Documento de informe

Mateo Saravia, Joseph Ortiz
Contexto de Presentación del documento
Universidad de los Andes, Bogotá, Colombia
{m.saravia, js.ortiz}@uniandes.edu.co
Fecha de presentación: Abril 24 de 2018

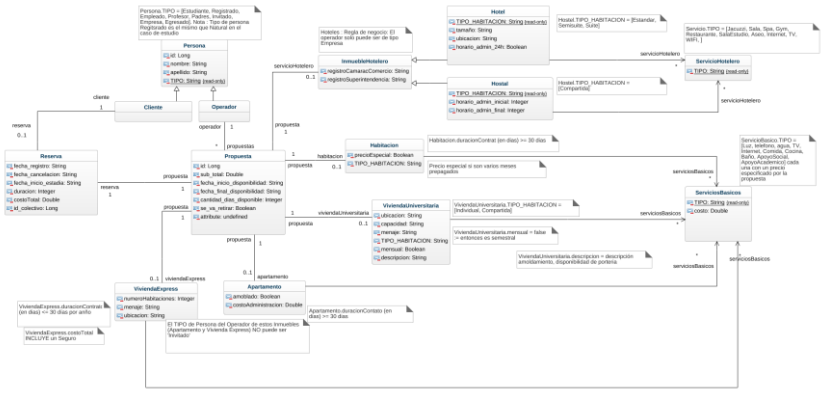
Tabla de contenido

1 Análisis

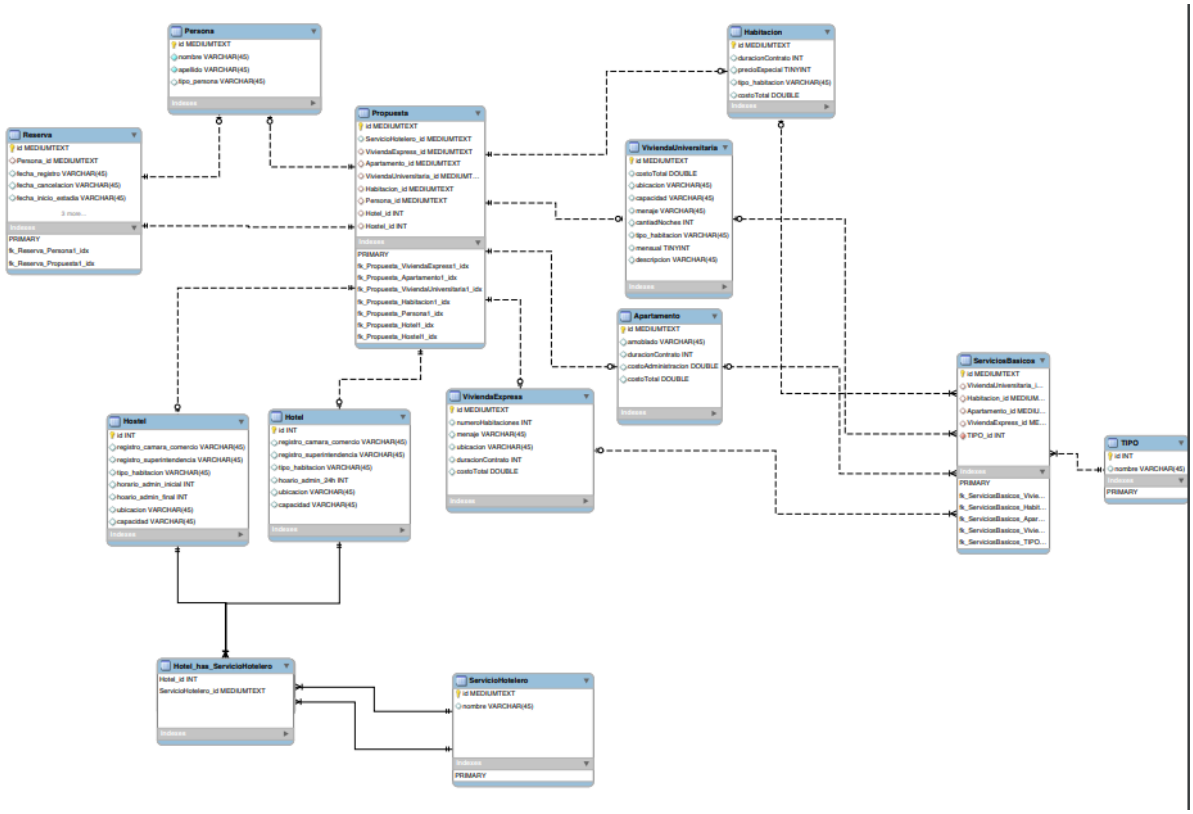
En la segunda iteración se Agregó una nueva clase en los Vos y el service, puesto que facilitaban el desarrollo de la aplicación. Se agregó la clase reserva colectiva que al hacer un json no había necesidad de registrar una colección, sino que se agregaba en la tabla de reserva la cantidad de inmuebles que se desea separar. También se agregó una tabla en la base de datos que muestra el tiempo de des habilitación de las propuestas registradas. Se agregaron los nuevos métodos de consulta. Le cambiamos las columnas a las tablas de la iteración pasada ya que había muchas que no servían y contenían datos que no importaban.

