**Informe Final Hotel de los Andes**

# Diseño de la Base de Datos

## Análisis de la carga de trabajo

### Entidades

En esta sección se encuentra una lista de las entidades de la aplicación con sus atributos simples

* Hotel: id, nombre, numEstrellas
* Habitación: id, disponible
* TipoHabitacion: id, nombre, capacidad, precioNoche
* Reserva: id, idCliente, idHabitacion, fechaEntrada, fechaSalida, cobroTotal, numPersonas
* Cliente: nombre, documento, tipoDocumento, metodoPago
* ServicioBasico: id, nombre, capacidad, costo, horaApertura, horaCierre
* Producto: id, nombre, precio, tipoProducto, cantidadDisponible
* Piscina: id, profundidad
* Lavanderia: id, costoPorPrenda, costoPorParZapatos
* Tienda: id, tipo, productos
* RestauranteBar: id, estilo, platos
* Plato: id, descripcion
* Prestamo: id, idProducto
* Gimnasio: id, maquinas
* Sala: id, tipo, equipos
* SPA: id, servicios
* ServicioSPA: id, duración
* ConsumoServicioCliente: idReserva, idServicio, fecha, descripción, costoServicio
* ClienteConsumeProducto: idReserva, idProducto, fecha, descripción, cantidad

Las entidades Piscina, Lavandería, Tienda, RestauranteBar, Gimnasio, Sala y SPA heredan de ServicioBasico, por lo que su id es el id de uno de los servicios básicos, así que no puede existir uno de ellos sin su respectivo ServicioBasico. Al mismo tiempo, la mayoría de estas entidades tienen una relación con Producto que muestran los productos que ofrecen o los equipos que tienen, etc.

Las Entidades Plato y ServicioSPA heredan de Producto, dado que estos son casos específicos de productos.

Sin embargo, como estamos manejado una base de datos no relacional, las referencias no funcionan de la misma manera. En el apartado de Selección de esquema de asociación encontrara la explicación de como se manejaran las referencias para cada entidad.

### Cantidad de registros

Para hacer este análisis debemos tener en cuenta los estimados mencionados en el enunciado.

* Hotel: Seria uno solo, pues según el enunciado, la aplicación debería ser para que un hotel maneje su operación, por lo que solo es uno
* Habitaciones: Según el enunciado una cadena hotelera grande tiene aproximadamente 200 habitaciones, pero teniendo en cuenta que puede que se abran más o que cierren por temporada digamos que la cantidad en la base de datos podría ser de hasta 230 habitaciones por cadena.
* TipoHabitacion: En el enunciado dice que aproximadamente 20, así que aproximadamente 25 parece adecuado por las mismas razones que la cantidad de habitaciones, además los tipos de habitaciones son más volátiles que las habitaciones, así que la creación o eliminación de nuevos tipos no es muy inusual.
* Reserva: El volumen de registros podría ser de hasta 20000 por año siendo un poco exagerado con la cantidad.
* Cliente: Aunque en el enunciado no se menciona nada sobre la cantidad de clientes de una cadena, podemos estimar que serían de 5000 a 10000 teniendo en cuenta la cantidad de Reservas y asumiendo que un cliente no se queda más de dos veces por año en un hotel.
* ServicioBasico: Este incluye Piscina, Lavanderia, Tienda, RestauranteBar, Prestamo, Gimnasio, Sala y SPA. De acuerdo con el enunciado, se aproxima que la cantidad de servicios es unos 35, así que podrían ser de 8 a 10 por cada tipo de servicio.
* Producto: El enunciado no menciona nada acerca de los productos, pero teniendo en cuenta que los productos incluyen tanto los productos que ofrecen los servicios del hotel como la utilería de las habitaciones como camas, minibares, yacusis, etc., se podría Mostrar un aproximado de unos 2000 productos incluyendo todo lo anterior mencionado.
* Plato y ServicioSPA: La cantidad de registros está limitada por la cantidad de productos, pero supongamos, en el caso de los platos, que un restaurante o bar ofrece en su carta de 30 a 50 productos y que, en el caso de servicio de SPA, un SPA ofrece a lo mucho 15 a 20. Siendo así, aproximamos que la cantidad de platos en total podría ser entre 240 y 400, y la cantidad de servicios en un SPA son entre 120 y 160.
* ClienteConsumeProducto y ConsumoServicioCliente: Para estos registros se estima que pueden ser unos 85000 anuales para cada tipo de consumo.

