Gestión de Datos

Trabajo Práctico

2° Cuatrimestre 2015

AEROLINEA FRBA

Grupo JANADIAN_DATE

Curso: K3073

Numero de grupo: 15

Nombre: Matias Basualdo

Legajo: 1214408

Version 0.1

Indice

Estrategia

Consideraciones Generales - Diseño de Base de Datos

Para el borrado lógico utilizaremos el campo habilitado en las tablas que lo necesiten

1. ABM ROL

Va a haber 2 roles (admin y cliente)
Por defecto estarán habilitados y el nombre sera único

Para poder agregar y eliminar de a uno las funcionalidades a cada rol hicimos una tabla intermedia (Rol_Funcionalidad) en la cual no puede haber pares repetidos.

Para la restricción de no poder asignar un rol inhabilitado a un usuario creamos una constrain check en el atributo Rol del usuario para chequear en cada insert de usuario, con una función (Rol_Habilitado) que el rol este habilitado.

Al momento de la baja lógica de un rol creamos un Trigger (trgQuitarRolDeshabilitadoDeUsuario) en esa tabla sobre los updates que elimina el rol de cada usuario asignado cuando se inhabilita, soportando la restricción que se debe quitar el rol inhabilitado a todos los que lo posean.

2. Login y seguridad

Creamos un trigger (trgDeshabilitarUserFallido) sobre la tabla Usuario que se ejecuta sobre el update verificando si el atributo Intentos es MAYOR O IGUAL a 3 deshabilita el usuario.

El login lo vamos a controlar desde la aplicación ya que es una funcionalidad especial.
También desde la aplicación vamos a limpiar la cantidad de intentos fallidos cada vez que se loguee correctamente.

3. Registro de Usuario

Generamos un set de usuarios Administradores:

admin (pedido en enunciado)
sucursal1
admin2
sucursal2

todos con rol Administrador General y password w23e que generamos con la función
```
LOWER(CONVERT(NVARCHAR(64),HashBytes('SHA2_256', 'w23e'),2))
```
para satisfacer lo pedido de tenerlo almacenado encriptado de forma irreversible bajo el algoritmo de encriptacion SHA256

También para pruebas generamos un usuario Cliente llamado invitado con la misma clave aunque la aplicación no controlara la clave pero utilizara el user cuando ingrese cualquier usuario para validar las funcionalidades.

4. ABM Ciudades

Como no se implementa solo se tendrá en cuenta que se cargaran los datos con todas las ciudades de tabla maestra verificando que no haya repetidos y estará relacionada con las rutas.

Como no se va a implementar el abm de ciudades inferimos que no va a haber bajas así que en la FK de rutas no tomaremos acciones al eliminar una ciudad

5. ABM Rutas

Ponemos una contraint check para que en toda ruta ciudad origen sea diferente a ciudad destino
El tipo de servicio es una FK a la tabla Tipo_Servicio

Al dar de baja una ruta se inhabilitara y se crearan cancelaciones para todas las compras (pasajes y emcomiendas) referenciadas a esa ruta. Esto lo resolveremos con los trigger trgInhabilitarPasajeRuta y trgInhabilitarPaqueteRuta que ejecutara el Store Procedure de cancelacion por cada compra paquete y encomienda que este asociada con compras futuras a esa ruta.

Tipo_Servicio atributo de Aeronave y de Ruta ya que debemos validar que una aeronave que se le asigna una ruta debe ser del tipo de servicio que ella ofrece.

6. ABM Aeronave

La fecha de alta la tomaremos del momento de cada insert
Estará por defecto habilitada
Y los campos bit de baja estarán por defecto en 0, indicando que no tiene ningún tipo de baja
Fabricante y Tipo_Servicio serán FK a las respectivas tablas sin restricciones ya que son datos fijos
La matricula debe ser única.
La cantidad de asientos ventanilla o pasillo la contaremos en el momento de importar los datos de la tabla maestra

Cada aeronave tendrá un Tipo_Servicio relacionado y una lista de butacas asociada fija que serán cargadas en la importación de la tabla maestra.

Bajas: si completo la vida útil , se inhabilita y se marca la baja vida útil (booleano) para que no se puedes volver a poner en servicio.

Si esta fuera de servicio, se inhabilita pero se puede revertir, por lo tanto consideraremos una función para volver a poner en servicio.

