inscripción en la nueva actividad, por lo que se incluye el correo electrónico como RDR y RDS.
- El número máximo de personas es un campo que se añade para que se pueda gestionar la reserva de actividades desde el subsistema de clientes, y se tiene en cuenta en RS 1.10 al crear una actividad al poderse incluir clientes directamente, pero no tiene demasiado peso en este subsistema.

RF 1.2: Mostrar listado de clientes inscritos en una actividad.

Entrada: Agente externo: administrador. Acción: solicitar listado de personas apuntadas a una actividad concreta. Requisito de datos de entrada: RDE 1.2.

BD: Requisito de datos de escritura: ninguno. Requisito de datos de lectura: RDR 1.2.

Salida: Agente externo: administrador. Acción: confirmación del resultado y menú desplegable de actividades disponibles. Requisito de datos de salida: RDS 1.2.

RDE 1.2: Datos de entrada para consulta de inscritos en una actividad.

Actividad ID: único.
Nombre de actividad: cadena de caracteres (50).

RDR 1.2: Datos que se leen de la BD.

Relativo a las actividades:

Actividad ID: único.
Nombre: cadena de caracteres (50).
Lugar: cadena de caracteres (50).
Fecha: dato de tipo *date*.
Horario de inicio: dato de tipo *time*.
Horario de finalización: dato de tipo *time*.
Número máximo de personas: entero.
Lista de documentos de clientes: vector/lista de cadenas de caracteres (12).

Relativo a los clientes inscritos:

Documento: cadena de caracteres (12).
Nombre: cadena de caracteres (20).
Apellidos: cadena de caracteres (40).
Teléfono: cadena de caracteres (20).
Correo electrónico: Cadena de caracteres (50).

RDS 1.2: Datos que se muestran como salida:

Igual que RDR 1.2.

Observaciones:

- La idea será construir un menú desplegable para la selección de la actividad, en donde se muestren todas aquellas que coincidan en ID o nombre con lo escrito en una barra de búsqueda.
- El RDS se refiere tanto a la salida de datos en el menú de selección de actividades como a la lista de clientes inscritos que se muestra tras realizar la consulta.

RF 1.3: Actualización de horario de actividad.

Entrada: Agente externo: administrador. Acción: solicitar actualización del horario de una actividad existente. Requisito de datos de entrada: RDE 1.3.
BD: Requisito de datos de escritura: RDW 1.3. Requisito de datos de lectura: RDR 1.3.
Salida: Agente externo: administrador y clientes asociados a la actividad. Acción: confirmación del resultado, menú desplegable de actividades y envío de correo a los clientes asociados. Requisito de datos de salida: RDS 1.3.

RDE 1.3: Datos de entrada para la actualización del horario de actividad.
Actividad ID: único.
Nombre de actividad: cadena de caracteres (50).
Fecha: dato de tipo *date*.
Horario de inicio: dato de tipo *time*.
Horario de finalización: dato de tipo *time*.

RDW 1.3: Datos de actividad con la actualización de horario.
ID de actividad: único.
Fecha: dato de tipo *date*.
Horario de inicio: dato de tipo *time*.
Horario de finalización: dato de tipo *time*.

RDR 1.3: Datos que se leen de la BD.
Relativo a las actividades:
Actividad ID: único.
Nombre: cadena de caracteres (50).
Lugar: cadena de caracteres (50).
Fecha: dato de tipo *date*.
Horario de inicio: dato de tipo *time*.
Horario de finalización: dato de tipo *time*.
Lista de documentos de clientes: vector/lista de cadenas de caracteres (12).

Relativo a los clientes inscritos:
Documento: cadena de caracteres (12).
Nombre: cadena de caracteres (20).
Apellidos: cadena de caracteres (40).
Correo electrónico: cadena de caracteres (50).

RDS 1.3: Datos que se muestran como salida:
Igual que RDR 1.3.

Observaciones:

- El RDS se refiere tanto a la salida del menú desplegable donde podemos seleccionar actividad como a la información que mostramos en el correo de aviso a los clientes.
- Se deben leer los datos de los clientes inscritos para rescatar su correo y avisarles del cambio en la actividad. Además, también se recoge su nombre y apellidos para la personalización del correo, en el cual se les detallarán las características de la actividad cuyo horario se ha modificado.

RF 1.4: Actualización de lugar de actividad.

Entrada: Agente externo: administrador. Acción: solicitar actualización del lugar donde se desarrollará una actividad existente. Requisito de datos de entrada: RDE 1.4.
BD: Requisito de datos de escritura: RDW 1.4. Requisito de datos de lectura: RDR 1.4.
Salida: Agente externo: administrador y clientes asociados a la actividad. Acción: confirmación del resultado, menú desplegable de actividades y envío de correo a los clientes asociados. Requisito de datos de salida: RDS 1.4.

RDE 1.4: Datos de entrada para la actualización del lugar de actividad.
Actividad ID: único.
Nombre de actividad: cadena de caracteres (50).
Nuevo lugar: cadena de caracteres (50).

RDW 1.4: Datos de actividad con actualización de lugar.
ID de actividad: único.
Nuevo lugar: cadena de caracteres (50).

RDR 1.4: Datos que se leen de la BD.
Igual que RDR 1.3.

RDS 1.4: Datos que se muestran como salida:
Igual que RDS 1.3.

Observaciones:

- El RDS se refiere tanto a la salida del menú desplegable donde podemos seleccionar actividad como a la información que mostramos en el correo de aviso a los clientes.

RF 1.5: Asignar encargado a actividad.

Entrada: Agente externo: administrador. Acción: solicitar asociación de nuevo empleado a cierta actividad. Requisito de datos de entrada: RDE 1.5.
BD: Requisito de datos de escritura RDW 1.5. Requisito de datos de lectura: RDR 1.5.
Salida: Agente externo: administrador. Acción: confirmación del resultado y menús desplegables de actividades y empleados. Requisito de datos de salida: RDS 1.5.

RDE 1.5: Datos de entrada necesarios para asociar un encargado a una actividad.

Relativo a la actividad:
- Actividad ID: único.
- Nombre de actividad: cadena de caracteres (50).

Relativo al encargado:
- Documento: cadena de caracteres (12).
- Nombre: cadena de caracteres (20).
- Apellidos: cadena de caracteres (40).

RDW 1.5: Datos de actividad con actualización de encargado.

ID de actividad: único.
Documento del encargado: cadena de caracteres (12).

Se añade el ID de la actividad en la lista de actividades asignadas al empleado escogido.

RDR 1.5: Datos que se leen de la BD:

Relativo a las actividades:
- Actividad ID: único.
- Nombre: cadena de caracteres (50).
- Lugar: cadena de caracteres (50).
- Fecha: dato de tipo *date*.
- Horario de inicio: dato de tipo *time*.
- Horario de finalización: dato de tipo *time*.
- Documento del encargado: cadena de caracteres (12).

Relativo a los empleados:
- Documento: cadena de caracteres (12).
- Nombre: cadena de caracteres (20).
- Apellidos: cadena de caracteres (40).
- Fecha fin: dato de tipo *date*.

RDS 1.5: Datos que se muestran como salida:

Relativo a las actividades:
- Actividad ID: único.
- Nombre: cadena de caracteres (50).
- Lugar: cadena de caracteres (50).
- Fecha: dato de tipo *date*.
- Horario de inicio: dato de tipo *time*.
- Horario de finalización: dato de tipo *time*.
- Documento del encargado: cadena de caracteres (12).

Relativo a los empleados:
- Documento: cadena de caracteres (12).
- Nombre: cadena de caracteres (20).
- Apellidos: cadena de caracteres (40).

Observaciones:

- De nuevo consideramos sendos menús desplegables que nos permiten seleccionar un empleado disponible o una actividad, pudiendo filtrar la búsqueda, para así facilitar el proceso de asignación de encargado desde la interfaz.
- Al mostrar las actividades desde el menú, se indicará si esta actividad tiene asignado un encargado o no, y en caso positivo, se podrá ver su información (documento, nombre y apellidos).
- Aunque una actividad tenga ya asignado un encargado, se puede asignar otro, sobreescribiendo.
- En principio, los empleados no tienen una lista de actividades asignadas, ya que estas se pueden consultar fácilmente desde la BD de actividades, al haber un único encargado por cada actividad.
- Los datos que se consideran en RDR se deben leer tanto para mostrar la salida de los menús desplegables como para posibilitar el cumplimiento de las restricciones semánticas relativas a horarios de trabajadores.

RF 1.6: Eliminación de actividad.
Entrada: Agente externo: administrador. Acción: eliminar una determinada actividad de la base de datos. Requisito de datos de entrada: RDE 1.6.
BD: Requisito de datos de lectura: RDW 1.6. Requisito de datos de lectura: RDR 1.6.
Salida: Agente externo: administrador y clientes inscritos en la actividad. Acción: confirmación del resultado, menú desplegable de actividades y aviso de cancelación a los clientes por correo. Requisito de datos de salida: RDS 1.6.

RDE 1.6: Datos de entrada para la eliminación de actividad.
Actividad ID: único.
Nombre de actividad: cadena de caracteres (50).

RDW 1.6: Datos almacenados de actividad.
Igual que RDW 1.1, pero en este caso, si la actividad seleccionada tiene una lista de documentos de clientes, entonces para cada uno de ellos, borramos dicha actividad de su lista de actividades reservadas. También se elimina la actividad de la lista de actividades asignadas al empleado encargado de la misma (si lo hay).

RDR 1.6: Datos que se leen de la BD.
Igual que RDR 1.3.

RDS 1.6: Datos que se muestran como salida.
Igual que RDS 1.3.

Observaciones:
- Se incluyen los mismos requisitos de lectura y de salida que en RF 1.3 y RF 1.4 para manejar tanto los menús como el envío de correos.

RS 1.1. Un empleado no puede trabajar en dos actividades a la vez.
RF: RF 1.1.

RD(s): RDE 1.1, RDR 1.1
Descripción: “Si al crear una actividad y asignar a esta un encargado, resulta que el empleado no tiene disponible el tramo horario en que se desarrolla la actividad (o parte del mismo) en la fecha correspondiente, entonces no se permite realizar la asignación y se devuelve un error”

RS 1.2. Un empleado no puede trabajar en dos actividades a la vez.
RF: RF 1.5.
RD(s): RDE 1.5, RDR 1.5
Descripción: “Si al asignar un encargado a una actividad ya creada, resulta que dicho empleado no tiene disponible el tramo horario en que se desarrolla la actividad (o parte del mismo) en la fecha correspondiente, entonces no se permite realizar la asignación y se devuelve un error”

RS 1.3. Una actividad no puede durar más de 8 horas.
RF: RF 1.1
RD(s): RDE 1.1
Descripción: “Al crear una actividad, el tramo horario que dura no puede superar las 8 horas, ya que ningún encargado podría ocuparse de ella al superar las restricciones laborales estipuladas”

RS 1.4. Una actividad no puede durar más de 8 horas.
RF: RF 1.3
RD(s): RDE 1.3
Descripción: “Al modificar el horario de una actividad, el tramo horario que dura no puede superar las 8 horas, ya que ningún encargado podría ocuparse de ella al superar las restricciones laborales estipuladas”

RS 1.5. Un empleado no puede trabajar más de 8 horas diarias.
RF: RF 1.1
RD(s): RDE 1.1, RDR 1.1
Descripción: “Si al crear una actividad con encargado asignado, resulta que la duración de las actividades que tiene asignadas dicho empleado en ese día, junto con la de la nueva asignación, superan el límite de las 8 horas, entonces no se permite realizar la operación y se devuelve un aviso en el que se indica el convenio laboral actual.”

RS 1.6. Un empleado no puede trabajar más de 8 horas diarias.
RF: RF 1.5
RD(s): RDE 1.5, RDR 1.5
Descripción: “Si al asignar un encargado a una actividad ya creada, resulta que la duración de las actividades que tiene asignadas dicho empleado en ese día, junto con la de la nueva asignación, superan el límite de las 8 horas, entonces no se permite realizar la operación y se devuelve un aviso en el que se indica el convenio laboral actual.”

RS 1.7. Un empleado que está fuera de su periodo laboral no puede trabajar.
RF: RF 1.1
RD(s): RDE 1.1, RDR 1.1

Descripción: “Si al crear una actividad asignando encargado, este tiene una fecha de fin de contrato estrictamente anterior a la fecha de inicio de la actividad, entonces la operación resulta inválida y se debe indicar la situación mediante un aviso”.

RS 1.8. Un empleado que está fuera de su periodo laboral no puede trabajar.
RF: RF 1.5
RD(s): RDE 1.5, RDR 1.5
Descripción: “Si al asignar un encargado a una actividad existente, este tiene una fecha de fin de contrato estrictamente anterior a la fecha de inicio de la actividad, entonces la operación resulta inválida y se debe indicar la situación mediante un aviso”.

RS 1.9. Un cliente cuyo tipo de usuario del hotel es menor que el tipo de actividad, no puede participar en dicha actividad.
RF: RF 1.1
RD(s): RDE 1.1, RDR 1.1
Descripción: “Si al crear una actividad se incluyen en la lista de clientes uno o varios cuyo tipo de usuario no les permite acceder a la actividad que se está creando, por ser esta de un rango superior, se crea la actividad pero excluyendo a dichos clientes, y se genera un aviso para indicar que no pueden ser inscritos”.

RS 1.10. La lista de clientes inscritos en una actividad no puede superar el número máximo de personas indicado.
RF: RF 1.1
RD(s): RDE 1.1
Descripción: “Si al crear una actividad se incluye una lista de clientes que participarán en ella cuya longitud es mayor que el máximo de personas estipulado para la actividad, entonces no se crea el registro y se genera un aviso indicando la situación”.

Gestión de clientes

RF 2.1: Dar de alta un cliente

Entrada: *Agente externo:* Usuario. *Acción:* Solicitar inserción. *Requisitos de datos de entrada:* **RDE2.1**
BD: *Requisito de datos de escritura* **RDW2.1.**
Salida: *Agente externo:* Usuario. *Acción:* Confirmación resultado. *Requisitos de datos de salida:* Ninguno.

RDE2.1: Datos de entrada de alta de cliente
- ID Cliente: Cadena de caracteres (12)
- Nombre: Cadena de caracteres (20)
- Apellidos: Cadena de caracteres (40)
- Teléfono: Cadena de caracteres (15)
- Correo electrónico: Cadena de caracteres (50)
- Tipo Cliente: Entero
- Importe: Numérico
- Lista Actividades reservadas: ID Actividad

RDW2.1: Datos almacenados del cliente

Los mismos que **RDE2.1**

RF 2.2: Dar de baja un cliente

Entrada: *Agente externo:* Usuario. *Acción:* Solicitar borrado. *Requisitos de datos de entrada:* **RDE2.2**

BD: *Requisitos de datos de escritura* **RDW2.2.** *Requisitos de datos de lectura* **RDR2.1**

Salida: *Agente externo:* Usuario. *Acción:* Confirmación de resultado. *Requisitos de datos de salida:* Ninguno.

RDE2.2: Datos de entrada de baja de usuario

ID Cliente: Cadena de caracteres (12)

RDW2.2: Datos almacenados del cliente

Los mismos que **RDR2.1**

Relativo a las Actividades que tiene reservadas:

ID Actividad: Entero

Lista Usuarios de la Actividad (para borrar al Usuario de dicha lista): ID Usuario

RDR2.2: Datos almacenados del cliente

ID Cliente: Cadena de caracteres (12) (para comprobar la existencia del cliente que va a reservar la actividad)

Lista de Actividades Reservadas (para poder borrar al Usuario de la lista de Usuarios de dicha actividad): ID Actividad

Importe: Numérico (para comprobar el RS2.4)

RF 2.3: Reservar una actividad

Entrada: *Agente externo:* Usuario. *Acción:* Solicitar reserva del servicio. *Requisitos de datos de entrada*: **RDE2.3**

BD: *Requisitos de datos de escritura* **RDW2.3**. *Requisitos de datos de lectura* **RDR2.3**.

Salida: *Agente externo:* Usuario. *Acción:* Confirmación de resultado. *Requisito de datos de salida:* ninguno.

RDE2.3: Datos de lectura del cliente

Los mismos que **RDE2.2**

Relativo a la Actividad que se va a reservar:

ID Actividad: Entero

RDW2.3: Datos almacenados del cliente y de las actividades

Los mismos que **RDR2.2**

Relativo a la Actividad que se va a reservar:

ID Actividad: Entero

Lista Usuarios de la Actividad: ID Cliente

RDR2.3: Datos almacenados del cliente y de las actividades

ID Cliente: Cadena de caracteres (12) (para comprobar la existencia del cliente que va a reservar la actividad)

Tipo Cliente: Entero (para comprobar el RS2.1)

Importe: Numérico

Relativo a la Actividad que se va a reservar:

Nombre Actividad: cadena de caracteres
ID Actividad: Entero
Fecha Actividad: date
Hora Actividad: time
Lista Usuarios de la Actividad: ID Cliente
Número máximo usuarios actividad: Entero (para comprobar el RS2.2)
Relativo a las Actividades reservadas:
Lista Actividades Reservadas: ID Actividad (para comprobar el RS2.3)

RF 2.4: Liquidar los pagos

Entrada: *Agente externo:* Usuario. *Acción*: Solicitar la liquidación de los pagos. *Requisitos de datos de entrada:* **RDE2.4**

BD: *Requisitos de datos de lectura* **RDR2.4.** *Requisitos de datos de escritura* **RDW2.4**

Salida: *Agente externo:* Usuario. *Acción:* Confirmación de resultado. *Requisito de datos de salida:* ninguno.

RDE2.4: Datos de lectura del cliente
ID Cliente: Cadena de caracteres (12)
Método de Pago: Cadena de caracteres (24)

RDR2.4: Datos económicos del cliente
ID Cliente: Cadena de caracteres (12) (para comprobar la existencia del cliente que va a liquidar sus pagos)
Importe: Numérico

RDW2.4: Datos almacenados del cliente
ID Cliente: Cadena de caracteres (12)
Importe: Numérico.

RF 2.5: Consultar los servicios a los que tiene acceso

Entrada: *Agente externo:* Usuario. *Acción:* Solicitar lista de servicios a los que tiene acceso el cliente. *Requisitos de datos de entrada:* **RDE2.5**

BD: *Requisitos de datos de lectura* **RDR2.5**

Salida: *Agente externo:* Usuario. *Acción:* Mostrar lista de servicios. *Requisito de datos de salida:* **RDS2.5**

RDE2.5: Datos de lectura del cliente
ID Cliente: Cadena de caracteres (12)
Tipo Cliente: Entero

RDR2.5: Datos almacenados del cliente y de las actividades
ID Cliente: Cadena de caracteres (12) (para comprobar la existencia del cliente que va a consultar las actividades a las que tiene acceso)
Tipo Cliente: Entero
Relativo a las Actividades disponibles:
Lista IDs Actividades: Entero
Lista Nombres Actividades: cadena de caracteres
Tipo Actividad: Entero (entre 0 y 5)

RDS2.5: Datos de actividades disponibles
Relativo a las Actividades disponibles:

Lista Nombres Actividades: cadena de caracteres

RF 2.6: Cambiar el tipo de un cliente

Entrada: *Agente externo:* Administrador. *Acción:* Solicitar cambio de tipo del cliente. *Requisitos de datos de entrada:* **RDE2.6**

BD: *Requisitos de datos de lectura* **RDR2.6**. *Requisitos de datos de escritura* **RDW2.6**

Salida: *Agente externo:* Usuario. *Acción:* Confirmación de resultado. *Requisito de datos de salida:* ninguno

RDE2.6: Datos de lectura del cliente

Los mismos que en **RDE2.5**

RDW2.6: Datos almacenados del cliente

Los mismos que en **RDE2.5**

RDR2.6: Datos relativos al tipo actual del cliente

Los mismos que en **RDE2.5**

RS 2.1: Un cliente sólo puede reservar una actividad que su tipo le permita

RF: RF 2.3

RD: RDE2.3, RDR2.3

Descripción: Si el Tipo Cliente asociado al ID Cliente que se ha introducido no incluye el servicio que se solicita, no se reserva el servicio y se devuelve un error.