### Análisis y Cuantificación de Operaciones

En nuestra aplicación tener el CRUD de una salida del hotel o una entrada al mismo es imposible, dado que esto se connota con la disponibilidad de una habitación, así que “registrar” la entrada al hotel es cambiar el atributo “disponible” de una habitación a false, y, en el caso de la salida, cambiarlo a true, por lo cual no se considera para tener un CRUD.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entidades | Operaciones | Información Necesaria | Tipo | Frecuencia |
| TipoHabitacion | Mostrar todos los tipos de habitación en un hotel | El hotel del cual se quieren consulta los tipos | Lectura |  |
| TipoHabitacion | Borrar, Registrar, Actualizar un tipo de habitación | La información del tipo de habitación a borrar, registrar o modificar | Escritura |  |
| Habitación | Mostrar todas las habitaciones de hotel | El hotel del cual se quieren las habitaciones | Lectura |  |
| Habitación | Borrar, Registrar, Actualizar una habitación | La información de la habitación a borrar, registrar o modificar | Escritura |  |
| Habitación | Mostrar el dinero recolectado por cada habitación en el último año | La información de cada habitación y la información de las reservas de cada habitación | Lectura |  |
| Habitación | Mostrar el índice de ocupación de cada habitación en el último año corrido | La información de las habitaciones ocupadas en el último año y la información de las reservas de dichas habitaciones | Lectura |  |
| ServicioBasico | Mostrar todos los servicios de un hotel | El hotel del cual se quieren ver los servicios | Lectura |  |
| ServicioBasico | Borrar, Registrar o Actualizar un servicio | La información del servicio a borrar, registrar o modificar. | Escritura |  |
| Reserva | Mostrar todas las reservas de un hotel | El hotel del cual se quieren consultar las reservas | Lectura |  |
| Reserva | Borrar, Registrar o Actualizar una reserva | La información de la reserva a borrar, registrar o modificar | Escritura |  |
| ConsumoServicioCliente | Mostrar todos los consumos de servicios que se han hecho en un hotel | El hotel sobre el cual se hará la consulta | Lectura |  |
| ConsumoServicioCliente | Borrar, Registrar o Actualizar el consumo de un servicio | La información del consumo de servicio que se quiere borrar, registrar o modificar | Escritura |  |
| ClienteConsumeProducto | Mostrar todos los consumos de productos que se han hecho en un hotel | El hotel sobre el cual se hará la consulta | Lectura |  |
| ClienteConsumeProducto | Borrar, Registrar o Actualizar el consumo de un producto | La información del consumo del producto que se quiere borrar, registrar o modificar | Escritura |  |

## Descripción de las entidades

* Hotel: Hotel tiene una relación con ServicioBasico esta relación es de uno a muchos, dado que un hotel tiene varios servicios o no tiene, pero cada servicio debe pertenecer a un único hotel. En iteraciones anteriores se había modelado para que un servicio pudiera pertenecer a varios hoteles, pero al realizar de nuevo el análisis del caso nos dimos cuenta de que tiene más sentido que un servicio tenga un solo hotel, pues al ser un servicio algo físico, sería muy poco probable que se ofreciera de la misma manera en otro hotel, además a un hotel no le importa si un servicio se ofrece en otro hotel que no sea el propio, así que no tiene sentido que sea muchos a muchos.

Hotel también tiene una relación de uno a muchos con Habitación, siendo que una habitación tiene un hotel y un hotel muchas habitaciones.