Si pasa alguna de las bajas se debe cancelar los pasajes/encomiendas o suplantar la aeronave (sin generar cancelaciones) por otra (=tipo y fabricante =tipo_servicio y con capacidad de cumplir los

recorridos programados por la anterior)

En el caso de que sea solo por fuera de servicio la baja es para los viajes del tiempo en que estará fuera de servicio, no para todos los futuros viajes.

Si no hay aeronave para reemplazar también se puede dar de alta una nueva, esto lo decidirá la empresa, así que sera una acción de la aplicación.

Para las bajas por fuera de servicio moderaremos la entidad Fuera_Servicio que tendrá las fechas para calcular el intervalo sin servicio. Y en la tabla aeronave se marcara con un booleano si esta o no fuera de servicio

El piso de cada butaca depende del avión (sacamos los datos de la maestra)
La cantidad de butacas en pasillo y en ventanilla depende del avión (sacamos la cuenta de la información de Maestra).

Consideramos basado en la respuesta del grupo:

La cantidad de butacas en una aeronave depende del máximo nro de pasajes que se registraron en una ruta de avión (vuelo especifico) (sacamos la cuenta de la información de Maestra)

Butaca: Las butacas se deben instanciar por aeronave pero para cada viaje que tenga programado la aeronave se debe tener las butacas disponibles para comprar, razón por la cual una butaca se relaciona con una aeronave pero a la vez con muchos viajes. Debiendo validar al momento de comprar que no este ocupada la butaca para ese viaje, por eso creamos una tabla intermedia Butaca_Viaje en la cual solo debe haber un registro por butaca para cada viaje.

Tendremos dos Entidades que una tendrá Relación a butaca y la otra KG seleccionados, cuando necesitemos saber la cantidad de KG disponibles para una aeronave en un viaje haremos el calculo y para saber las butacas disponibles verificaremos las relaciones que faltan en Butaca_Viaje para el viaje en cuestión

Tomado del mail del grupo:
el atributo butaca_piso, hace referencia a la ubicación tanto del pasajero como de la encomienda.

Teniendo en cuenta la tabla Maestra, los casos en que corresponde a un encomienda va a tener asignado el valor 0 y si es un pasajero el valor 1.

Por lo tanto solo creamos butacas que estén en maestra con valor de piso 1, ya que las que son de paquetes tienen valor 0 para Nro,Piso y Tipo

7. Generar Viaje

Constraint check sobe las fechas para que sean futuras (en el caso de las que ya estaban en tabla maestra se hará un TRIGGER con estas restricciones después de insertar los datos)

FechaLlegada puede ser null porque la cargaran en algún momento luego que el avión llegue a destino.

Aceptaran null las FK a Ruta y aeronave ya que puede haber viajes recién dados de alta sin servicios asignados

Para chequear que el tipo de servicio de la ruta sea el mismo que el de la aeronave hacemos un check sobre los valores que ejecute una función sobre cada una de las tablas mencionadas.

Para que el viaje tenga una aeronave disponible crearemos una constraint check que verifique que este habilitada la aeronave y que no tenga otro viaje en ese día de salida.

REVISAMOS LAS RUTAS Y VIAJES QUE ESTÁN MUY INCONSISTENTES, HAY DIFERENTES RUTAS PARA EL MISMO AVIÓN AL MISMO TIEMPO, Y VIAJES CON VARIAS FECHAS DE LLEGADA, esto implica que hay viajes para aeronaves no disponibles, nuestra estrategia es: CREAREMOS NUEVAS AERONAVES EN LA IMPORTACIÓN DE MAESTRA)

8. Registro de llegada Destino

Para resolver este requerimiento desarrollaremos una vista Itinerario_Aeronave que contara con los campos para la búsqueda de itinerario y así directamente poner la fecha de llegada sobre la vista.

También en este momento se deben computar las millas , 1 milla cada $10 , eso lo haremos con un trigger en el momento de update de FechaLlegada que computara las millas de todos los clientes que volaron en ese viaje (tienen pasajes o encomiendas) registrando la fecha y el motivo de computo de millas [Codigo Viaje] [Detalle de Ruta] para cuando se liste.