RS 2.2: Un cliente sólo puede reservar una actividad que tenga hueco disponible

RF: RF 2.3

RD: RDE2.3, RDR2.3

Descripción: Si la Lista de Usuarios asociada a la Actividad que se ha introducido tiene el tamaño máximo permitido en dicha actividad, no se realiza la reserva y se devuelve un error.

RS 2.3: Un cliente no puede reservar una actividad que le coincida en fecha y hora con otra que ya tiene reservada

RF: RF 2.3

RD: RDE2.3, RDR2.3

Descripción: Si la Lista de Actividades Reservadas asociada al ID Cliente que se ha introducido tiene una actividad cuya fecha y hora coincide con las introducidas, no se realiza la reserva y se devuelve un error.

RS 2.4: Un cliente no puede darse de baja si el importe que debe es mayor que 0

RF: RF 2.2

RD: RDE2.2, RDR2.2

Descripción: Si el importe asociado al ID Cliente introducido es mayor que 0, no se realiza el borrado del cliente y se devuelve un error.

Gestión de reservas de habitaciones.

Descripción: No se puede salir de una reserva antes de entrar

RF 3.1: Hacer una reserva

Entrada: *Agente externo:* Administrativo. *Acción*: Solicitar la inserción de una reserva en el sistema del Hotel. Req*uisitos de datos de entrada:* **RDE3.1**

BD: *Requisitos de datos de escritura* **RDW3.1**

Salida: *Agente externo:* Recepcionista. *Acción:* Confirmación de resultado. *Requisito de datos de salida:* **RDS3.1**

RDE3.1: Datos de lectura del cliente que hace la reserva
- DNI/Pasaporte: cadena de caracteres (12)
- Teléfono: cadena de caracteres (20)
- Correo electrónico: Cadena de caracteres (50)
- Método de pago: cadena de caracteres (24)
- Fecha de entrada: date
- Fecha de salida: date
- Tipo de habitación: número entero positivo (2)
- Código identificador de la habitación: entero positivo (4)
- Código identificador de la reserva: numérico autoincremental

RDW3.1: Datos almacenados del cliente
- Los mismos que **RDE3.1**
- Código identificador de la habitación: entero positivo (4)

RDS3.1: Datos asociados a la reserva realizada
- Código identificador de la reserva: numérico autoincremental

RF 3.2: Cancelar una reserva

Entrada: *Agente externo:* Administrativo. *Acción*: Solicitar la anulación de una reserva previamente registrada en el sistema del Hotel. Req*uisitos de datos de entrada:* **RDE3.2**

BD: *Requisitos de datos de escritura* **RDW3.2**

Salida: *Agente externo:* Recepcionista. *Acción:* Confirmación de resultado. *Requisito de datos de salida:* ninguno

RDE3.2: Datos necesarios para cancelar la reserva
- Código identificador de la reserva: numérico autoincremental

RDW3.2: Datos eliminados asociados a la reserva
- Código identificador de la reserva: cadena de caracteres(12)
- DNI/Pasaporte: cadena de caracteres (12)
- Teléfono: cadena de caracteres (20)
- Correo electrónico: Cadena de caracteres (50)
- Método de pago: cadena de caracteres (24)
- Fecha de entrada: date
- Fecha de salida: date
- Datos para eliminar la relación
- Código identificador de la habitación: entero positivo (4)

RF 3.3: Entrada del cliente

Entrada: *Agente externo:* Administrativo. *Acción*: Proporcionar una habitación al cliente que ha hecho una reserva. Requisitos de entrada*:* **RDE3.3**

BD: *Requisitos de datos de lectura* **RDR3.3** *y de escritura* **RDW3.3**

Salida: *Agente externo:* Recepcionista. *Acción:* Muestra por pantalla el código identificador de la habitación a ocupar. *Requisito de datos de salida:* **RDS3.1**

RDE3.3: Datos de lectura del cliente para entrar al hotel
- Código identificador de la reserva: cadena de caracteres(12)

RDR3.3: Datos de la reserva
- DNI/Pasaporte: cadena de caracteres (12)
- Fecha de entrada: date
- Fecha de salida: date
- Código identificador de la habitación: entero positivo (4)

RDW3.3: Asociado a la reserva
- Check-in: boolean
- Código identificador de la reserva: cadena de caracteres(12)

RDS3.3: Datos asociados a la habitación a ocupar
- Código identificador de la habitación: entero positivo (4)

RF 3.4: Salida del cliente

Entrada: *Agente externo:* Administrativo. *Acción*: Liberar la habitación de un cliente que abandone el hotel. Requisitos de entrada*:* **RDE3.4**

BD: *Requisitos de datos de de lectura* **RDR3.4** *y de escritura* **RDW3.4**

Salida: *Agente externo:* Recepcionista. *Acción:* Confirmación de que la reserva ha sido completada exitosamente. *Requisito de datos de salida:* ninguno

RDE3.4: Los mismos que **RDE3.3**

RDR3.4: ~~Los mismos que RDR3.3~~
- Datos asociados a la reserva
- Fecha de entrada: date
- Fecha de salida: date
- Check-in: boolean

RDW3.4: Datos asociados a la reserva
- Check-out: boolean
- Código identificador de la reserva: cadena de caracteres(12)

RF 3.5: Reserva de inventario para una reserva

Entrada: *Agente externo:* Administrativo. *Acción*: Solicitar la reserva de inventario que será asociado a una reserva. Req*uisitos de datos de entrada:* **RDE3.5**

BD: *Requisitos de datos de escritura* **RDW3.5** *y de lectura* **RDR3.5**

Salida: *Agente externo:* Trabajador. *Acción:* Confirmación de resultado. *Requisito de datos de salida: ninguno*

RDE3.5: Datos de la reserva
- Código identificador de la reserva: número autoincremental
- Listado del inventario de la habitación con su cantidad correspondiente

RDR3.5: Lectura del subsistema de gestión de inventario
Los mismos que **RDE5.1**
RDW3.5: Datos almacenados de la habitación
Listado del inventario de la reserva con su cantidad correspondiente

RS 3.1: Un cliente no puede hacer una reserva de una habitación que coincida, aunque sea parcialmente, en fechas con la misma habitación en otra reserva ya registrada.
RF: RF 3.1
RD: RDE3.1
Descripción: No se puede reservar una habitación en unos días en los que ya esté previamente reservada.

RS 3.2: El listado de inventario tiene que contener ID correctos del subsistema de Gestión de inventario
RF: RF 3.5
RD: RDE3.5
Descripción: No se puede reservar inventario no registrado en la base de datos.

RS 3.3: Para la salida del cliente el cliente debe de haber entrado previamente al hotel.
RF: RF 3.4
RD: RDW 3.4
Descripción: No se puede marcar check-out a true si previamente no se ha marcado check-in a true.

RS 3.4: No se puede cancelar una reserva en uso
RF: RF 3.2
RD: RDW 3.2
Descripción: No se puede cancelar una reserva con el check-in a true.

RS 3.5: La fecha de salida no puede ser anterior o igual a la de entrada
RF: RF 3.1
RD: RDE 3.1

Gestión de personal

OBJ

RF4.1: Dar de alta a un empleado.
Entrada: *Agente externo:* administrador. *Acción:* solicitar inserción.
Requisito de datos de entrada **RDE4.1**
BD: *Requisito de datos de escritura* **RDW4.1**
Salida: *Agente externo:* administrador. *Acción:* confirmación resultado.
Requisito de datos de salida: ninguno
RDE4.1: Datos para dar de alta a un trabajador.
Nombre: Cadena de caracteres(20).

Apellidos: Cadena de caracteres(40).
ID: Cadena de caracteres(12).
Fecha de nacimiento: Date.
Área: Cadena de caracteres(20).
Fecha fin de contrato: Date

RDW4.1: Datos almacenados para cada empleado.
Los mismos que **RDE4.1**

RF4.2: Dar de baja a un empleado.

Entrada: *Agente externo:* administrador. *Acción*: solicitar borrado.
Requisito de datos de entrada **RDE4.2**

BD: *Requisito de datos de escritura* **RDW4.2**

Salida: *Agente externo*: administrador. *Acción*: confirmación resultado.
Requisito de datos de salida: ninguno

RDE4.2: Datos de entrada para dar de baja a un trabajador.
ID: Cadena de caracteres(12).

RDW4.2: Datos almacenados del objeto.
Los mismos que **RDW4.1.** También se eliminará al trabajador de las actividades en las que esté asignado.

RF4.3: Modificar datos de un empleado.

Entrada: *Agente externo:* administrador. *Acción*: solicitar modificación.
Requisito de datos de entrada **RDE4.3**

BD: *Requisito de datos de escritura* **RDW4.3**

Salida: *Agente externo*: administrador. *Acción*: confirmación resultado.
Requisito de datos de salida: ninguno

RDE4.3: Datos de entrada del trabajador
Los mismos que **RDE4.1**

RDW4.3: Datos almacenados del trabajador
Los mismos que **RDE4.1**

RF4.4: Mostrar empleados de un mismo área.

Entrada: *Agente externo:* administrador o trabajador. *Acción:* solicitar listado de empleados.
Requisito de datos de entrada **RDE4.4**

BD: *Requisito de datos de lectura* **RDR4.4**

Salida: *Agente externo:* administrador. *Acción:* confirmación resultado.
Requisito de datos de salida: **RDS4.4**

RDE4.4: Datos de entrada del trabajador.
Área: Cadena de caracteres(20)

RDR4.4: Datos almacenados del trabajador.
Los mismos que **RDW4.1**

RDS4.4: Listado de registros, cada uno de ellos con los mismos datos que **RDR4.4**

RF4.5: Consultar la lista de actividades de un trabajador.
Entrada: *Agente externo:* administrador. *Acción*: solicitar horario de un trabajador.
Requisito de datos de entrada **RDE4.5**
BD: Requisito de datos de lectura **RDR4.5**
Salida: *Agente externo:* administrador. *Acción:* confirmación resultado.
Requisito de datos de salida: **RDS4.5**
RDE4.5: Datos de entrada de un trabajador.
Los mismos que **RDE4.2**
RDR4.5: Lectura del subsistema Gestión de Actividades.
Los mismos que **RDE1.1**
RDS4.5: Listado de las actividades asociadas al trabajador.

RS4.1: La edad debe ser mayor o igual a 16.
RF: RF4.1
RD(s): RDE4.1
Descripción: "No se puede registrar a empleados con una edad menor a 16"

RS4.2: La edad debe ser mayor o igual a 16.
RF: RF4.3
RD(s): RDE4.3
Descripción: "No se puede modificar la edad de un empleado a un natural menor a 16"

Gestión de inventario

RF5.1: Dar de alta un objeto
Entrada: *Agente externo:* Administrador. *Acción*: Solicitar inserción. *Requisitos de datos de entrada* **RDE5.1**.
BD: *Requisito de datos de escritura* **RDW5.1**
Salida: *Agente externo:* Administrador. *Acción:* Confirmación resultado. *Requisitos de datos de salida:* Ninguno.
RDE5.1: Datos de entrada de alta de objeto
ID: Cadena de caracteres (20)
Cantidad Disponible: Numérico
Categoría: Cadena de caracteres (30)
Distribuidora: Cadena de caracteres (40)
RDW5.1: Datos almacenados del objeto
Los mismos que **RDE5.1**

RF5.2: Dar de baja un objeto

Entrada: *Agente externo:* Administrador. *Acción:* Solicitar borrado. *Requisitos de datos de entrada:* **RDE5.2**.

BD: *Requisitos de datos de lectura* **RDR5.2**. *Requisitos de datos de escritura* **RDW5.2**.

Salida: *Agente externo:* Administrador. *Acción:* Confirmación resultado. *Requisitos de datos de salida:* Ninguno.

RDE5.2: Datos de entrada de baja de objeto

ID: Cadena de caracteres (20)

RDR5.2: Datos de lectura del objeto a dar de baja

Los mismos que **RDE5.1** y RDE3.5

RDW5.2: Datos almacenados del objeto

Los mismos que **RDE5.1** y RDW3.5

RF5.3: Mostrar listado de los objeto por categoría

Entrada: *Agente externo*: Administrador o Trabajador. *Acción:* Solicitar listado. *Requisitos de datos de entrada:* **RDE5.3**.

BD: *Requisitos de datos de lectura* **RDR5.3**

Salida: *Agente externo:* Administrador o Trabajador. *Acción:* Confirmación de resultado. *Requisito de datos de salida:* **RDS5.3**.

RDE5.3: Datos de entrada de objeto.

Categoría: Cadena de caracteres (30)

RDR5.3: Datos de objeto almacenado.

Los mismos datos que **RDW5.1**

RDS5.3: Listado de registros, cada uno de ellos con los mismos datos que **RDR5.3**.

RF5.4: Mostrar datos de un objeto

Entrada: *Agente externo:* Administrador o Trabajador. *Acción:* Solicitar campos de un objeto. *Requisitos de datos de entrada:* **RDE5.4**.

BD: *Requisitos de datos de lectura* **RDR5.4**.

Salida: *Agente externo:* Administrador o Trabajador. *Acción:* Confirmación del resultado. *Requisitos de datos de salida:* **RDS5.4**.

RDE5.4: Datos de entrada de objeto.

Los mismos que **RDE5.2**.

RDR5.4: Datos de objeto almacenado.

Los mismos que **RDE5.1**.

RDS5.4: Datos del objeto, los mismos que **RDR5.4**.

RF5.5: Cambiar datos de un objeto

Entrada: *Agente externo:* Administrador. *Acción:* Solicitar modificación de los datos de un objeto. *Requisito de datos de entrada:* **RDE5.5**.

BD: Requisitos de datos de lectura **RDR5.5**. Requisitos de datos de escritura **RDW5.5**.

Salida: *Agente externo:* Administrador. *Acción:* Confirmación del resultado. *Requisitos de datos de salida:* Ninguno.

RDE5.5: Datos de entrada del objeto.

Los mismos que **RDE5.1**.

RDR5.5: Datos almacenados del objeto.

Los mismos que **RDE5.1**. y RDE3.5

RDW5.5: Datos almacenados del objeto.

Los mismos que **RDE5.1**. y RDW3.5

RF5.6: Mostrar objetos sin disponibilidad

Entrada: *Agente externo:* Administrador o Trabajador. *Acción:* Solicitar listado de objetos cuya cantidad sea igual a 0. *Requisitos de datos de entrada:* Ninguno.
BD: *Requisitos de datos de lectura* **RDR5.6**.
Salida: *Agente externo:* Administrador o Trabajador. *Acción:* Confirmación del resultado. *Requisitos de salida*: **RDS5.6**.

RDR5.6: Datos del objeto almacenado.

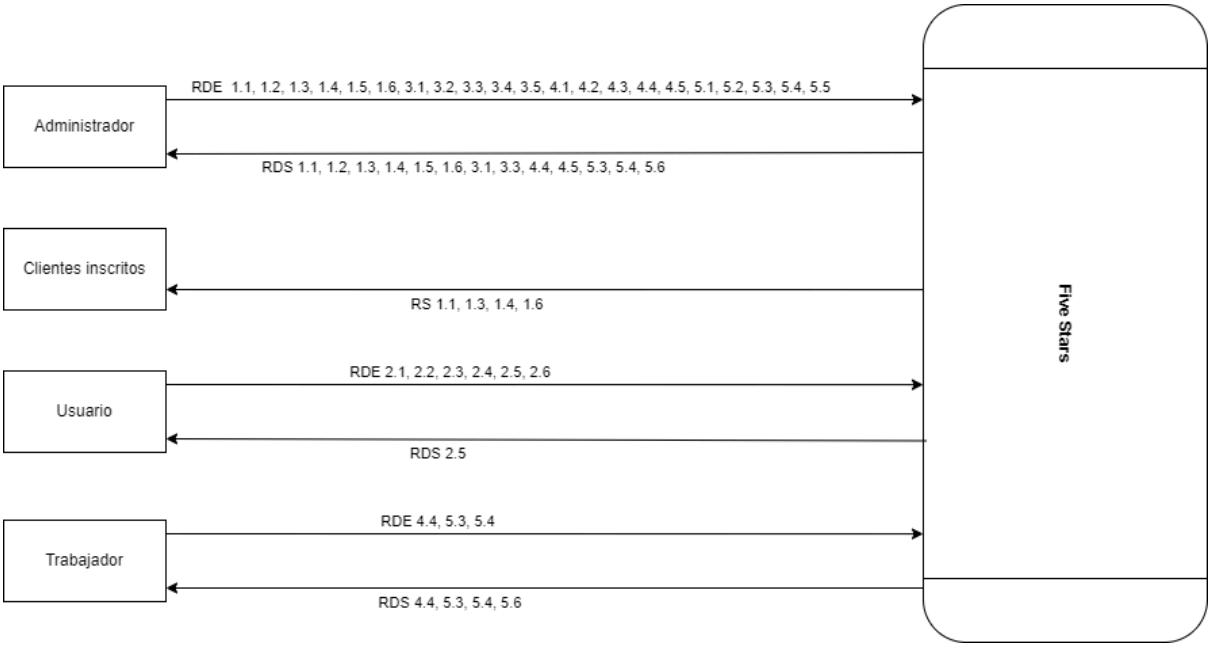
Los mismos que **RDW5.1**.

RDS5.6: Listado de objetos. cada uno con los mismos datos que **RDR5.6**.

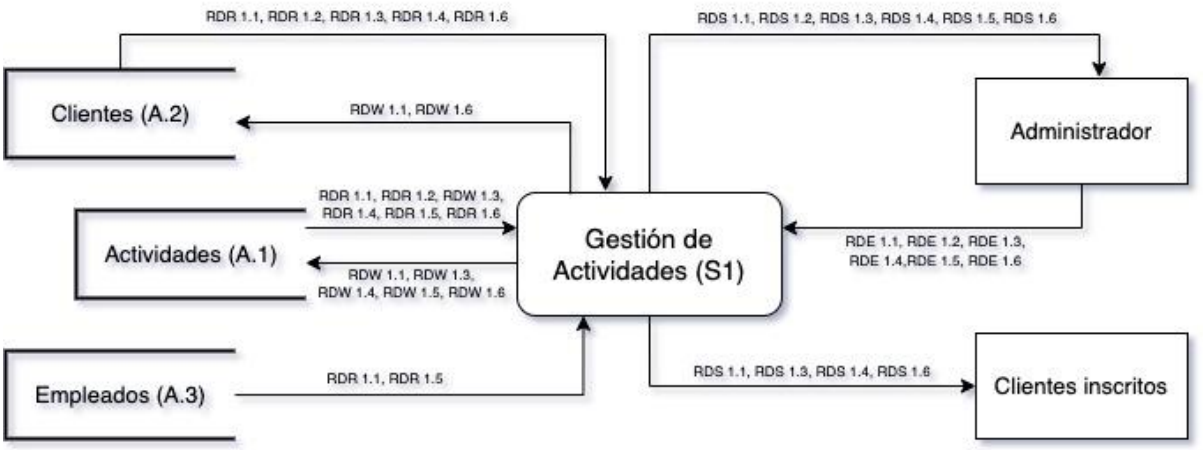
RS5.1: La cantidad debe ser un número positivo (incluyendo al 0).
RF: RF5.1. RF5.5.
RD(s): RDW5.1. RDW5.5.
Descripción: “El dominio del atributo cantidad son los números positivos, incluyendo al 0”.