* Habitación: Esta entidad tiene tres relaciones: una relación de uno a muchos con Hotel, teniendo una habitación un hotel y un hotel muchas habitaciones, una relación de uno a muchos con Reserva, teniendo una reserva una habitación, pero una habitación pudiendo estar en varias reservas o en ninguna; una relación de uno a muchos con TipoHabitacion, teniendo una habitación un tipo, pero cada tipo varias habitaciones.
* TipoHabitacion: Tiene dos relaciones: una relación de uno a muchos con Habitación, teniendo cada habitación un tipo y cada tipo varias habitaciones o no tiene; una relación de muchos a muchos con productos donde cada tipo tiene varios productos y cada producto puede tener varios tipos de habitación o no tener.
* Reserva: Tiene cuatro relaciones: una relación de uno a muchos con Cliente, siendo que un cliente puede tener varias reservas o ninguna, pero una reserva solo puede tener un cliente; una relación muchos a muchos con Producto llamada ClienteConsumeProducto donde una reserva puede o no tener un producto y un producto puede o no tener una reserva, esta relación representa el consumo de un cliente por parte de un cliente, además cuenta con atributos adicionales para modelar de mejor manera un consumo; una relación de muchos a muchos con ServicioBasico que representa el consumo de un servicio por parte de un cliente y funciona de la misma manera que la relación con productos; una relación de uno a muchos con Habitación, dado que una habitación puede tener o no muchas reservas, pero una reserva solo tiene una habitación.
* Cliente: Tiene una relación de uno a muchos con reserva, pues un cliente puede tener muchas reservas o ninguna, pero una reserva debe tener un cliente.
* ServicioBasico: Tiene dos relaciones principales: una relación de uno a muchos con Hotel donde un hotel tiene varios o no tiene servicios y un servicio pertenece a un hotel; una relación de muchos a muchos con Reserva llamada ConsumoServicioCliente que modela los consumos de servicios por parte de un cliente. Adicionalmente, esta entidad también tiene una relación de herencia, o una relación de uno a uno, con cada servicio específico, siendo estos Piscina, Lavanderia, Tienda, RestauranteBar, Prestamo, Gimnasio, Sala y SPA, por lo que esta relación no será mencionada cuando se describan las relaciones de los servicios.
* Producto: Esta entidad tiene dos relaciones principales: una relación de muchos a muchos con Reserva que modela el consumo de productos por parte de un cliente; una relación de muchos a muchos con TipoHabitacion donde un tipo tiene varios productos y un producto puede o no tener varios tipos de habitación. Además de las relaciones anterior mencionadas, Producto tiene una relación de muchos a muchos con todos los servicios, con excepción de Piscina y Lavandería, que modelan los diferentes productos que ofrecen como por ejemplo los productos de una tienda, los platos de un restaurante, las bebidas de un bar, etc. Hay dos casos especiales que son las relaciones de herencia con Plato y ServicioSPA dado que estas dos son casos de productos que contienen más características que las ofrecidas por Producto, por lo cual son necesarias.
* Piscina y Lavanderia: Tienen una relación uno a uno con ServicioBasico que es la que modela la herencia.
* Tienda, Gimnasio y Sala: Tienen una relación de muchos a muchos con Producto que modela los productos en caso de la tienda, las maquinas en caso del gimnasio, y los equipos en caso de la sala.
* RestauranteBar: Tiene una relación de muchos a muchos con Plato que como su nombre lo indica, representa los platos que ofrece cada Restaurante o las bebidas de cada Bar, cual sea el caso.
* SPA: Tiene una relación de muchos a muchos con ServicioSPA que representa los servicios que ofrece cada SPA además de los productos.