VERIFICAR QUE NO SEA UN PASAJE o paquete CANCELADO

9. Compra

Modelaremos un cliente (que puede ser o no pasajero)

El numero de compra o PNR sera el mismo para todos los pasajeros de una misma compra, diferenciándose en el Código Pasaje/encomienda. Siendo PK de la tabla Compra

A los pasajes y paquetes (encomiendas) cargados en Tabla Maestra los tratamos como una unidad de compra como informa la descripción general del TP; diferenciando los casos de pasajes que tendrán una relación a butaca y si es un paquete tendrá el campo KG con su correspondiente valor. Para simplificar la importacion, basandonos en esta premisa guardaremos el codigo de pasaje/paquete en esta tabla a modo informativo para luego importar, ya que no se repiten los codigos entre pasaje y paquete la relacion sera simple

Para la búsqueda de viajes disponibles realizaremos una vista Viaje_Disponible donde contamos con las columnas que necesitamos para buscar (fechaSalida, origen y destino)

Agregaremos una tabla Datos_Tarjeta donde quede almacenada la tarjeta de crédito si la forma de pago fue TC y esta asociada a una compra y un cliente
Valores posibles `('TC','EFECTIVO'), default efectivo`

Una validación a tener en cuenta es que un pasajero no puede viajar a más de un destino a la vez. Lo haremos desde la aplicación que verificara con un store procedure que un cliente no tenga pasajes

en vuelos de la misma fecha que quiere comprar

En esta entidad modelamos la relación a un usuario del sistema que puede ser null si el Usuario compro por Kiosco y tendrá un usuario User si compro por Autoservicio. POR DEFECTO NULL

10. Cancelación

Esta funcionalidad la resolveremos con un Store Procedure Cancelar_Pasaje
de cancelación que se encargara de resolver la cuestión de devolver el dinero de las compras (no dice que haya que implementar una funcionalidad sobre esto) ,cancelar los pasajes y liberar las butacas / kg de la aeronave, No computar los posibles puntos (al informar una llegada a destino validar que no este cancelado para computar el pasaje) y devolver el dinero en la misma forma de pago en la cual se realizo la compra.

Agregaremos un atributo booleano a la tabla Pasaje para informar si esta cancelado, por defecto en false.

11. Consulta Millas

Las millas se acumulan cuando se registra una llegada a destino de una aeronave y debemos hacer un listado de Acumulación con sus canjes fechas y Motivos

12. Canje de Millas

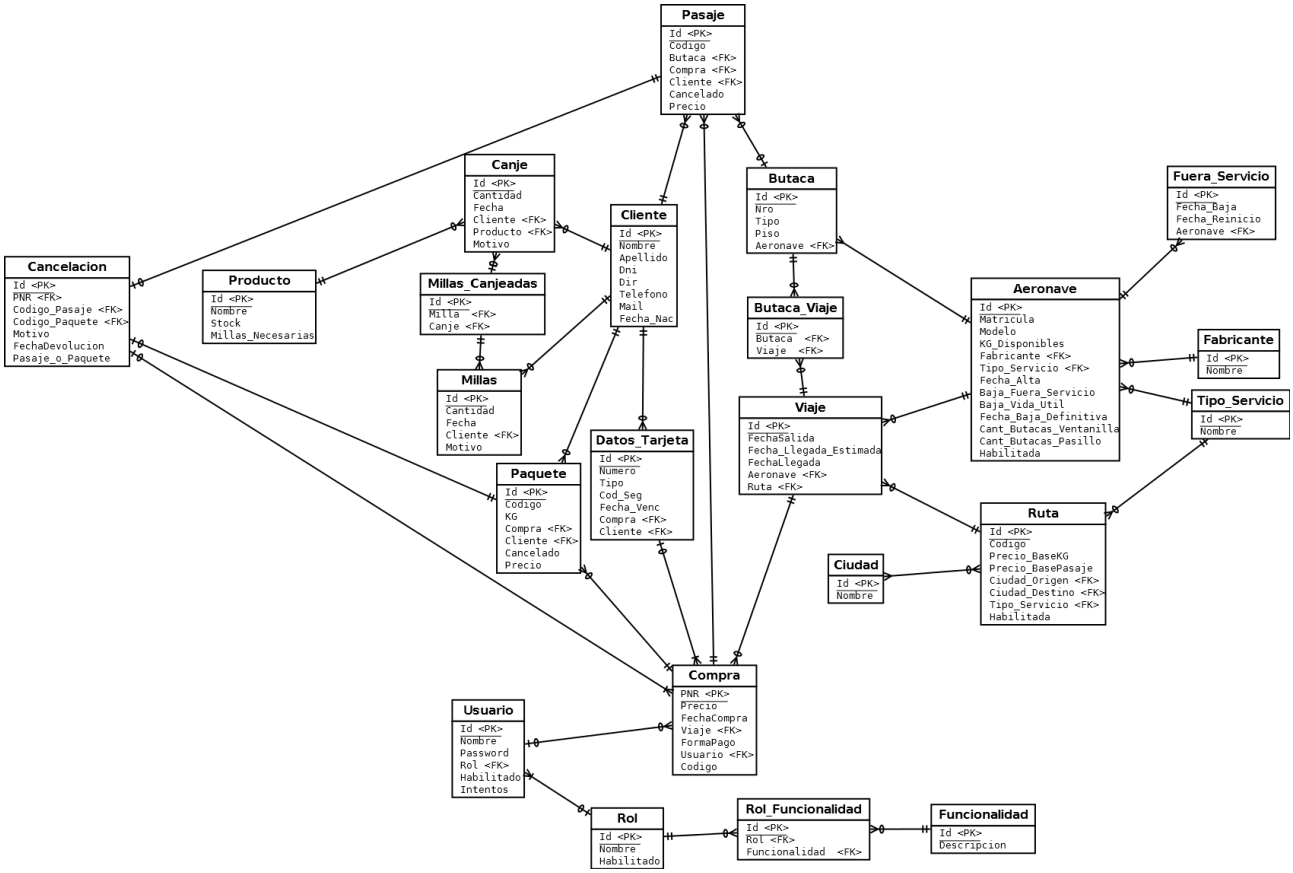
Modelamos Productos con nombre y Stock y millas necesarias para canjear cada producto. Cuando se realice un canje se debe validar que cumpla las millas necesarias, descontarlas de las disponibles, restar 1 al stock y grabar el canje en la tabla Canje con fecha y motivo y cantidad canjeada.
El motivo dira [canje de X millas por Y productos], validar que la milla consultada no este vencida (tienen validez de un año)