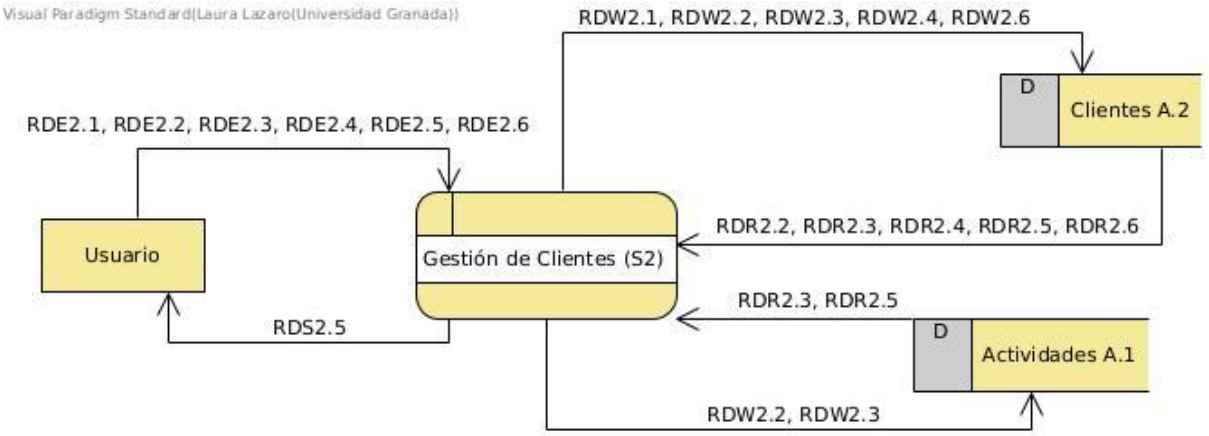
RS5.2: Todos los objetos tienen que tener asignada una categoría
RF: RF5.1. RF5.5.
RD(s): RDW5.1. RDW5.5.
Descripción: “Un objeto no puede tener el atributo categoría con valor nulo o vacío“.