### Selección de esquema de asociación

* Hotel: Para esta entidad no importa mucho que tipo de esquema se use pues en el marco de la aplicación no es necesario hacer consultas sobre el recurso hotel. Sin embargo, si se debieran realizar consultas, sería bueno usar un esquema embebido para los servicios, puesto que la cantidad de registros no es muy densa. Mientras que para las habitaciones dependiendo de la cadena hotelera, sería mejor un modelo referenciado debido a la cantidad de registros que hay para este recurso, ya que, si se realizara un modelo embebido, habría muchos casos de inconsistencia en los datos dado que modificar varios registros en la colección de Habitaciones haría que tuviéramos que modificarlos en la colección de Hotel.
* Habitación: Para esta entidad la mejor opción es un modelo embebido para el caso de TipoHabitacion y Hotel, pues si esta se modifica no es complicado realizar los cambios, además cuando traemos las habitaciones es de utilidad traer sus tipos y el hotel al que pertenecen. Por otro lado, en el caso de reservas, sería mejor opción usar un modelo referenciado dado que puede que los datos sean demasiados, además de que no es muy importante traer todas las reservas de una habitación cada vez que se consulte esta última, también es mejor este modelo ya que por motivos de las consultas que se harán en la aplicación, Reserva debe ser su propia colección.
* TipoHabitacion: Creemos que el modelo embebido es el mejor, dado que es útil tener las habitaciones de cada tipo de habitación al igual que su dotación, y la densidad de datos lo permite así.
* Reserva: En este caso para la relación con Cliente y con Habitación usaremos un modelo embabido debido a la simplicidad de los datos y a que es útil que con cada consulta de una reserva se consulten también los otros dos recursos implicados. En el caso de los consumos, creemos que la densidad de datos permite la utilización de un modelo embebido también, dado que no creemos que por reserva hayan más de unos 30 o 40 consumos y eso es ser exagerado incluso.
* Cliente: Para esta entidad es útil tener las reservas embebidas dentro del cliente, ya que es útil saber ese dato al consultar un cliente.
* ServicioBasico: Para esta entidad lo ideal sería embeber el servicio específico al servicio básico debido a que solo es uno así que es buena idea. En el caso de hotel, se usará un modelo embebido pues al ser uno solo es una buena idea. En el caso de los consumos, se tendrá un modelo referenciado, ya que la cantidad de consumos que puede tener un servicio es muy alta, pues es la cantidad total de veces que se ha consumido, así que no es conveniente tener un modelo embebido en este caso.
* Producto: En esta entidad hay muchas relaciones, sin embargo, consideramos que no es importante modelar las relaciones desde esta entidad, dado que no es necesario hacer consultas sobre los Productos, además de que no es importante saber con qué objetos está relacionado los productos, es decir, a los productos no les importa con quien esté relacionado, así que esta colección tendrá solamente los atributos básicos de la entidad y los atributos descripción y duración correspondientes a los platos y a los servicios de un spa respectivamente.
* Piscina y Lavanderia: Estas entidades tendrán embebido su respectivo ServicioBasico
* Gimnasio, Sala, RestauranteBar, Tienda y Prestamo: Estas entidades tendrán embebidas sus máquinas, en el caso de Gimnasio, sus equipos, en el caso de Sala, sus platos, en el caso de RestauranteBar, sus productos, en el caso de Tienda, o su objeto, en el caso de Prestamo, además de tener sus atributos específicos para cada uno.
* ConsumoServicioCliente y ClienteConsumeProducto: Para estas entidades se usará un modelo embebido para tener tanto la reserva como el servicio o producto correspondiente al consumo.

Carga de Datos

Colección cliente

* db.clientes.insertMany([

{ nombre: "Juan Pérez", documento: 12345678, tipoDocumento: "CC", metodoPago: "Efectivo", reservas: [] },

* { nombre: "Laura Martínez", documento: 87654321, tipoDocumento: "CC", metodoPago: "Tarjeta", reservas: [] },
* { nombre: "Carlos López", documento: 23456789, tipoDocumento: "TI", metodoPago: "Efectivo", reservas: [] },
* { nombre: "Diana Gómez", documento: 98765432, tipoDocumento: "CC", metodoPago: "Tarjeta", reservas: [] },
* { nombre: "Mario Ruiz", documento: 34567890, tipoDocumento: "TI", metodoPago: "Efectivo", reservas: [] },
* { nombre: "Sofía Hernández", documento: 09876543, tipoDocumento: "CC", metodoPago: "Tarjeta", reservas: [] },
* { nombre: "Andrés Jiménez", documento: 45678901, tipoDocumento: "TI", metodoPago: "Efectivo", reservas: [] },
* { nombre: "Ana Torres", documento: 10987654, tipoDocumento: "CC", metodoPago: "Tarjeta", reservas: [] },
* { nombre: "Luisa Fernández", documento: 56789012, tipoDocumento: "TI", metodoPago: "Efectivo", reservas: [] },
* { nombre: "David Ramírez", documento: 21098765, tipoDocumento: "CC", metodoPago: "Tarjeta", reservas: [] },
* { nombre: "Isabel Castro", documento: 67890123, tipoDocumento: "TI", metodoPago: "Efectivo", reservas: [] },
* { nombre: "Oscar Morales", documento: 32109876, tipoDocumento: "CC", metodoPago: "Tarjeta", reservas: [] },
* { nombre: "Patricia Rodríguez", documento: 78901234, tipoDocumento: "TI", metodoPago: "Efectivo", reservas: [] },
* { nombre: "Esteban Vargas", documento: 43210987, tipoDocumento: "CC", metodoPago: "Tarjeta", reservas: [] },
* { nombre: "Sandra Mendoza", documento: 89012345, tipoDocumento: "TI", metodoPago: "Efectivo", reservas: [] }