2 Modelos

2.1. Modelo conceptual



2.2. Modelo Relacional



2.3. tablas

Persona

id	nombre	apellido	cedula	tipo
PK	NN	NN		NN

Reserva

id	id_persona	id_propuesta	fecha_registro	fecha_cancelacio
PK	FK(personas.id)	FK(propuestas.id)	NN (YYYY-MM-DD)	(YYYY-MM-DD)

Propuesta

id	tipo_inmueble	id_persona	id_hotel	id_hostel	id
PK	NN	FK(personas.id)	FK(hoteles.id)	FK(hosteles.id)	FK

Hotel

id	registro_camara_comercio	registro_superintendencia	tipo_habitacion	horar
PK	NN	NN	NN	0|1

Hostal

id	registro_camara_comercio	registro_superintendencia	tipo_habitacion	horario_
PK	NN	NN	NN	hh:mm:s

Hotel_has_servicios

id_servicio_basico_hotelero	id_hotel	id_hostel
PK1, FK(servicios_basicos_hoteleros.id)	PK2, FK(Hoteles.id)	PK3, FK(Hosteles.id)

Servicio_hotelero

id	nombre
PK	NN

Vivienda_express

id	numero_habitaciones	menaje	ubicación
PK	NN		NN

Apartamento

Id	amoblado	costo_admin
PK	0|1	NN

Vivienda_universiaria

id	ubicación	capacidad	menaje	tipo_habitacion	mensual
PK	NN	NN		NN	0|1

Habitacion

id	precio_especial	tipo_habitacion
PK	0|1	NN

Servicio_basicos

id	id_tipo	id_habitacion	id_vivienda_universitaria	id_apartame
PK	FK(Tipos.id)	FK(Habitaciones.id)	FK(viviendas_universitarias.id)	FK(apartame

Tipos

id	nombre	costo
PK	NN	NN

Propuesta_deshabilitada

id_propuesta	fecha_inicio_deshabilitada	fecha_fin_deshabilitada
FK(Propuesta.id)	NN	NN

3 Requerimientos

Requerimientos funcionales

3.1 RF7: Registrar reserva colectiva

En la reserva colectiva, recibe un objeto de tipo ReservaColectiva. Primero valida las propuestas que cumplen con lo que es requerido, luego valida que cumplan con los servicios requeridos y el tipo de inmueble, retornando una lista de las propuestas en donde se van a hacer las reservas. Después genera el objeto de tipo reserva para así registrarlo en la base de datos (llamando el método de registrar reserva que valida las reglas de negocio).

3.2 RF8: Cancelar reserva colectiva

En este método, recibe un id como parámetro para así buscar la reserva que se necesita y aplicarle la multa. Después se llama al método de cancelarReserva() que valida las reglas de negocio.

3.3 RF9: Deshabilitar propuesta

En este método, se valida que no se vaya a retirar la propuesta y que esté habilitada, enviando una excepción de lo contrario. Luego se obtiene la fecha del sistema para registrarla en la des habilitación de la propuesta. Después Obtengo todas las reservas que están en esa propuesta, ya que según el requerimiento que deben cambiarse de propuesta, ordenadas según

el id de la persona para agruparlas. Después se toma una propuesta que tenga el mismo tipo de inmueble en la cual va ser asignada a la reserva. Ahora al tener las reservas con la nueva propuesta genera un objeto de tipo ReservaColectiva y ser llama el método de registrarReservaColectiva para que cambie las reservas. Luego se cambia el atributo de "habilitada", se asigna la fecha en su atributo correspondiente, y por último se cambia en la base de datos.

3.4 RF10: rehabilitar propuesta

El sistema valida que este deshabilitada, enviando una excepción de lo contrario. Después llama el método de habilitar propuesta y el cual valida la fecha con respecto al sistema. Al cambiarlo, cambia el atributo en "habilitada" y luego se persiste en la base de datos.

3.5 RFC 8: Encontrar las reservas más populares

En este requerimiento se hace una conexión entre la tabla de reserva y personas, donde se pregunta por la duración de una reserva o el número de reservas que haya hecho de esa propuesta.

3.6 RFC 9: Alojamientos de baja demanda

Se conectan las tablas de propuesta con la de reserva y se conecta con la tabla de propuestas y se verifica si la duración es menor o igual a 30

4 Conclusiones

Funcionaron los dos primeros requerimientos funcionales, pero los dos últimos no. Puesto que no dieron los resultados esperados, ya que enviaba un erro de un nulo, y una columna que no era válida, según el sistema, error en el objeto de entrada en postman. Faltaron los requerimientos de consulta del 1 al 4 y el 7 no funcionó. Hubo conflictos en el desarrollo de la aplicación.