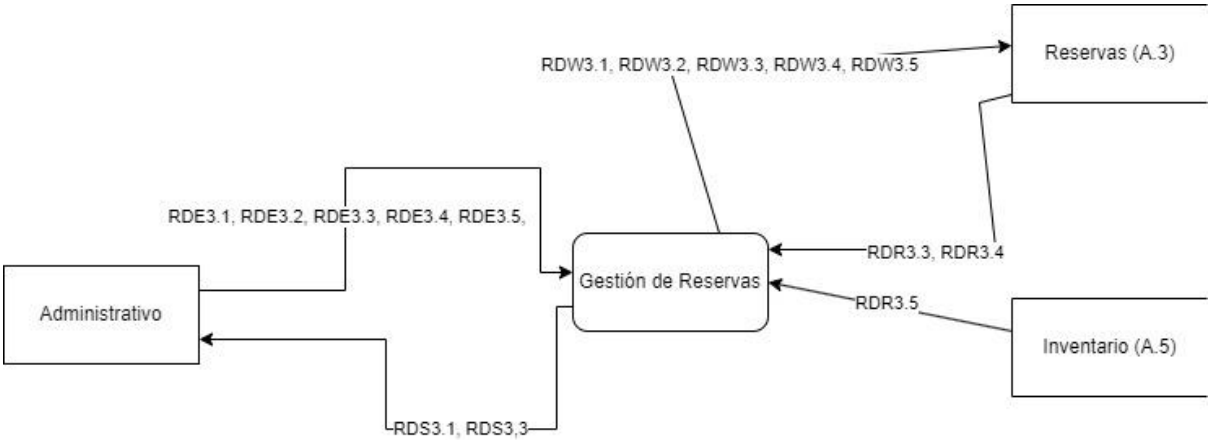
4. DFD Esquema de caja negra

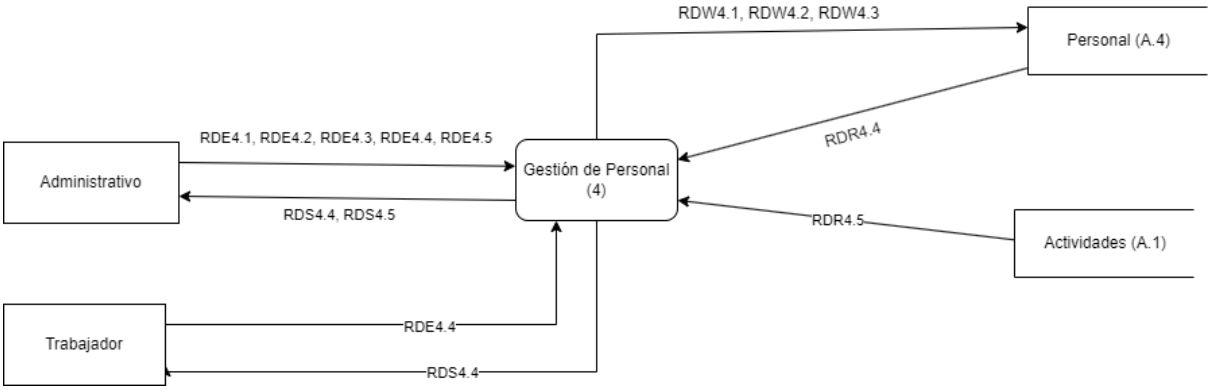


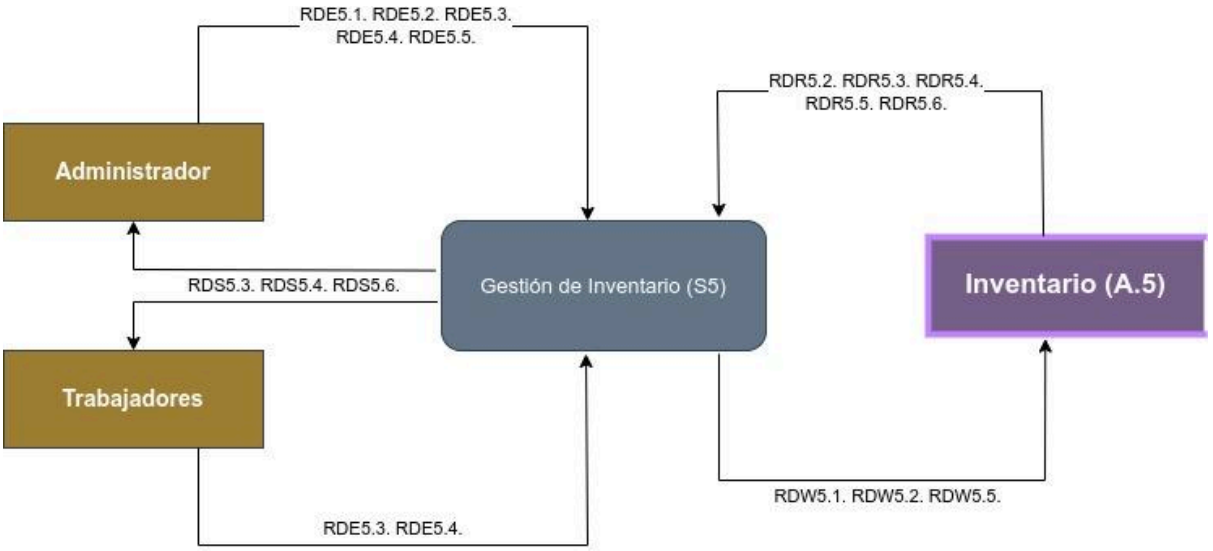
5. DFD Armazón



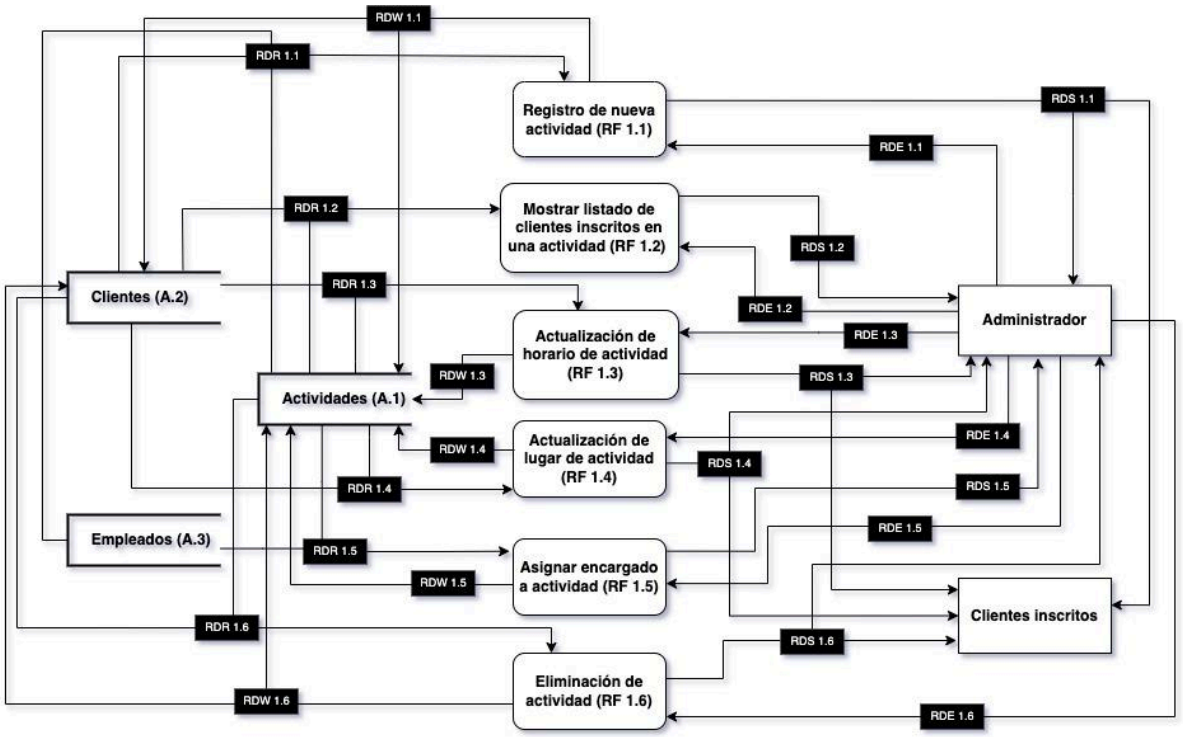


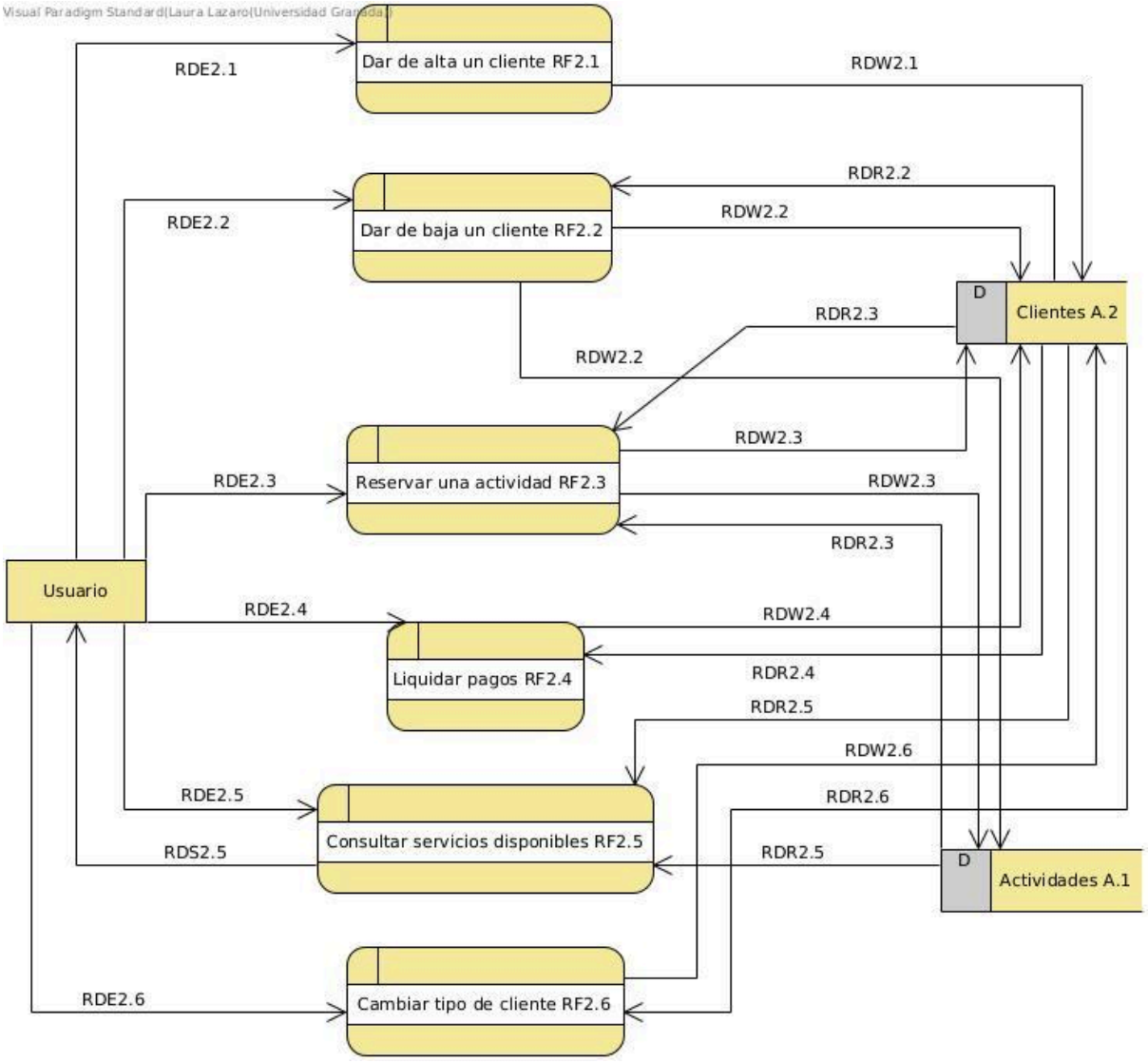


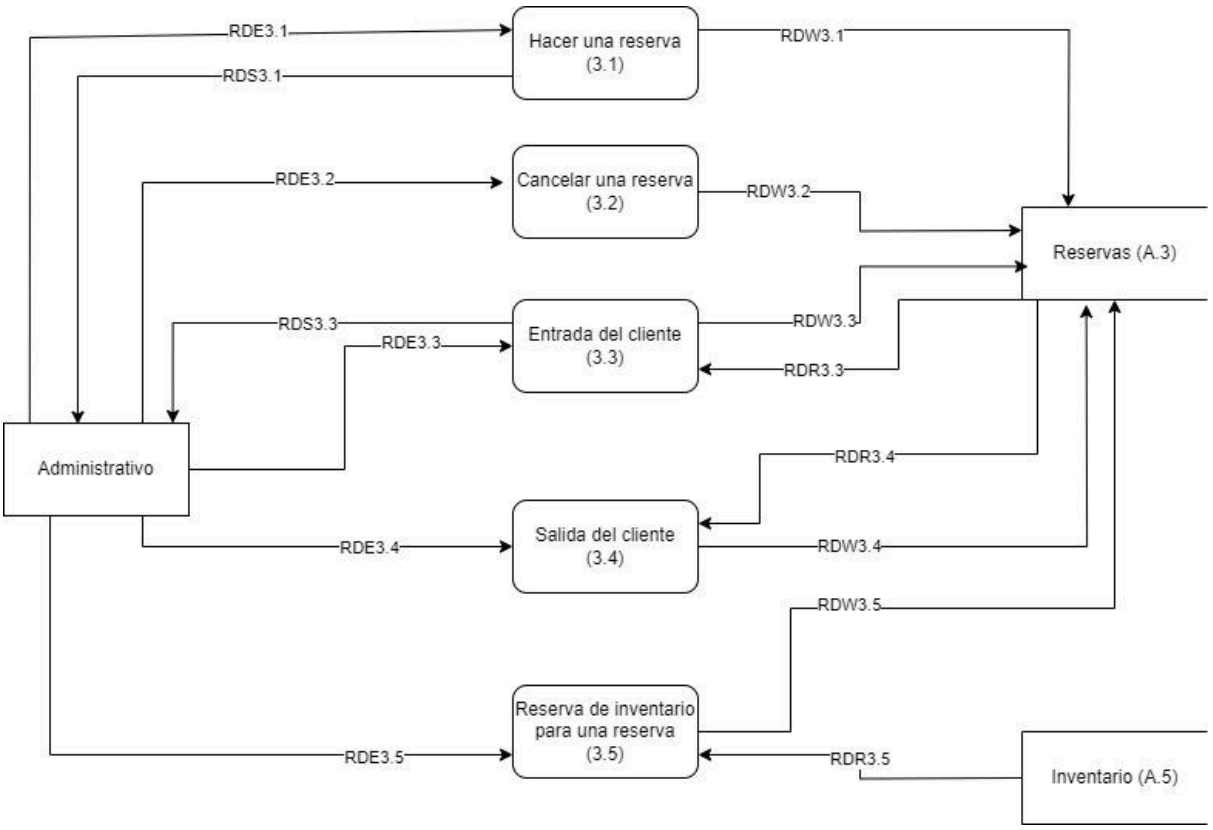


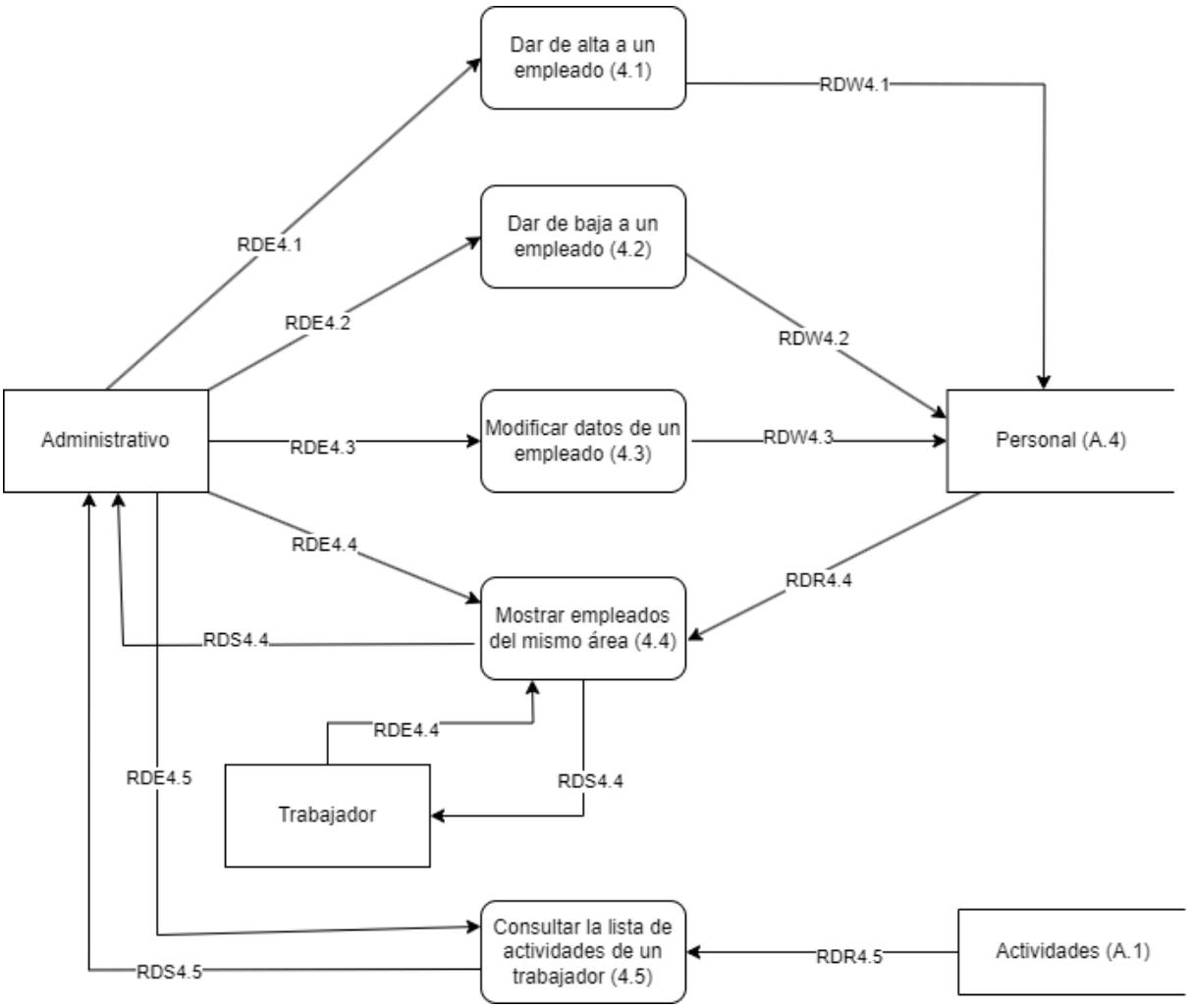


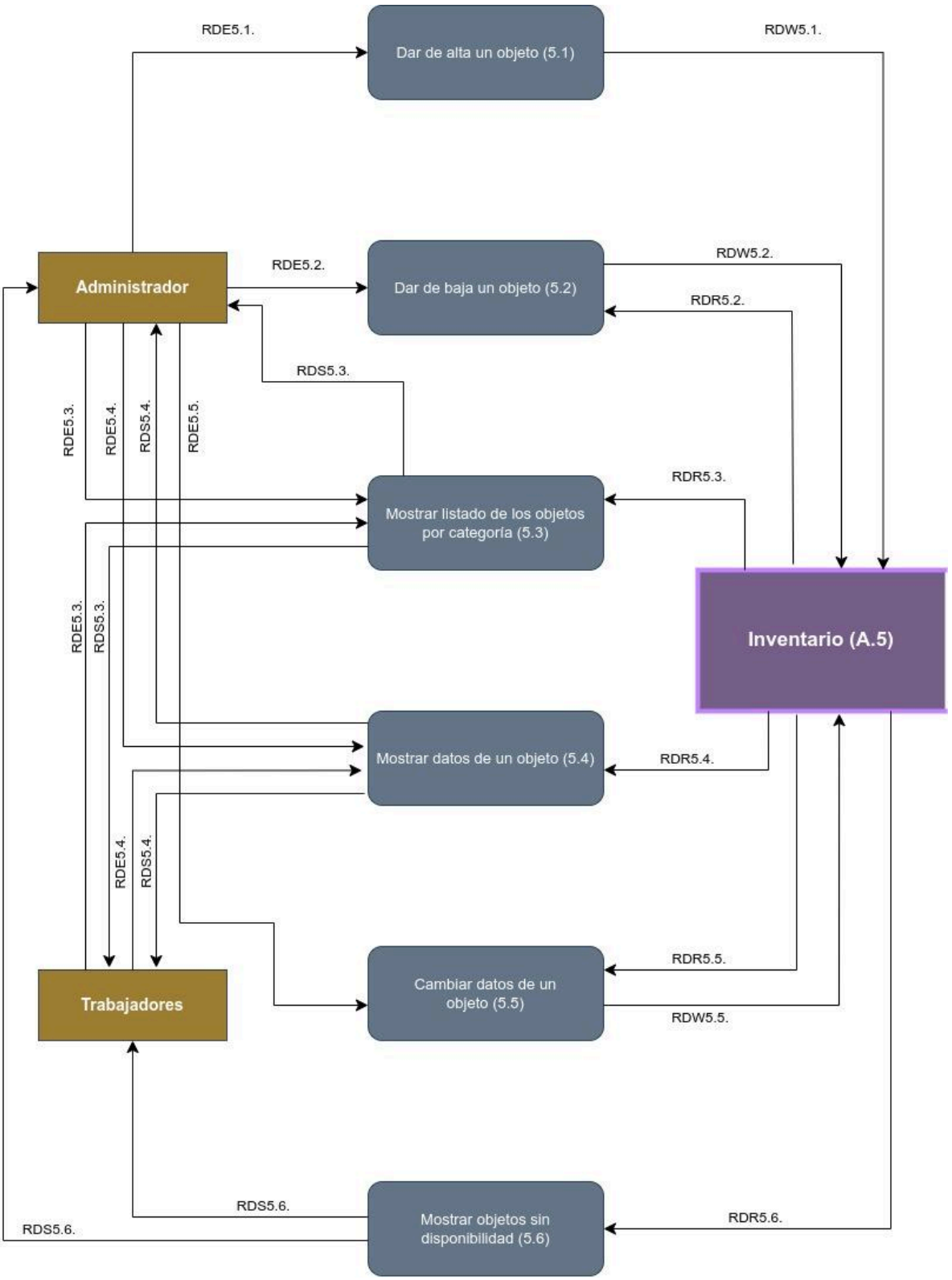
6. DFD de los subsistemas





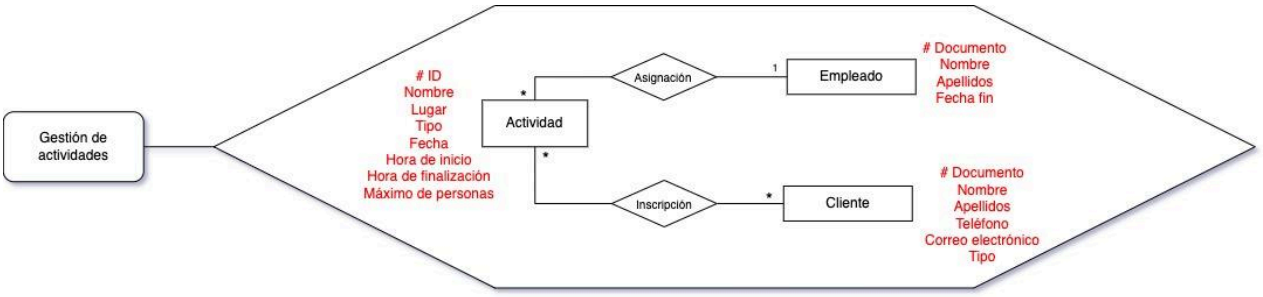

RDE3.1
RDS3.1
Hacer una reserva
(3.1)
RDW3.1
RDE3.2
Cancelar una reserva
(3.2)
RDW3.2
Reservas (A.3)
RDS3.3
RDE3.3
Entrada del cliente
(3.3)
RDW3.3
RDR3.3
Administrativo
RDR3.4
RDE3.4
Salida del cliente
(3.4)
RDW3.4
RDW3.5
Reserva de inventario
para una reserva
(3.5)
RDE3.5
RDR3.5
Inventario (A.5)


Dar de alta a un empleado (4.1)
RDW4.1
Dar de baja a un empleado (4.2)
RDE4.1
RDE4.2
RDW4.2
Administrativo
RDE4.3
Modificar datos de un empleado (4.3)
RDW4.3
Personal (A.4)
RDE4.4
RDR4.4
Mostrar empleados del mismo área (4.4)
RDS4.4
RDE4.4
RDE4.5
RDS4.4
Trabajador
Consultar la lista de actividades de un trabajador (4.5)
RDR4.5
Actividades (A.1)
RDS4.5

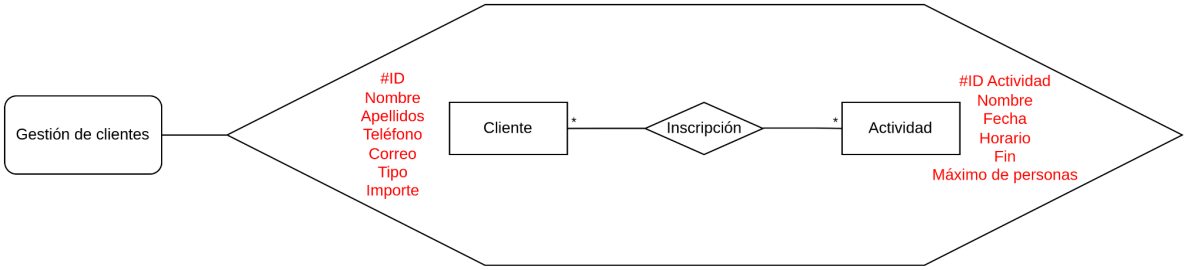

RDE5.1.
Dar de alta un objeto (5.1)
RDW5.1.
Administrador
RDE5.2.
Dar de baja un objeto (5.2)
RDW5.2.
RDR5.2.
RDS5.3.
RDE5.3.
RDE5.4.
RDS5.4.
RDE5.5.
RDR5.3.
Mostrar listado de los objetos por categoría (5.3)
Inventario (A.5)
RDE5.3.
RDS5.3.
Mostrar datos de un objeto (5.4)
RDR5.4.
RDE5.4.
RDS5.4.
RDR5.5.
Trabajadores
Cambiar datos de un objeto (5.5)
RDW5.5.
RDS5.6.
Mostrar objetos sin disponibilidad (5.6)
RDR5.6.
RDS5.6.