]);

Se insertaron 15 diferentes clientes sin reservas

-Para insertar 15 hoteles se uso un script desde la consola de mongo

db.Hotel.insertMany([

...Array.from({ length: 15 }, (\_, i) => ({

id: "hotel" + (i + 1),

nombre: "Hotel " + (i + 1),

numEstrellas: 3 + (i % 3),

habitaciones: [],

servicios: []

  }))

]);

-Para insertar 15 habitaciones se uso el script

db.Habitacion.insertMany([

...Array.from({ length: 15 }, (\_, i) => ({

id: "habitacion" + (i + 1),

disponible: true,

tipoHabitacion: {

id: "tipo" + (i % 3 + 1),

nombre: "Tipo " + (i % 3 + 1),

capacidad: 2 + (i % 3),

precioNoche: 100.00 + (i % 3 \* 50)

},

hotel: {

id: "hotel" + (i % 2 + 1),

nombre: "Hotel " + (i % 2 + 1),

numEstrellas: 3 + (i % 3)

}

}))

]);

-Para cargar datos a SPA se uso el script

db.SPA.insertMany([

...Array.from({ length: 15 }, (\_, i) => ({

id: "spa" + (i + 1),

servicio: {

id: "servicio" + (i + 1),

nombre: "Servicio SPA " + (i + 1),

capacidad: 10 + i,

costo: 50.00 + (i \* 5),

horaApertura: "08:00",

horaCierre: "20:00"

},

serviciosSPA: []

}))

]);

-Para cargar datos a ServicioBasico

db.ServicioBasico.insertMany([

...Array.from({ length: 15 }, (\_, i) => ({

id: "servicioBasico" + (i + 1),

nombre: "Servicio Básico " + (i + 1),

capacidad: 20 + i,

costo: 30.00 + (i \* 2),

horaApertura: "08:00",

horaCierre: "22:00",

consumos: [],

servicioEspecifico: "Especifico " + (i + 1)

  }))

]);

-Para cargar datos a TipoHabitacion se uso el script

db.TipoHabitacion.insertMany([

...Array.from({ length: 15 }, (\_, i) => ({

id: "tipoHabitacion" + (i + 1),

nombre: "Tipo Habitación " + (i + 1),

capacidad: 2 + (i % 3),

precioNoche: 100.00 + (i \* 10),

habitaciones: [],

dotacion: []

}))

]);

-Para insertar datos a Gimnasio

db.Gimnasio.insertMany([ ...Array.from({ length: 15 }, (\_, i) => ({ id: "gimnasio" + (i + 1), servicio: { id: "servicioGimnasio" + (i + 1), nombre: "Gimnasio " + (i + 1), capacidad: 30 + i, costo: 150.00 + (i \* 5), horaApertura: "05:00", horaCierre: "23:00" }, maquinas: [] }))]);

-Para insertar datos a Piscina

db.Piscina.insertMany([

...Array.from({ length: 15 }, (\_, i) => ({

id: "piscina" + (i + 1),

servicio: {

id: "servicioPiscina" + (i + 1),

nombre: "Servicio de Piscina " + (i + 1),

capacidad: 50 + i,

costo: 200.00 + (i \* 10),

horaApertura: "06:00",

horaCierre: "22:00"

},

profundidad: 1.5 + (i \* 0.1)

  }))

]);

-Para insertar datos a Tienda

db.Tienda.insertMany([ ...Array.from({ length: 15 }, (\_, i) => ({ id: "tienda" + (i + 1), servicio: { id: "servicioTienda" + (i + 1), nombre: "Tienda " + (i + 1), capacidad: 20 + i, costo: 100.00 + (i \* 5), horaApertura: "09:00", horaCierre: "21:00" }, tipoTienda: "Tipo " + (i % 3 + 1), productos: [] }))]);

Para el resto de los servicios se usó scripts similares a los anteriores.