Cargaremos un stock variado de productos.

Relacionaremos canje con millas para que quede grabado el canje a que millas esta relacionado

13. Listado estadistico

DER


Pasaje
Id <PK>
Codigo
Butaca <FK>
Compra <FK>
Cliente <FK>
Cancelado
Precio
Canje
Id <PK>
Cantidad
Fecha
Cliente <FK>
Producto <FK>
Motivo
Cliente
Id <PK>
Nombre
Apellido
Dni
Dir
Telefono
Mail
Fecha_Nac
Butaca
Id <PK>
Nro
Tipo
Piso
Aeronave <FK>
Fuera_Servicio
Id <PK>
Fecha_Baja
Fecha_Reinicio
Aeronave <FK>
Cancelacion
Id <PK>
PNR <FK>
Codigo_Pasaje <FK>
Codigo_Paquete <FK>
Motivo
FechaDevolucion
Pasaje_o_Paquete
Producto
Id <PK>
Nombre
Stock
Millas_Necesarias
Millas_Canjeadas
Id <PK>
Milla <FK>
Canje <FK>
Butaca_Viaje
Id <PK>
Butaca <FK>
Viaje <FK>
Aeronave
Id <PK>
Matricula
Modelo
KG_Disponibles
Fabricante <FK>
Tipo_Servicio <FK>
Fecha_Alta
Baja_Fuera_Servicio
Baja_Vida_Util
Fecha_Baja_Definitiva
Cant_Butacas_Ventanilla
Cant_Butacas_Pasillo
Habilitada
Fabricante
Id <PK>
Nombre
Tipo_Servicio
Id <PK>
Nombre
Millas
Id <PK>
Cantidad
Fecha
Cliente <FK>
Motivo
Datos_Tarjeta
Id <PK>
Numero
Tipo
Cod_Seg
Fecha_Venc
Compra <FK>
Cliente <FK>
Viaje
Id <PK>
FechaSalida
Fecha_Llegada_Estimada
FechaLlegada
Aeronave <FK>
Ruta <FK>
Paquete
Id <PK>
Codigo
KG
Compra <FK>
Cliente <FK>
Cancelado
Precio
Ruta
Id <PK>
Codigo
Precio_BaseKG
Precio_BasePasaje
Ciudad_Origen <FK>
Ciudad_Destino <FK>
Tipo_Servicio <FK>
Habilitada
Ciudad
Id <PK>
Nombre
Compra
PNR <PK>
Precio
FechaCompra
Viaje <FK>
FormaPago
Usuario <FK>
Codigo
Usuario
Id <PK>
Nombre
Password
Rol <FK>
Habilitado
Intentos
Rol
Id <PK>
Nombre
Habilitado
Rol_Funcionalidad
Id <PK>
Rol <FK>
Funcionalidad <FK>
Funcionalidad
Id <PK>
Descripcion

Estructuras de Datos

Entidades