7. Esquemas externos DFD0

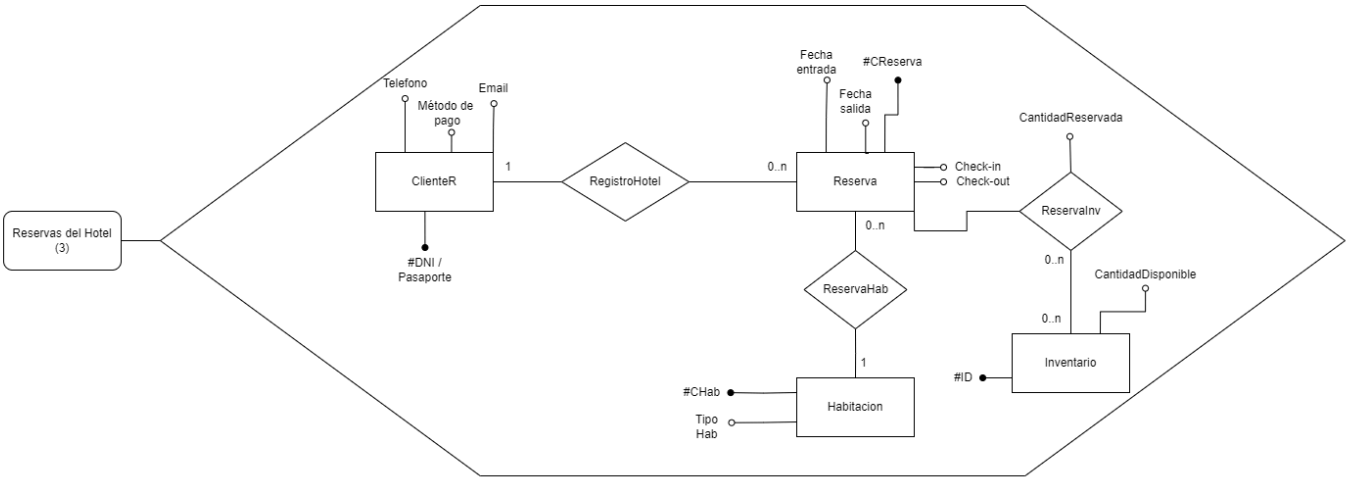
Gestión de actividades



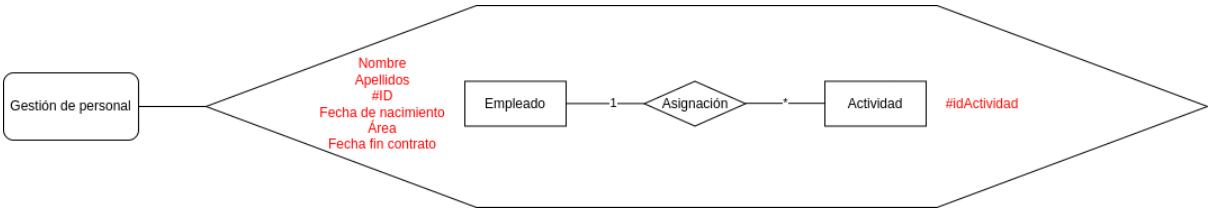
Gestión de clientes



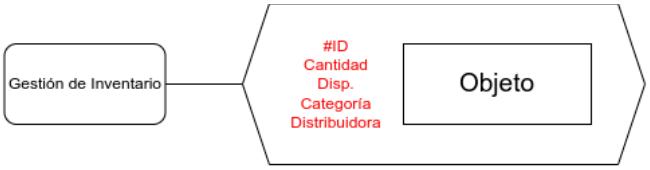
Gestión de reservas



Gestión de trabajadores



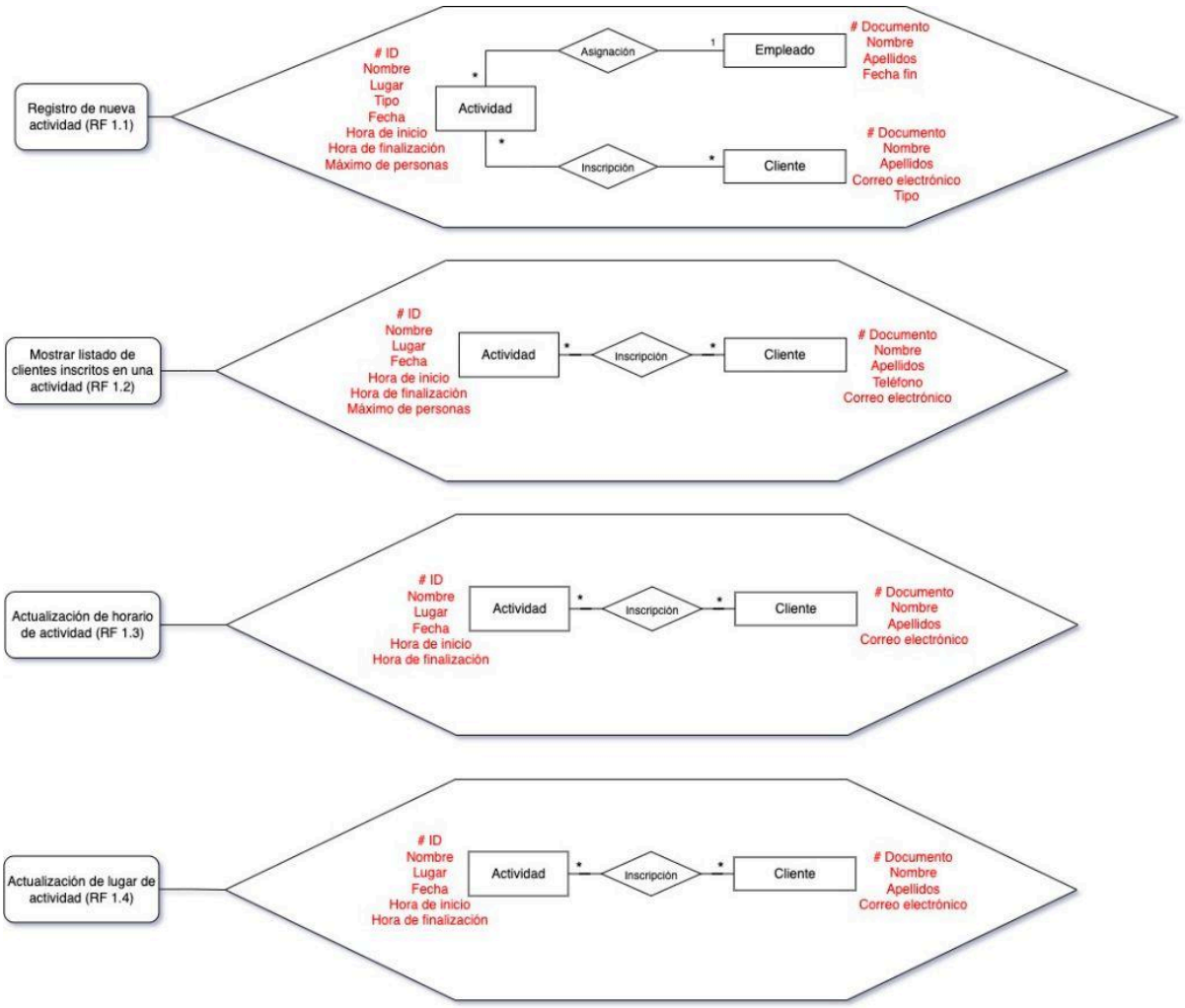
Gestión de inventario

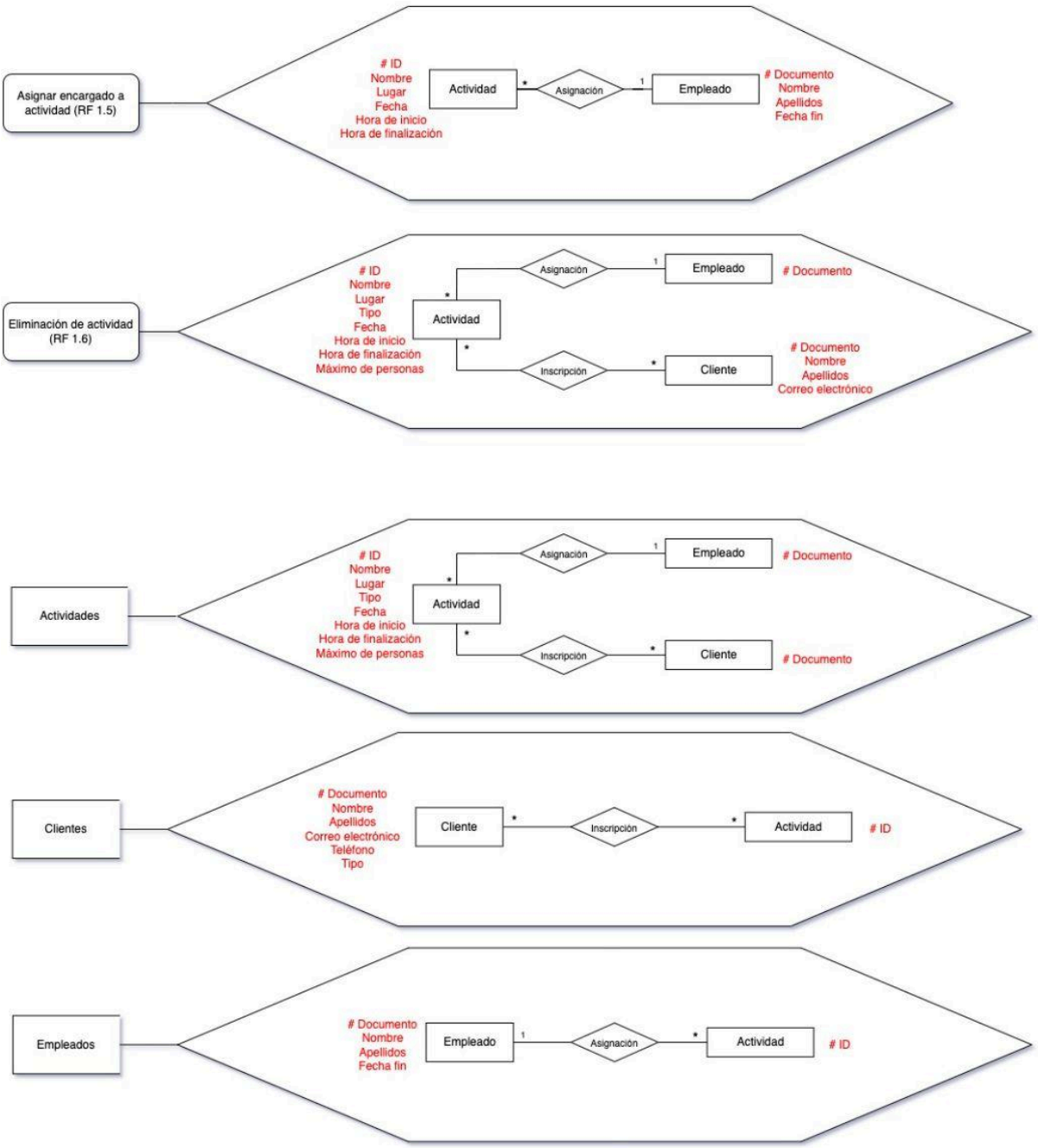


8. Esquemas externos DFD1

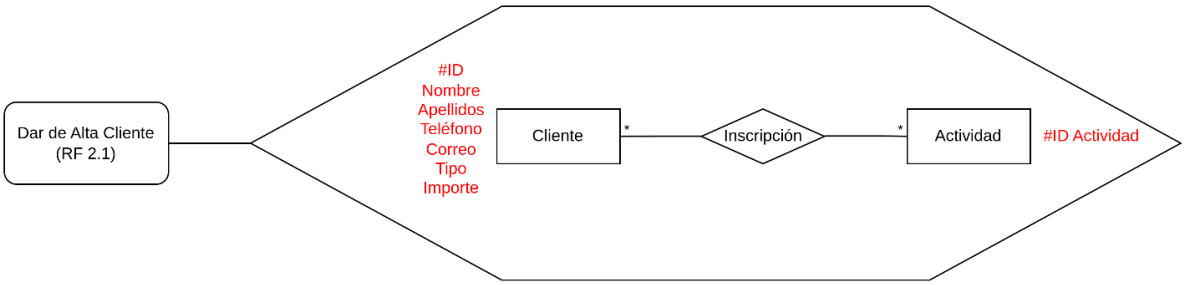
Se incluyen los esquemas de los almacenes aquí, ya que no se redujeron y son los mismos que en el DFD0.

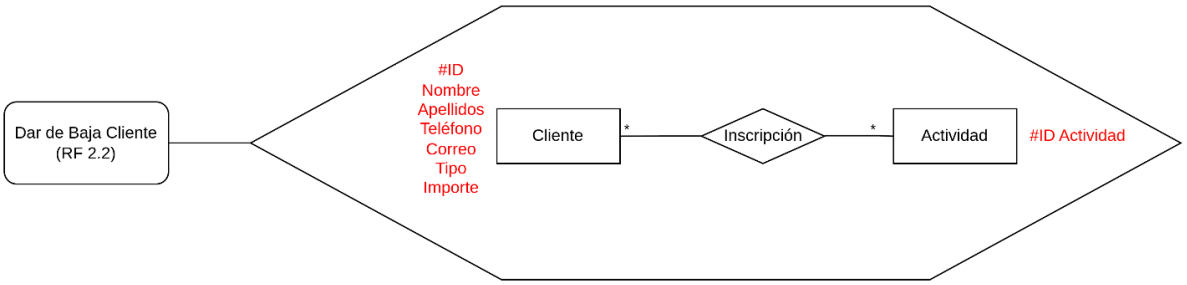
Gestión de actividades

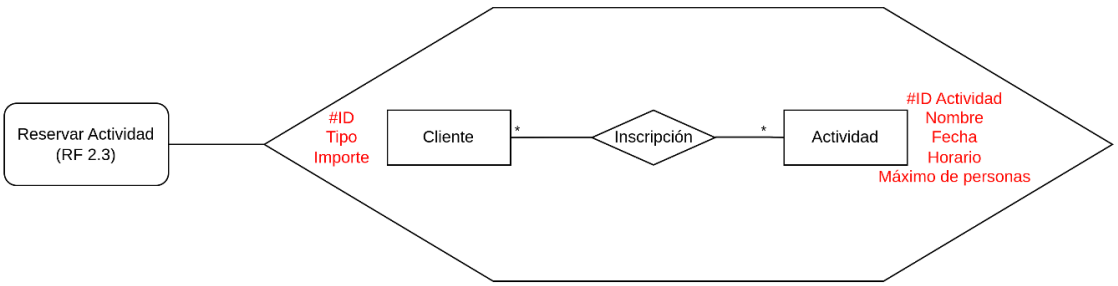


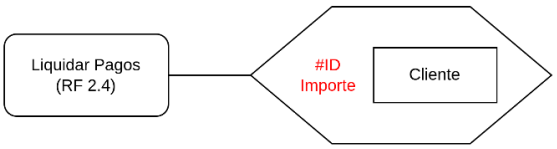

Asignar encargado a actividad (RF 1.5)
ID
Nombre
Lugar
Fecha
Hora de inicio
Hora de finalización
Actividad
*
Asignación
1
Empleado
Documento
Nombre
Apellidos
Fecha fin
Eliminación de actividad (RF 1.6)
ID
Nombre
Lugar
Tipo
Fecha
Hora de inicio
Hora de finalización
Máximo de personas
Actividad
Asignación
1
Empleado
Documento
Inscripción
Cliente
Documento
Nombre
Apellidos
Correo electrónico
Actividades
ID
Nombre
Lugar
Tipo
Fecha
Hora de inicio
Hora de finalización
Máximo de personas
Actividad
Asignación
Empleado
Documento
Inscripción
Cliente
Documento
Clientes
Documento
Nombre
Apellidos
Correo electrónico
Teléfono
Tipo
Cliente
Inscripción
Actividad
ID
Empleados
Documento
Nombre
Apellidos
Fecha fin
Empleado
Asignación
Actividad
ID

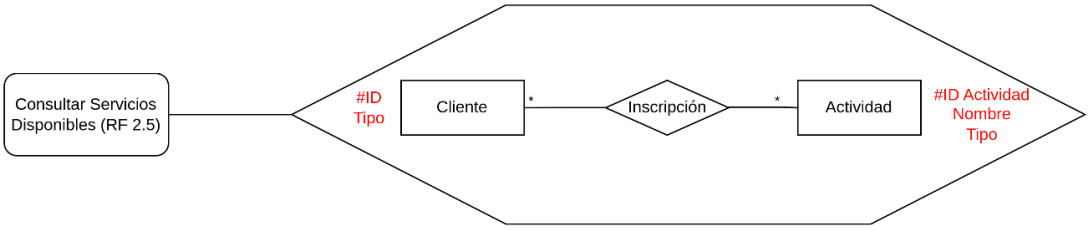
Gestión de clientes

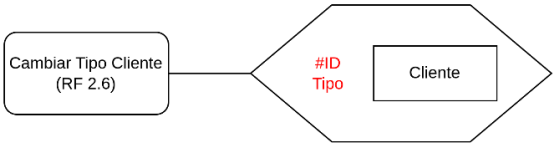


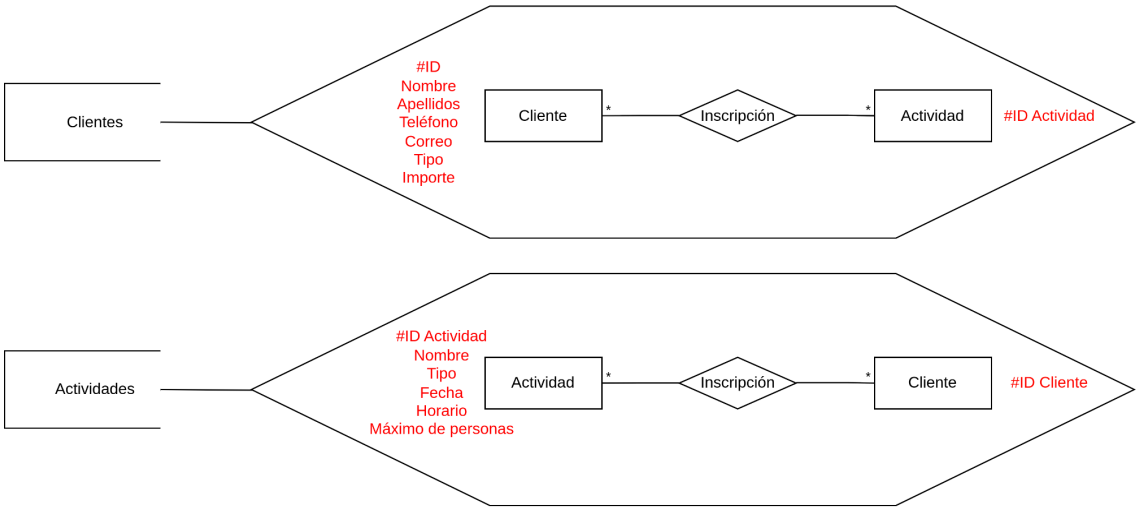




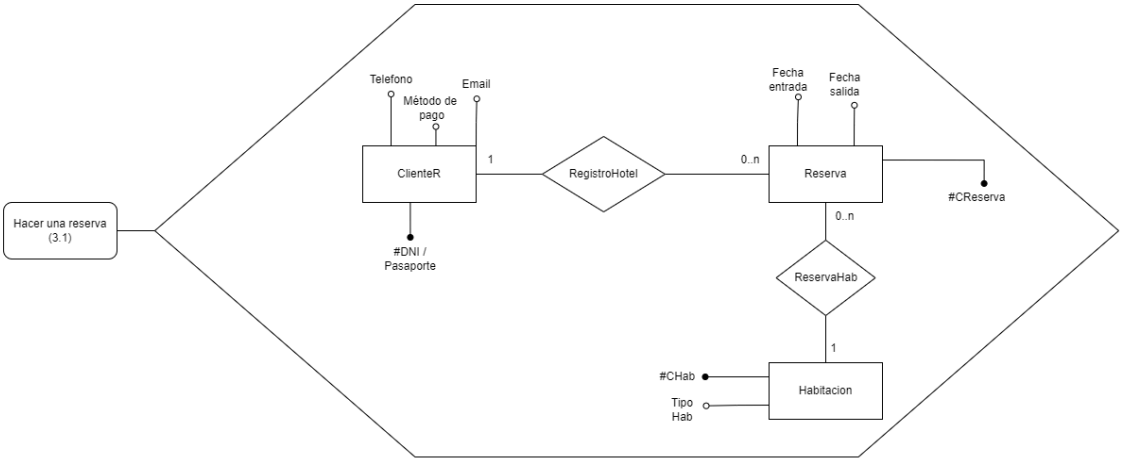


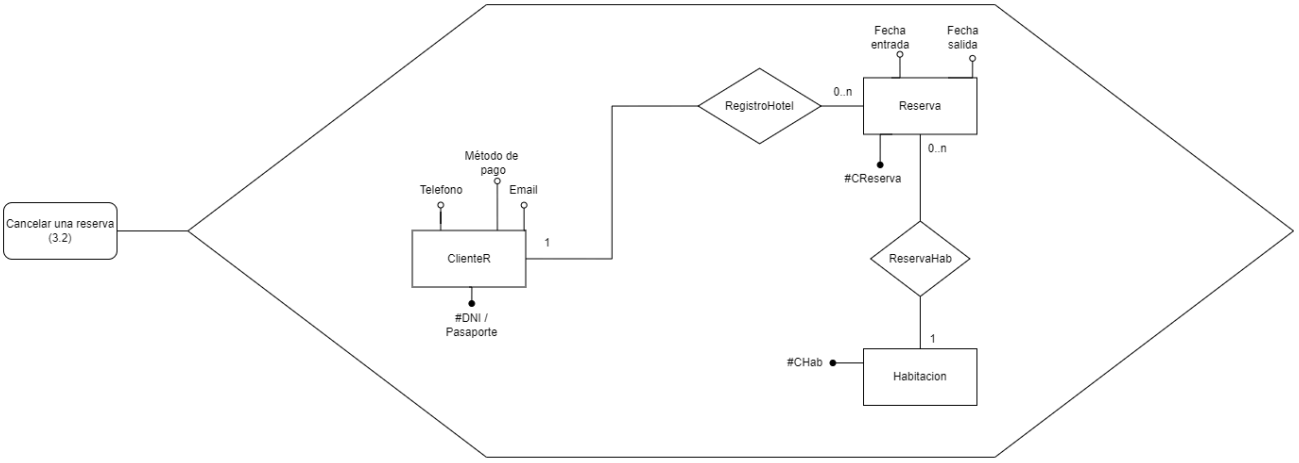


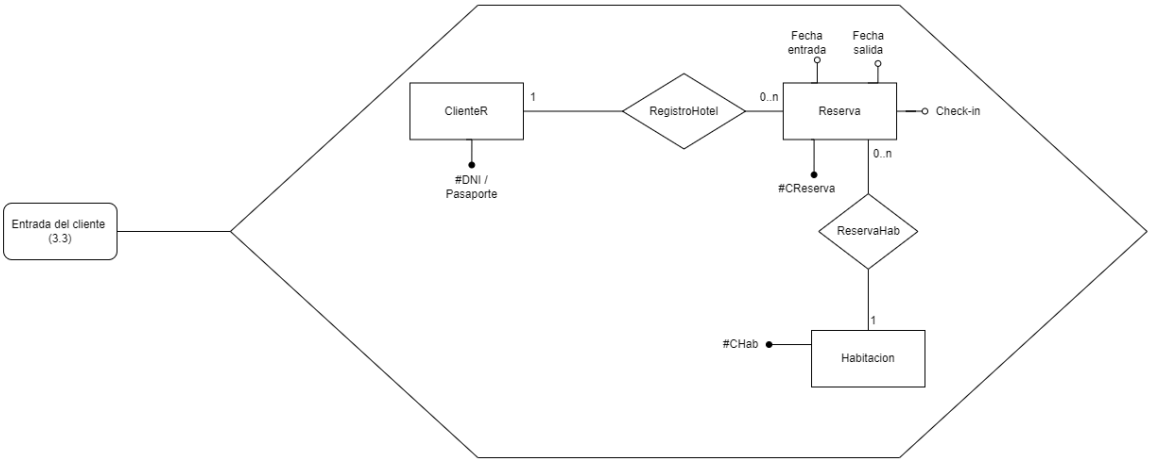


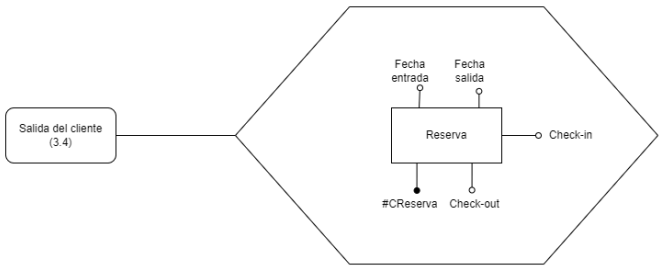

Clientes
#ID
Nombre
Apellidos
Teléfono
Correo
Tipo
Importe
Cliente
*
Inscripción
*
Actividad
#ID Actividad
Actividades
#ID Actividad
Nombre
Tipo
Fecha
Horario
Máximo de personas
Actividad
*
Inscripción
*
Cliente
#ID Cliente

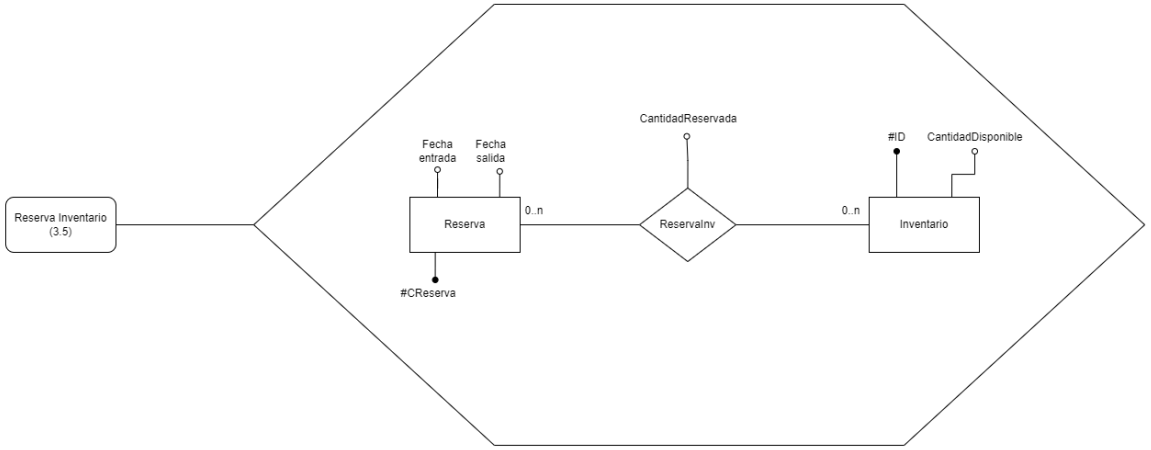
Gestión de reservas

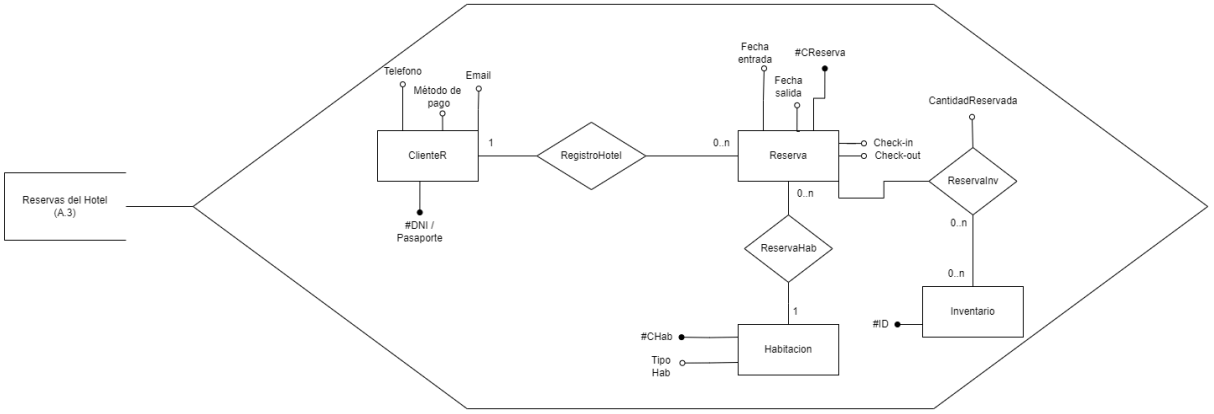




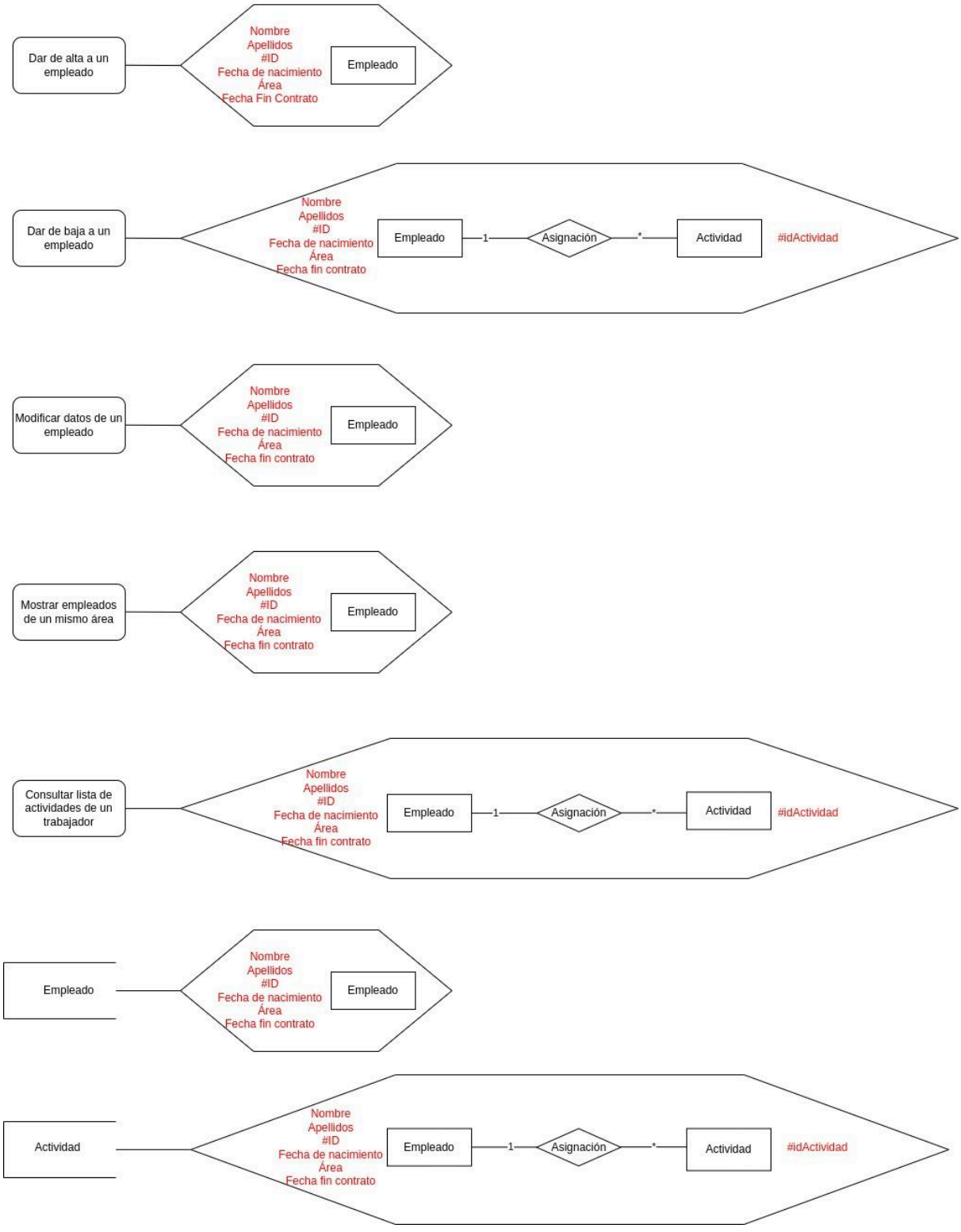



Salida del cliente
(3.4)
Fecha
entrada
Fecha
salida
Reserva
Check-in
#CReserva
Check-out

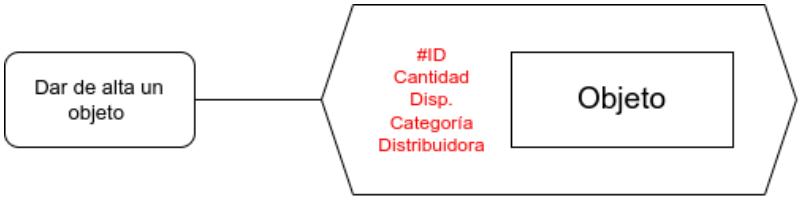

Reserva Inventario
(3.5)
Fecha
entrada
Fecha
salida
Reserva
#CReserva
0..n
CantidadReservada
ReservaInv
0..n
#ID
CantidadDisponible
Inventario

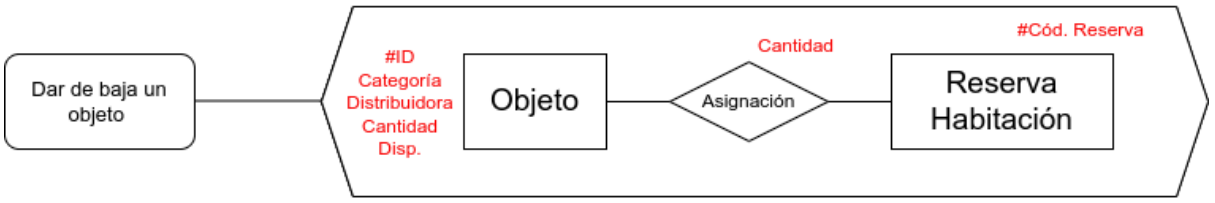

Reservas del Hotel
(A.3)
Telefono
Email
Método de
pago
ClienteR
#DNI /
Pasaporte
1
RegistroHotel
0..n
Fecha
entrada
#CReserva
Fecha
salida
Reserva
Check-in
Check-out
0..n
ReservaHab
1
#CHab
Tipo
Hab
Habitacion
CantidadReservada
ReservaInv
0..n
0..n
#ID
Inventario

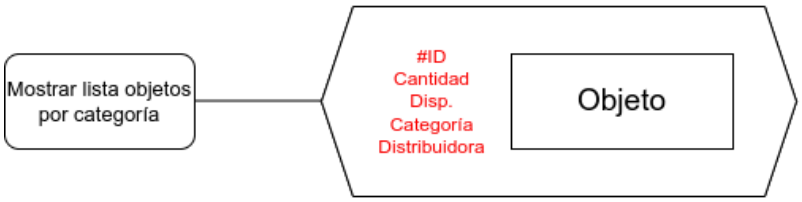
Gestión de personal

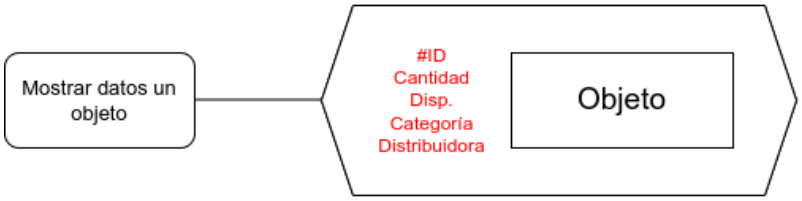


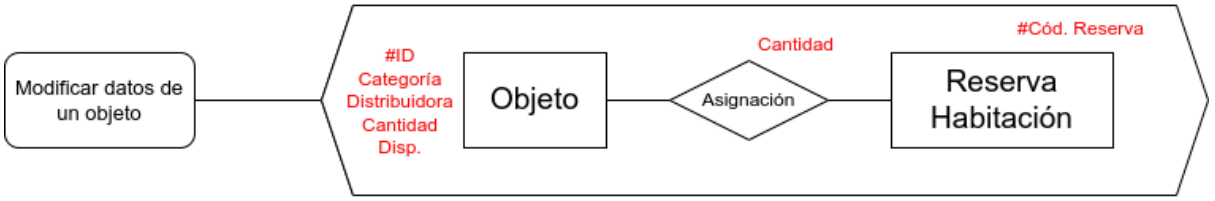
Gestión de Inventario

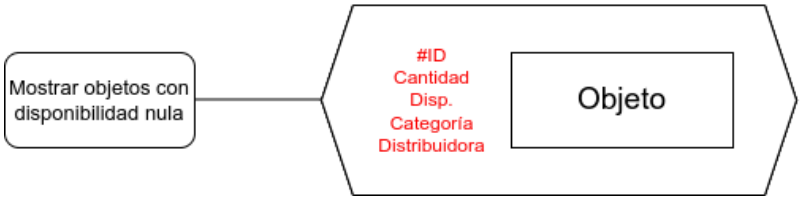


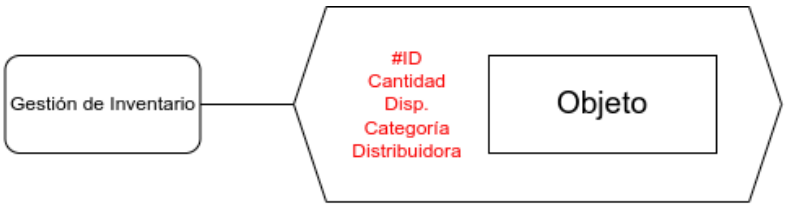


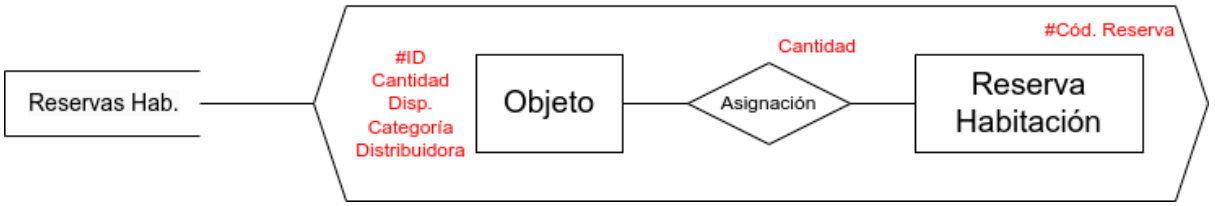




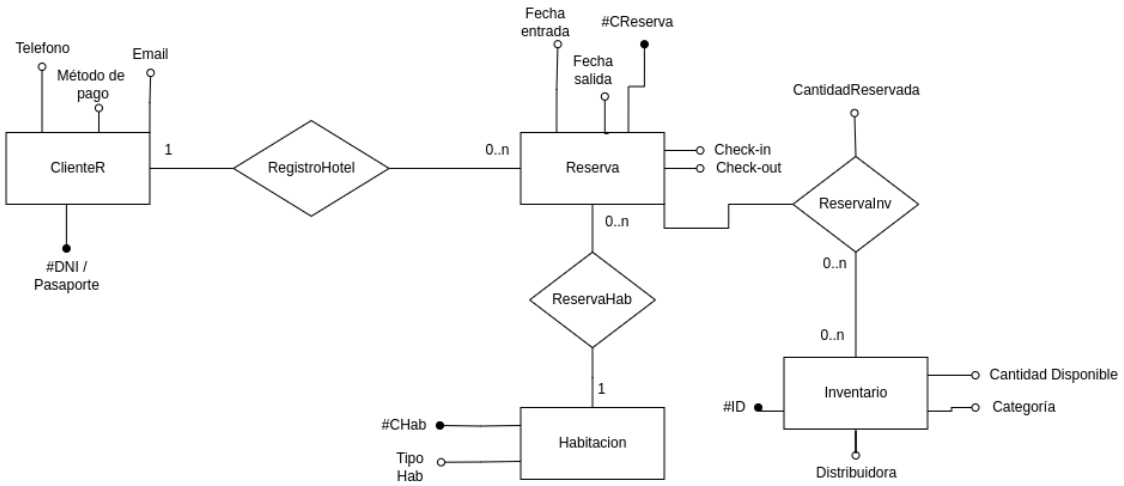


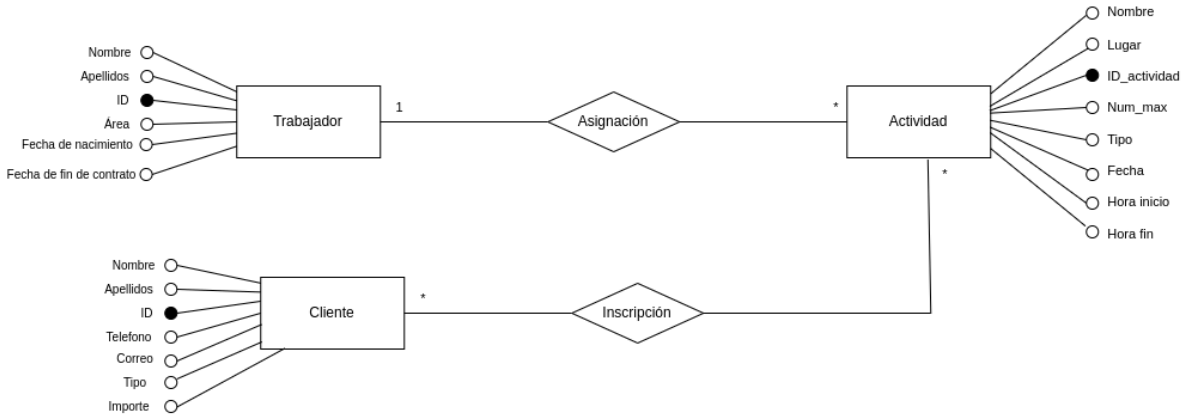



Gestión de Inventario
#ID
Cantidad
Disp.
Categoría
Distribuidora
Objeto

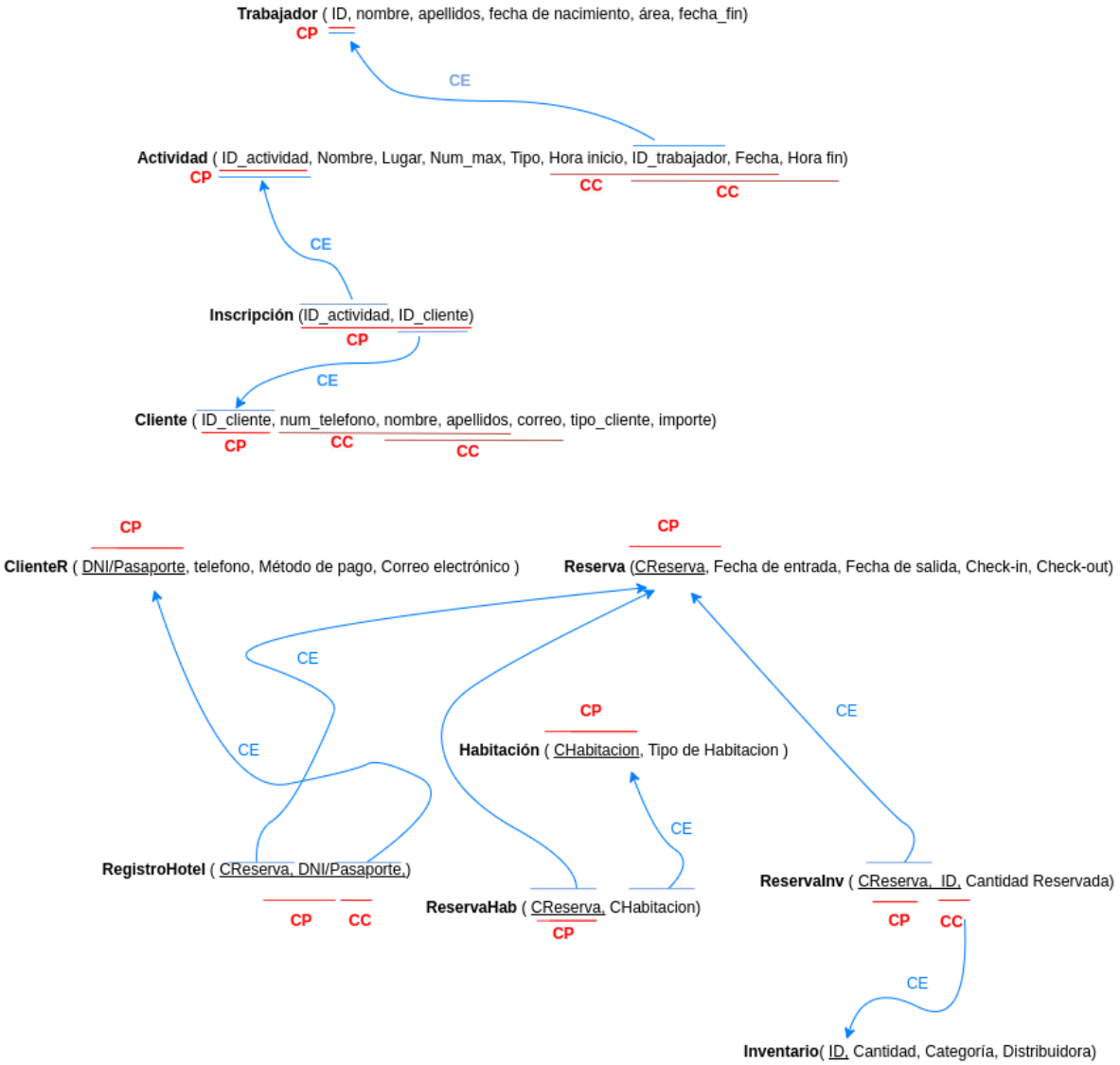

Reservas Hab.
#ID
Cantidad
Disp.
Categoría
Distribuidora
Objeto
Cantidad
Asignación
#Cód. Reserva
Reserva
Habitación

9. Diagramas E/R.





10. Paso a tablas



11. Dependencias funcionales y normalización.

Tabla Actividad: *Actividad (#ID_actividad, Nombre, Lugar, Num_max, Tipo, Hora inicio, ID_trabajador, Fecha, Hora fin).*
Dependencias funcionales completas:

ID_actividad → Nombre
ID_actividad → Lugar
ID_actividad → Tipo
ID_actividad → Fecha
ID_actividad → Hora inicio
ID_actividad → Hora fin
ID_actividad → Num_max
ID_actividad → ID_trabajador

Fecha, Hora inicio, ID_trabajador → ID_actividad
Fecha, Hora inicio, ID_trabajador → Nombre
Fecha, Hora inicio, ID_trabajador → Lugar
Fecha, Hora inicio, ID_trabajador → Tipo
Fecha, Hora inicio, ID_trabajador → Hora fin
Fecha, Hora inicio, ID_trabajador → Num_max

Fecha, Hora fin, ID_trabajador → ID_actividad
Fecha, Hora fin, ID_trabajador → Nombre
Fecha, Hora fin, ID_trabajador → Lugar
Fecha, Hora fin, ID_trabajador → Tipo
Fecha, Hora fin, ID_trabajador → Hora inicio
Fecha, Hora fin, ID_trabajador → Num_max

- **ID_actividad** es la clave primaria.
- "**Fecha, Hora inicio, ID_trabajador**" y **"Fecha, Hora fin, ID_trabajador**" son claves candidatas, ya que cumplen el criterio de **unicidad** (determinan el resto de atributos por RS 1.1-1.2, ya que un mismo empleado no puede trabajar en dos actividades al mismo tiempo, suponiendo una duración de actividad no nula) y de **minimalidad** (ningún subconjunto propio de estos determina unívocamente el resto de atributos: puede haber varias actividades con el mismo horario y fecha, varias actividades asignadas a un empleado en la misma fecha, o por supuesto, varias actividades con el mismo horario y mismo encargado asignado pero en distintas fechas).
- **ID_trabajador** es una clave externa que referencia a la tabla "Trabajador".

La tabla está en Forma Normal de Boyce-Codd (**FNBC**) porque toda dependencia funcional no trivial cumple que a la izquierda hay una superclave, es decir, un conjunto de atributos que determina de forma única el resto de atributos del registro. En consecuencia, estará en Tercera Forma Normal (3FN) y Segunda Forma Normal (2FN). La tabla ya está normalizada.

Nota: Dos actividades pueden coincidir en nombre, lugar, tipo, fecha, hora de inicio, hora de finalización y número máximo de personas. No existe restricción al respecto. Si coinciden en

todo esto, podrían asignarse a distintos encargados, considerándose como “grupos” de la misma actividad.

Tabla Inscripcion: *Inscripción (#ID_actividad, #ID_cliente)*

No hay dependencias funcionales no triviales ya que solo hay dos campos en la tabla (ID_actividad, ID_cliente) y conforman una clave primaria. Ambas son claves externas que referencian a “Actividad” y “Cliente” respectivamente. Por tanto, ya está normalizada.

Tabla Cliente: *Cliente (#ID_cliente, num_telefono, nombre, apellidos, correo, tipo_cliente, importe)*

Dependencias funcionales en la tabla:

ID → Nombre
ID → Apellidos
ID → Número de teléfono
ID → Correo electrónico
ID → Tipo
ID → Importe
ID → Lista ID Actividades

Nombre, Apellidos, Correo electrónico → ID
Nombre, Apellidos, Correo electrónico → Número de teléfono
Nombre, Apellidos, Correo electrónico → Tipo
Nombre, Apellidos, Correo electrónico → Importe
Nombre, Apellidos, Correo electrónico → Lista ID Actividades

Nombre, Apellidos, Teléfono → ID
Nombre, Apellidos, Teléfono → Correo electrónico
Nombre, Apellidos, Teléfono → Tipo
Nombre, Apellidos, Teléfono → Importe
Nombre, Apellidos, Teléfono → Lista ID Actividades

Se encuentra en segunda forma normal, ya que todos sus atributos no primos dependen de claves candidatas, ya sea de la clave primaria (ID cliente) o de las otras candidatas (Nombre+Apellidos+Correo o Nombre+Apellidos+Teléfono). Además, se encuentra en tercera forma normal debido a que en las dependencias a la izquierda, tenemos siempre la clave primaria o las candidatas, todas superclaves. Como todo determinante es una clave candidata, afirmamos que está en la forma normal de Boyce-Codd. En conclusión, la tabla de clientes ya está normalizada.

Tabla ClienteR: *ClienteR (#DNI/Pasaporte, Teléfono, Correo electrónico, Método de pago)*

Dependencias funcionales en la tabla de los clientes de reservas:

DNI/Pasaporte → Teléfono
DNI/Pasaporte→ Correo electrónico

DNI/Pasaporte→ Método de pago

Para toda dependencia funcional no trivial descrita en el punto anterior, a la izquierda hay una superclave (#DNI/Pasaporte), luego, ya está en FNBC y, por tanto, también está en 3FN, 2FN y 1FN. No tenemos en cuenta ninguna de las otras claves como candidatas ya que en caso de que por ejemplo una pareja hiciera una reserva dependiendo del miembro de la pareja que hiciera la reserva podrían coincidir los datos, mientras que el DNI es único e intransferible.

Tabla de Reservas: *Reserva (#CReserva, Fecha entrada, Fecha salida, Check-in, Check-out)*
Dependencias funcionales en la tabla reservas:

CReserva → Fecha entrada
CReserva → Fecha salida
CReserva → Check-in
CReserva → Check-out

Para toda dependencia funcional no trivial descrita en el punto anterior, a la izquierda hay una superclave (#CReserva), luego, ya está en FNBC y, por tanto, también está en 3FN, 2FN y 1FN. No tenemos en cuenta ninguna de las otras claves como candidatas ya que en podrían repetirse en distintas reservas.

Tabla de Habitaciones: *Habitacion (#CHabitación, Tipo de Habitación)*
Dependencias funcionales en la tabla de habitaciones:

CHabitación→ Tipo de habitación

Para toda dependencia funcional no trivial descrita en el punto anterior, a la izquierda hay una superclave (#CHabitación), luego, ya está en FNBC y, por tanto, también está en 3FN, 2FN y 1FN.

Tabla RegistroHotel: *RegistroHotel (#CReserva, #DNI/Pasaporte)*
Dependencias funcionales en la tabla RegistroHotel:

CReserva → DNI/Pasaporte
DNI/Pasaporte → CReserva

Claves candidatas: {#DNI/Pasaporte}.
Para toda dependencia funcional no trivial descrita en el punto anterior, a la izquierda hay una superclave (#CReserva, que es la clave primaria) o (#DNI/Pasaporte), luego, ya está en FNBC y, por tanto, también está en 3FN, 2FN y 1FN.

Tabla ReservaHab: *ReservaHab (#CReserva, CHabitación)*
Dependencias funcionales en la tabla:

CReserva → CHabitación

Para toda dependencia funcional no trivial descrita en el punto anterior, a la izquierda hay una superclave (#CReserva), luego, ya está en FNBC y, por tanto, también está en 3FN, 2FN y 1FN. No usamos CHabitación como clave candidata ya que la misma habitación dependiendo de las fechas puede estar en varias reservas.

Tabla ReservaInv: *ReservaInv (#CReserva, #ID, Cantidad Reservada, Cantidad Disponible, Fecha entrada, Fecha salida)*
Dependencias funcionales en la tabla ReservaInv:

CReserva → ID
CReserva → Cantidad Reservada
CReserva → Cantidad Disponible
CReserva → Fecha entrada
CReserva → Fecha salida
ID→ Cantidad Reservada
ID→ Cantidad Disponible
ID→ Fecha entrada
ID→ Fecha salida

Claves candidatas: {#ID}
Para toda dependencia funcional no trivial descrita en el punto anterior, a la izquierda hay una superclave (#CReserva, esta es la clave primaria) o (#ID), luego, ya está en FNBC y, por tanto, también está en 3FN, 2FN y 1FN.

Tabla trabajador: *Trabajador (#ID, nombre, apellidos, fecha de nacimiento, área, fecha_fin)*
Dependencias funcionales en la tabla personal:

ID → Nombre
ID → Apellidos
ID → Área
ID → Fecha de nacimiento

Podemos afirmar que se encuentra en segunda forma normal, ya que todos sus atributos no primos dependen de claves candidatas, en este caso, de la clave primaria, el ID del trabajador. Además, se encuentra en tercera forma normal debido a que no existen dependencias transitivas problemáticas. Esto último se puede observar gracias a que en las dependencias a la izquierda, tenemos la clave primaria, que sería una superclave. Finalmente, por esta última razón y que solo tenemos una clave candidata no compuesta, afirmamos que está en la forma normal de Boyce-Codd.
En conclusión, la tabla de personal ya está normalizada

Tabla de Inventario: *Inventario(#ID, Cantidad, Categoría, Distribuidora)*
Dependencias funcionales en la tabla inventario:

ID/Nombre → Categoría
ID/Nombre → Cantidad Disponible
ID/Nombre → Distribuidora

Podemos afirmar que se encuentra en segunda forma normal, debido a que todos sus atributos no primos dependen de claves candidatas. En nuestro caso, solo tenemos una clave candidata, que es a su vez la clave primaria ID o Nombre. Además, se encuentra en tercera forma normal, debido a que no existen dependencias transitivas problemáticas, ya que a la izquierda tenemos una superclave, que es la clave primaria.
Por último, como solo tenemos una clave candidata no compuesta y no existen dependencias transitivas problemáticas, afirmamos que está en forma normal de Boyce-Codd, por lo que la tabla está ya normalizada.

12. Sentencias SQL. Creación de tablas e inserción de tuplas

Creación de tablas

```
CREATE TABLE Actividad (
    ID_actividad INT PRIMARY KEY,
    Nombre VARCHAR2(50) CONSTRAINT nombre_no_nulo NOT NULL,
    Lugar VARCHAR2(50) CONSTRAINT lugar_no_nulo NOT NULL,
   Tipo INT CONSTRAINT tipo_no_nulo NOT NULL CONSTRAINT tipo_rango CHECK (Tipo BETWEEN 0 AND 5),
    Fecha DATE CONSTRAINT fecha_no_nula NOT NULL,
    Hora_inicio TIMESTAMP CONSTRAINT inicio_no_nulo NOT NULL,
    Hora_fin TIMESTAMP CONSTRAINT fin_no_nulo NOT NULL,
     Num_max INT CONSTRAINT max_no_nulo NOT NULL CONSTRAINT max_valido CHECK (Num_max > 0),
    ID_trabajador VARCHAR2(12),

    CONSTRAINT fk_dni_encargado FOREIGN KEY (ID_trabajador) REFERENCES Trabajador(ID)
CONSTRAINT check_horas_validas CHECK (Hora_inicio < Hora_fin)
);

CREATE TABLE Inscripcion (
    ID_cliente VARCHAR2(12),
    ID_actividad INT,

    PRIMARY KEY (ID_cliente, ID_actividad),
    FOREIGN KEY (ID_cliente) REFERENCES Cliente(ID),
    FOREIGN KEY (ID_actividad) REFERENCES Actividad(ID_actividad)
);
```

```
CREATE TABLE Cliente(
   ID VARCHAR(12) CONSTRAINT id_cliente PRIMARY KEY,
   Nombre VARCHAR(20) CONSTRAINT nombre NOT NULL,
   Apellidos VARCHAR(40) CONSTRAINT apellidos NOT NULL,
   NumeroTelefono VARCHAR(12) CONSTRAINT num_tfno NOT NULL,
   CorreoElectronico VARCHAR(50) CONSTRAINT correo NOT NULL,
   TipoCliente INT CONSTRAINT tipo CHECK ((TipoCliente>=0 AND TipoCliente<=5)),
   Importe DOUBLE
);


CREATE TABLE ClienteR (
   DNI VARCHAR(12),
   Telefono VARCHAR(20) NOT NULL,
   MetodoDePago VARCHAR(24) NOT NULL,
   CorreoElectronico VARCHAR(50) NOT NULL,
   CONSTRAINT PK_ClienteR PRIMARY KEY (DNI)
);

CREATE TABLE Reserva (
        CReserva INT PRIMARY KEY,
        FechaEntrada DATE NOT NULL,
        FechaSalida DATE NOT NULL,
        CheckIn NUMBER(1) DEFAULT 0,
        CheckOut NUMBER(1) DEFAULT 0,
        CONSTRAINT CHK_Fecha CHECK (FechaSalida > FechaEntrada),
        CONSTRAINT CHK_CheckOut CHECK (NOT (CheckOut = 1 AND CheckIn = 0))
);

CREATE SEQUENCE Reserva_seq;

CREATE OR REPLACE TRIGGER Reserva_bir
BEFORE INSERT ON Reserva
FOR EACH ROW
BEGIN
  SELECT Reserva_seq.NEXTVAL
  INTO   :new.CReserva
  FROM   dual;
END;


CREATE TABLE Habitacion (
        CHabitacion INT PRIMARY KEY,
        TipoHabitacion SMALLINT NOT NULL CHECK (TipoHabitacion BETWEEN 0 AND
99)
);

CREATE SEQUENCE Habitacion_seq;
```

```
CREATE OR REPLACE TRIGGER Habitacion_bir
BEFORE INSERT ON Habitacion
FOR EACH ROW
BEGIN
  SELECT Habitacion_seq.NEXTVAL
  INTO   :new.CHabitacion
  FROM   dual;
END;



CREATE TABLE RegistroHotel (
        CReserva INT,
        DNI VARCHAR2(12),
        CONSTRAINT PK_RegistroHotel PRIMARY KEY (CReserva),
        CONSTRAINT FK_RegistroHotel_CReserva FOREIGN KEY (CReserva)
REFERENCES Reserva(CReserva),
        CONSTRAINT FK_RegistroHotel_DNI FOREIGN KEY (DNI) REFERENCES
ClienteR(DNI),
        CONSTRAINT UQ_RegistroHotel_DNI UNIQUE (DNI)
);


CREATE TABLE ReservaHab (
        CReserva INT,
        CHabitacion INT,
        CONSTRAINT FK_ReservaHab_CReserva FOREIGN KEY (CReserva)
REFERENCES Reserva(CReserva),
        CONSTRAINT FK_ReservaHab_CHabitacion FOREIGN KEY (CHabitacion)
REFERENCES Habitacion(CHabitacion)
);


CREATE TABLE ReservaInv (
        CReserva INT,
        ID VARCHAR(20),
        CantidadReservada INT NOT NULL CHECK (CantidadReservada > 0),
        CONSTRAINT PK_ReservaInv PRIMARY KEY (CReserva, ID),
        CONSTRAINT FK_ReservaInv_CReserva FOREIGN KEY (CReserva)
REFERENCES Reserva(CReserva),
        CONSTRAINT FK_ReservaInv_ID FOREIGN KEY (ID) REFERENCES Inventario(ID)
);

CREATE TABLE Inventario(
    ID VARCHAR(20) CONSTRAINT id_inventario PRIMARY KEY,
    Categoria VARCHAR(30) CONSTRAINT categoria NOT NULL,
    Distribuidora VARCHAR(40),
```

```
    Cantidad INT CONSTRAINT cantidad NOT NULL CHECK (cantidad >= 0)
);

CREATE TABLE Trabajador(
   ID VARCHAR(12) CONSTRAINT id_clave_primaria PRIMARY KEY,
   Nombre VARCHAR(20) CONSTRAINT nombre_no_nulo NOT NULL,
   Apellidos VARCHAR(40) CONSTRAINT apellidos_no_nulo NOT NULL,
   FechaNacimiento DATE CONSTRAINT fecha_no_nula NOT NULL,
   Area VARCHAR(20) CONSTRAINT  area_no_nula NOT NULL
  );
```
He quitado la comprobación para la edad para que se encargue el disparador.

13. Inserción de tuplas

Tabla Actividad:

```
INSERT INTO Actividad (Nombre, Lugar, Tipo, Fecha, Hora_inicio, Hora_fin, Num_max,
ID_trabajador) VALUES ('Aquagym Avanzado', 'Piscina interior', 4, TO_DATE('2023-12-05',
'YYYY-MM-DD'), TO_TIMESTAMP(TO_CHAR(TO_DATE('2023-12-05', 'YYYY-MM-DD'),
'YYYY-MM-DD') || '08:00:00', 'YYYY-MM-DD HH24:MI:SS'),
TO_TIMESTAMP(TO_CHAR(TO_DATE('2023-12-05', 'YYYY-MM-DD'), 'YYYY-MM-DD') || '
10:00:00', 'YYYY-MM-DD HH24:MI:SS'), 30, '123456789012');

INSERT INTO Actividad (Nombre, Lugar, Tipo, Fecha, Hora_inicio, Hora_fin, Num_max,
ID_trabajador) VALUES ('Conferencia de Ximo Sanz', 'Sala de conferencias 1', 1,
TO_DATE('2023-12-10', 'YYYY-MM-DD'),
TO_TIMESTAMP(TO_CHAR(TO_DATE('2023-12-10', 'YYYY-MM-DD'), 'YYYY-MM-DD') || '
09:30:00', 'YYYY-MM-DD HH24:MI:SS'),
TO_TIMESTAMP(TO_CHAR(TO_DATE('2023-12-10', 'YYYY-MM-DD'), 'YYYY-MM-DD') || '
10:30:00', 'YYYY-MM-DD HH24:MI:SS'), 150, '345678901234');

INSERT INTO Actividad (Nombre, Lugar, Tipo, Fecha, Hora_inicio, Hora_fin, Num_max,
ID_trabajador) VALUES ('Parapente', 'Azotea', 5, TO_DATE('2023-12-15', 'YYYY-MM-DD'),
TO_TIMESTAMP(TO_CHAR(TO_DATE('2023-12-15', 'YYYY-MM-DD'), 'YYYY-MM-DD') || '
17:30:00', 'YYYY-MM-DD HH24:MI:SS'),
TO_TIMESTAMP(TO_CHAR(TO_DATE('2023-12-15', 'YYYY-MM-DD'), 'YYYY-MM-DD') || '
18:00:00', 'YYYY-MM-DD HH24:MI:SS'), 3, '345678901234');
```

También podemos crear registros sin indicar el DNI de empleado (no hay restricción de integridad NOT NULL):

```
INSERT INTO Actividad (Nombre, Lugar, Tipo, Fecha, Hora_inicio, Hora_fin, Num_max)
VALUES ('Taller de Fotografía', 'Planta Subterránea', 2, TO_DATE('2024-01-10',
'YYYY-MM-DD'),
TO_TIMESTAMP(TO_CHAR(TO_DATE('2024-01-10', 'YYYY-MM-DD'), 'YYYY-MM-DD') || '
12:00:00', 'YYYY-MM-DD HH24:MI:SS'),
TO_TIMESTAMP(TO_CHAR(TO_DATE('2024-01-10', 'YYYY-MM-DD'), 'YYYY-MM-DD') || '
14:00:00', 'YYYY-MM-DD HH24:MI:SS'), 15);
```

Tabla Inscripción:

INSERT INTO Inscripcion (ID_cliente, ID_actividad) VALUES ('C001', 1);
INSERT INTO Inscripcion (ID_cliente, ID_actividad) VALUES ('C002', 1);
INSERT INTO Inscripcion (ID_cliente, ID_actividad) VALUES ('C003', 2);
INSERT INTO Inscripcion (ID_cliente, ID_actividad) VALUES ('C004', 2);
INSERT INTO Inscripcion (ID_cliente, ID_actividad) VALUES ('C002', 3);

Tabla Cliente:
INSERT INTO Cliente (ID, Nombre, Apellidos, NumeroTelefono, CorreoElectronico, TipoCliente, Importe)
VALUES ('C001', 'Ana', 'García Ruiz', '600123456', 'ana.garcia@mail.com', 3, 1200.00);

INSERT INTO Cliente (ID, Nombre, Apellidos, NumeroTelefono, CorreoElectronico, TipoCliente, Importe)
VALUES ('C002', 'Luis', 'Martínez López', '610234567', 'luis.martinez@mail.com', 2, 800.50);

INSERT INTO Cliente (ID, Nombre, Apellidos, NumeroTelefono, CorreoElectronico, TipoCliente, Importe)
VALUES ('C003', 'María', 'Pérez Jiménez', '620345678', 'maria.perez@mail.com', 5, 500.00);

INSERT INTO Cliente (ID, Nombre, Apellidos, NumeroTelefono, CorreoElectronico, TipoCliente, Importe)
VALUES ('C004', 'Jorge', 'Fernández Álvarez', '630456789', 'jorge.fernandez@mail.com', 1, 1500.25);

INSERT INTO Cliente (ID, Nombre, Apellidos, NumeroTelefono, CorreoElectronico, TipoCliente, Importe)
VALUES ('C005', 'Sofía', 'Gómez Torres', '640567890', 'sofia.gomez@mail.com', 4, 950.75);

Tabla_ClienteR
INSERT INTO ClienteR (DNI, Telefono, MetodoDePago, CorreoElectronico) VALUES ('12345678A', '123456789', 'ES9100123456789012345678', 'cliente1@email.com');
INSERT INTO ClienteR (DNI, Telefono, MetodoDePago, CorreoElectronico) VALUES ('23456789B', '987654321', 'ES9100987654321098765432', 'cliente2@email.com');
INSERT INTO ClienteR (DNI, Telefono, MetodoDePago, CorreoElectronico) VALUES ('34567890C', '112233445', 'ES9100567891234567890123', 'cliente3@email.com');

Tabla Reserva
INSERT INTO Reserva (CReserva, FechaEntrada, FechaSalida) VALUES (1, TO_DATE('2023-12-10', 'YYYY-MM-DD'), TO_DATE('2023-12-15', 'YYYY-MM-DD'));
INSERT INTO Reserva (CReserva, FechaEntrada, FechaSalida) VALUES (2, TO_DATE('2023-12-16', 'YYYY-MM-DD'), TO_DATE('2023-12-20', 'YYYY-MM-DD'));
INSERT INTO Reserva (CReserva, FechaEntrada, FechaSalida) VALUES (3, TO_DATE('2023-12-21', 'YYYY-MM-DD'), TO_DATE('2023-12-25', 'YYYY-MM-DD'));

NSERT INTO Reserva (CReserva, FechaEntrada, FechaSalida) VALUES (4, TO_DATE('2023-12-21', 'YYYY-MM-DD'), TO_DATE('2023-12-25', 'YYYY-MM-DD'));

Tabla Habitación

```
INSERT INTO Habitacion(chabitacion, tipohabitacion) VALUES (1, 0);
INSERT INTO Habitacion(chabitacion, tipohabitacion) VALUES (2, 0);
INSERT INTO Habitacion(chabitacion, tipohabitacion) VALUES (3, 0);
INSERT INTO Habitacion(chabitacion, tipohabitacion) VALUES (4, 0);
INSERT INTO Habitacion(chabitacion, tipohabitacion) VALUES (5, 0);

INSERT INTO Habitacion(chabitacion, tipohabitacion) VALUES (6, 1);
INSERT INTO Habitacion(chabitacion, tipohabitacion) VALUES (7, 1);
INSERT INTO Habitacion(chabitacion, tipohabitacion) VALUES (8, 1);
INSERT INTO Habitacion(chabitacion, tipohabitacion) VALUES (9, 1);
INSERT INTO Habitacion(chabitacion, tipohabitacion) VALUES (10, 1);

INSERT INTO Habitacion(chabitacion, tipohabitacion) VALUES (11, 2);
INSERT INTO Habitacion(chabitacion, tipohabitacion) VALUES (12, 2);
INSERT INTO Habitacion(chabitacion, tipohabitacion) VALUES (13, 2);
INSERT INTO Habitacion(chabitacion, tipohabitacion) VALUES (14, 2);
INSERT INTO Habitacion(chabitacion, tipohabitacion) VALUES (15, 2);
```

Tabla ReservaHab

```
INSERT INTO ReservaHab (CReserva, CHabitacion) VALUES (1, 1);
INSERT INTO ReservaHab (CReserva, CHabitacion) VALUES (2, 6);
INSERT INTO ReservaHab (CReserva, CHabitacion) VALUES (3, 11);
INSERT INTO ReservaHab (CReserva, CHabitacion) VALUES (4, 12);
```

RegistroHotel

```
INSERT INTO RegistroHotel (CReserva, DNI) VALUES (1, '12345678A');
INSERT INTO RegistroHotel (CReserva, DNI) VALUES (2, '23456789B');
INSERT INTO RegistroHotel (CReserva, DNI) VALUES (3, '34567890C');
INSERT INTO RegistroHotel (CReserva, DNI) VALUES (4, '12345678A');
```

Tabla ReservaInv

```
INSERT INTO ReservaInv (CReserva, ID, CantidadReservada) VALUES (1, 'Pala de padel', 1);
INSERT INTO ReservaInv (CReserva, ID, CantidadReservada) VALUES (1, 'Secador', 2);
INSERT INTO ReservaInv (CReserva, ID, CantidadReservada) VALUES (2, 'Secador', 1);
```

Tabla Trabajador:

INSERT INTO Trabajador (ID, Nombre, Apellidos, FechaNacimiento, Area, FechaFinContrato) VALUES ('123456789012', 'Juan Andres', 'Mauricio Martin', '03-07-2001', 'Administración', '11-01-2025');

INSERT INTO Trabajador (ID, Nombre, Apellidos, FechaNacimiento, Area, FechaFinContrato) VALUES ('234567890123', 'Dani', 'Alconchel Calvo', '18-11-2001', 'Mantenimiento', '15-06-2025');

INSERT INTO Trabajador (ID, Nombre, Apellidos, FechaNacimiento, Area, FechaFinContrato) VALUES ('345678901234', 'Luis', 'Crespo Ortiz', '06-04-1931', 'Hostelería', '31-12-2024');

INSERT INTO Trabajador (ID, Nombre, Apellidos, FechaNacimiento, Area, FechaFinContrato) VALUES ('456789012345', 'Ximo', 'San Olivares', '11-11-2002', 'Recepción', '10-05-2026');

Tabla Inventario:
INSERT INTO Inventario (ID, Categoria, Distribuidora, Cantidad) VALUES
(‘Pala de padel’, ‘Deportes’, ‘Decathlon’, 30);

INSERT INTO Inventario (ID, Categoria, Distribuidora, Cantidad)
VALUES (‘Toalla’, ‘Aseo’, ‘Tissa Textil’, 120);

INSERT INTO Inventario (ID, Categoria, Distribuidora, Cantidad)
VALUES (‘Sábana’, ‘Cama’, ‘Tissa Textil’, 150);

INSERT INTO Inventario (ID, Categoria, Distribuidora, Cantidad)
VALUES (‘Secador’, ‘Electrodoméstico’, ‘Corte Inglés‘, 100);

14. Disparadores.

Gestión de actividades
Disparador para evitar que un empleado trabaje en dos actividades a la vez:
Se examina toda la tabla de actividades en busca de una que esté asignada al empleado asociado a la actividad que se está creando o actualizando, y que se desarrolle a la vez en el tiempo que dicha actividad. Si esto ocurre, saltará el disparador y no se realizará la inserción/actualización. Para la implementación, resulta necesario realizar una copia auxiliar de la tabla en una estructura de datos de PL/SQL.

TRIGGER actividad_overlap_check
FOR INSERT OR UPDATE ON ACTIVIDAD
COMPOUND TRIGGER

```
  TYPE t_actividad_type IS RECORD (
    ID_TRABAJADOR ACTIVIDAD.ID_TRABAJADOR%TYPE,
    FECHA ACTIVIDAD.FECHA%TYPE,
    HORA_INICIO ACTIVIDAD.HORA_INICIO%TYPE,
    HORA_FIN ACTIVIDAD.HORA_FIN%TYPE,
    ID_ACTIVIDAD ACTIVIDAD.ID_ACTIVIDAD%TYPE
  );
  TYPE t_actividad_tab IS TABLE OF t_actividad_type INDEX BY PLS_INTEGER;
  v_actividades t_actividad_tab;
  v_count NUMBER;
  v_index PLS_INTEGER := 0;

  AFTER EACH ROW IS
  BEGIN
    v_index := v_index + 1;
    v_actividades(v_index).ID_TRABAJADOR := :NEW.ID_TRABAJADOR;
    v_actividades(v_index).FECHA := :NEW.FECHA;
    v_actividades(v_index).HORA_INICIO := :NEW.HORA_INICIO;
    v_actividades(v_index).HORA_FIN := :NEW.HORA_FIN;
    v_actividades(v_index).ID_ACTIVIDAD := :NEW.ID_ACTIVIDAD;
  END AFTER EACH ROW;

  AFTER STATEMENT IS
  BEGIN
    FOR i IN 1..v_index LOOP
      SELECT COUNT(*)
      INTO v_count
      FROM ACTIVIDAD
      WHERE ID_TRABAJADOR = v_actividades(i).ID_TRABAJADOR
      AND FECHA = v_actividades(i).FECHA
      AND NOT (
        HORA_FIN <= v_actividades(i).HORA_INICIO
        OR HORA_INICIO >= v_actividades(i).HORA_FIN
      )
      AND ID_ACTIVIDAD != NVL(v_actividades(i).ID_ACTIVIDAD, -1);

      IF v_count > 0 THEN
            RAISE_APPLICATION_ERROR(-20002, 'El trabajador ya tiene una actividad
programada en este horario.');
      END IF;
    END LOOP;
  END AFTER STATEMENT;

END actividad_overlap_check;
```

Disparador para manejar la asignación única y autoincremental de un ID entero a cada actividad nueva:

Este disparador resulta necesario en Oracle si queremos implementar identificadores autoincrementales.

```
TRIGGER actividad_trigger
 BEFORE INSERT ON Actividad
 FOR EACH ROW
 BEGIN
 SELECT actividad_seq.NEXTVAL INTO :new.ID_actividad FROM dual;
 END;
```

Gestión de clientes

Disparador que comprueba si un cliente a eliminar tiene deudas pendientes:

```
CREATE OR REPLACE TRIGGER deudas_check
        BEFORE
        DELETE ON Cliente
        FOR EACH ROW
        BEGIN
                IF :NEW.importe > 0 THEN
                raise_application_error(-19998, 'No se puede eliminar a un cliente con deudas.');
                END IF;
        END;
```

Disparador que comprueba si un cliente puede reservar cierta actividad:

```
CREATE OR REPLACE TRIGGER rango_actividad_cliente
        BEFORE
        INSERT OR UPDATE OF ID_Actividad ON Inscripcion
        FOR EACH ROW
        DECLARE
                rangoCliente INT,
                rangoActividad INT;
        BEGIN
                SELECT TipoCliente INTO rangoCliente FROM Cliente WHERE ID == :NEW.ID_cliente
                SELECT Tipo INTO rangoActividad FROM Actividad WHERE ID_actividad == :NEW.ID_actividad
                IF (rangoCliente<rangoActividad) THEN
                raise_application_error(-19999, 'Un cliente no puede reservar una actividad de un rango superior.');
                END IF;
        END;
```

Gestión de actividades

```
TRIGGER actividad_overlap_check
```

```
FOR INSERT OR UPDATE ON ACTIVIDAD
COMPOUND TRIGGER

   TYPE t_actividad_type IS RECORD (
      ID_TRABAJADOR ACTIVIDAD.ID_TRABAJADOR%TYPE,
      FECHA ACTIVIDAD.FECHA%TYPE,
      HORA_INICIO ACTIVIDAD.HORA_INICIO%TYPE,
      HORA_FIN ACTIVIDAD.HORA_FIN%TYPE,
      ID_ACTIVIDAD ACTIVIDAD.ID_ACTIVIDAD%TYPE
   );
   TYPE t_actividad_tab IS TABLE OF t_actividad_type INDEX BY PLS_INTEGER;
   v_actividades t_actividad_tab;
   v_count NUMBER;
   v_index PLS_INTEGER := 0;

   AFTER EACH ROW IS
   BEGIN
      v_index := v_index + 1;
      v_actividades(v_index).ID_TRABAJADOR := :NEW.ID_TRABAJADOR;
      v_actividades(v_index).FECHA := :NEW.FECHA;
      v_actividades(v_index).HORA_INICIO := :NEW.HORA_INICIO;
      v_actividades(v_index).HORA_FIN := :NEW.HORA_FIN;
      v_actividades(v_index).ID_ACTIVIDAD := :NEW.ID_ACTIVIDAD;
   END AFTER EACH ROW;

   AFTER STATEMENT IS
   BEGIN
      FOR i IN 1..v_index LOOP
         SELECT COUNT(*)
         INTO v_count
         FROM ACTIVIDAD
         WHERE ID_TRABAJADOR = v_actividades(i).ID_TRABAJADOR
         AND FECHA = v_actividades(i).FECHA
         AND NOT (
            HORA_FIN <= v_actividades(i).HORA_INICIO
            OR HORA_INICIO >= v_actividades(i).HORA_FIN
         )
         AND ID_ACTIVIDAD != NVL(v_actividades(i).ID_ACTIVIDAD, -1);

         IF v_count > 0 THEN
                  RAISE_APPLICATION_ERROR(-20002, 'El trabajador ya tiene una actividad
programada en este horario.');
         END IF;
      END LOOP;
   END AFTER STATEMENT;

END actividad_overlap_check;
```

Gestión de personal

Disparador que comprueba que la edad de un trabajador sea válida

```
TRIGGER trabajador_age_check
BEFORE INSERT OR UPDATE ON Trabajador
FOR EACH ROW
BEGIN
        IF (MONTHS_BETWEEN(SYSDATE, :new.FechaNacimiento) / 12) < 16 THEN
        RAISE_APPLICATION_ERROR(-20000, 'La edad del trabajador debe ser al menos 16 años.');
        END IF;
END;
```

Disparador que comprueba que la fecha de fin de contrato es posterior a la del día de hoy:

```
TRIGGER verificar_fecha_fin_contrato
BEFORE INSERT OR UPDATE OF FechaFinContrato ON Trabajador
FOR EACH ROW
DECLARE
   v_fecha_hoy DATE := TRUNC(SYSDATE);
BEGIN
   -- Verificar que la fecha de fin de contrato sea mayor que la fecha actual
   IF :NEW.FechaFinContrato <= v_fecha_hoy THEN
      RAISE_APPLICATION_ERROR(-20001, 'La fecha de fin de contrato debe ser posterior a la fecha actual.');
   END IF;
END;
```

Disparador que pone a null a los trabajadores de las actividades asociadas y pone ID a null:

```
TRIGGER Trabajador_Cascada
BEFORE DELETE ON Trabajador FOR EACH ROW
BEGIN
     -- Poner a NULL el ID_Trabajador de las actividades asociadas al trabajador a ser borrado
   UPDATE Actividad
   SET ID_Trabajador = NULL
   WHERE ID_Trabajador = :OLD.ID;
END;
```

Gestión de inventario

Disparador que comprueba si un objeto a eliminar, está reservado:

```
CREATE OR REPLACE TRIGGER tr_borrar_objeto_reservado
   BEFORE DELETE ON Inventario
   FOR EACH ROW
BEGIN
   IF :OLD.ID IS NOT NULL THEN
      DELETE FROM ReservaInv WHERE ID = :OLD.ID;
   END IF;
```

```
END tr_borrar_objeto_reservado;
```

Disparador para comprobar que, al cambiar la cantidad de un objeto, esta nueva cantidad no es menor que la suma de cantidades reservadas:

```
CREATE OR REPLACE TRIGGER tr_verificar_actualizacion
   BEFORE UPDATE ON Inventario
   FOR EACH ROW
DECLARE
   total_reservas NUMBER;
BEGIN
   -- Calcular el total de reservas para el objeto
   SELECT NVL(SUM(CantidadReservada), 0) INTO total_reservas
   FROM ReservaInv
   WHERE ID = :NEW.ID;

   -- Verificar que la nueva cantidad no sea inferior al total de reservas
   IF :NEW.Cantidad < total_reservas THEN
        RAISE_APPLICATION_ERROR(-20001, 'La nueva cantidad no puede ser inferior al total de reservas.');
   END IF;
END tr_verificar_actualizacion;
```

Disparador que comprueba que al insertar un nuevo objeto en reserva inventario que la cantidad sea correcta, inclusive mirando las fechas seleccionadas:

```
CREATE OR REPLACE TRIGGER trg_reservar_inventario
BEFORE INSERT ON ReservaInv
FOR EACH ROW
DECLARE
        v_cantidad_disponible INT;
        v_cantidad_reservada INT;
BEGIN
        -- Obtener la cantidad disponible del objeto en Inventario
        SELECT Cantidad INTO v_cantidad_disponible FROM Inventario WHERE ID = :NEW.ID;

        -- Calcular la cantidad total ya reservada que coincide en fechas
        SELECT COALESCE(SUM(ri.CantidadReservada), 0) INTO v_cantidad_reservada
        FROM ReservaInv ri
        INNER JOIN Reserva r ON ri.CReserva = r.CReserva
        WHERE ri.ID = :NEW.ID
        AND (r.FechaEntrada <= (SELECT FechaSalida FROM Reserva WHERE CReserva = :NEW.CReserva)
        AND r.FechaSalida >= (SELECT FechaEntrada FROM Reserva WHERE CReserva = :NEW.CReserva));

        -- Verificar si la nueva reserva supera la cantidad disponible
```

```
        IF (v_cantidad_reservada + :NEW.CantidadReservada) > v_cantidad_disponible
THEN
        RAISE_APPLICATION_ERROR(-20002, 'La cantidad reservada supera la cantidad
disponible en el inventario.');
        END IF;
EXCEPTION
        WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20001, 'El objeto no existe en el inventario.');
END;
```

Subsistema de reservas de habitaciones:

- Cancelar reserva. Disparador que se encarga de eliminar en cadena para mantener la consistencia de la bd:

```
CREATE OR REPLACE TRIGGER cancelar_reserva
BEFORE DELETE ON Reserva
FOR EACH ROW
BEGIN
        -- Verificar si el CheckIn está a TRUE (1 representa TRUE)
        IF :OLD.CheckIn = 1 THEN
        RAISE_APPLICATION_ERROR(-20001, 'No se puede cancelar una reserva con
Check-In realizado.');
        ELSE
        -- Eliminar las tuplas relacionadas en la tabla RegistroHotel
        DELETE FROM RegistroHotel WHERE CReserva = :OLD.CReserva;

        -- Eliminar las tuplas relacionadas en la tabla ReservaHab
        DELETE FROM ReservaHab WHERE CReserva = :OLD.CReserva;

        -- Eliminar las tuplas relacionadas en la tabla ReservaInv
        DELETE FROM ReservaInv WHERE CReserva = :OLD.CReserva;

        -- Continuar con la eliminación en la tabla Reserva
        END IF;
EXCEPTION
        WHEN OTHERS THEN
        -- Manejar cualquier otra excepción si es necesario
        RAISE;
END;
```

15. Elección de software.

Hemos decidido utilizar Python para el desarrollo de nuestro Sistema de Información debido a que, tras completar el seminario inicial, nos familiarizamos ampliamente con las capacidades que ofrece la biblioteca pyodbc en términos de interacción con bases de datos. Además, ya contábamos con el conjunto completo de herramientas necesarias para aprovechar estas capacidades, lo que nos llevó a concluir que no era necesario explorar otros lenguajes de programación. Especialmente, hemos de destacar la facilidad que ofrece *pyodbc* para manejar las excepciones devueltas por el SGDB. No obstante, en el transcurso

de la elaboración de la práctica, encontramos ciertas dificultades técnicas con Python, particularmente en la implementación de disparadores. Esto nos motivó a integrar SQLDeveloper en nuestro flujo de trabajo, una herramienta que nos permitió interactuar directamente con la base de datos y facilitó significativamente la implementación de estas funciones más complejas del sistema.

Como comentario, tuvimos que cambiar el driver de ODBC, porque el anterior expiró la prueba gratuita (Instalación Nuevo Driver).