TRABAJO PRACTICO

1º CUATRIMESTRE

2019

UTN – FACULTAD REGIONAL BUENOS AIRES

GESTIÓN DE DATOS

- **Título**: Gestión de Datos
- **Año**: 3°
- **Código**: 082030
- **Curso**: K3012
- **Grupo**: MACACO_NOT_NULL

INTEGRANTES	LEGAJO
COBOS, Bruno	*159676-7*
JUGO, German	*158917-9*
NIKCEVICH, Carlos Alexis	*146698-7*
VERDILE, Guillermo	*158962-3*

UTN - FRBA

Índice

Introducción

A través de la Estrategia proponemos explicar las decisiones que fueron pertinentes tomar al momento de realizar la etapa del diseño de la aplicación Desktop con el framework de C# .Net versión 4.5, y su posterior implementación, como así también el modelado de los datos y sus relaciones, implementado en MS SQL Server 2012. Adjuntamos además el Diagrama Entidad Relación.

Como puntos a destacar, mencionamos que el tiempo de migración promedio aproximado es de 1:03 minutos.

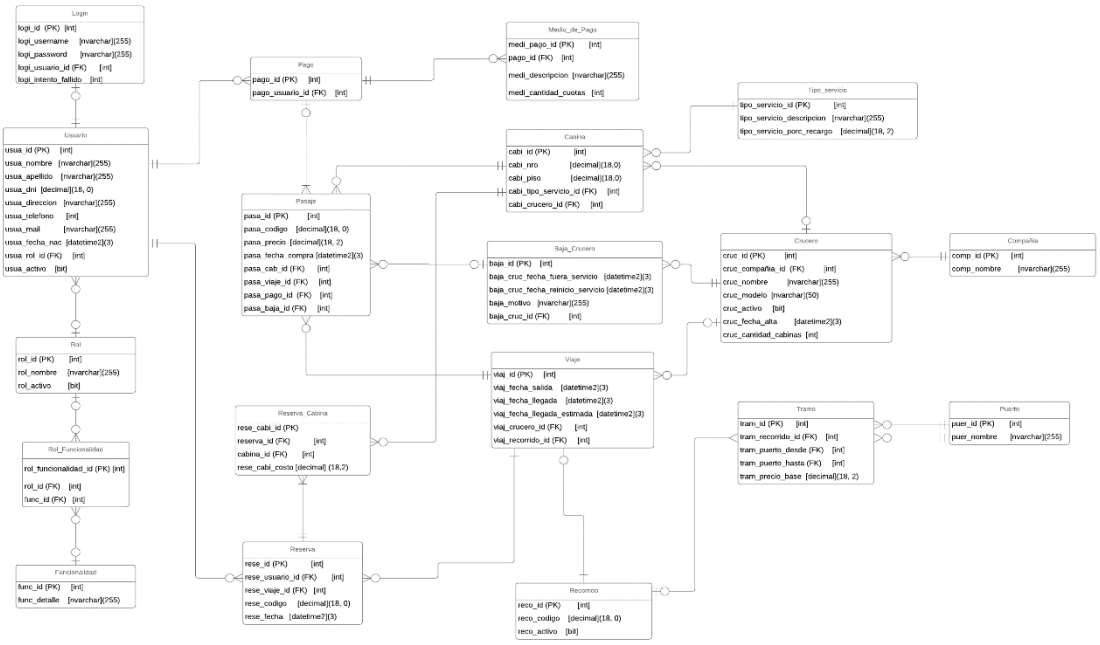
Antes de comenzar con el desarrollo se analizaron los requerimientos funcionales y no funcionales del sistema, para luego normalizar los datos provenientes de la tabla maestra provisto por la cátedra.

En la parte de pruebas de la aplicación, se asignan los siguientes nombres de usuarios con roles de administradores:

- *jPerez*
- *jGonzalez*
- *aMontana*
- *mMozart*
- *rtesoro*
- *admin*

Todos estos usuarios tienen como contraseña "*w23e*" para ingresar al sistema como ADMINISTRADOR.

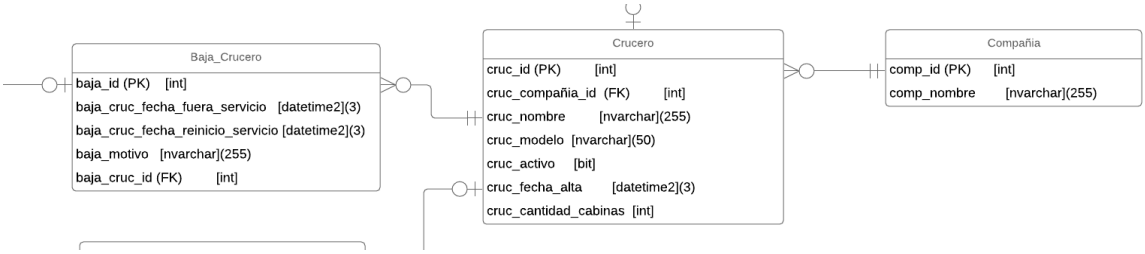
DER



Aclaración importante: para más detalles, *ver archivo DER.jpg adjunto en el archivo zip.*

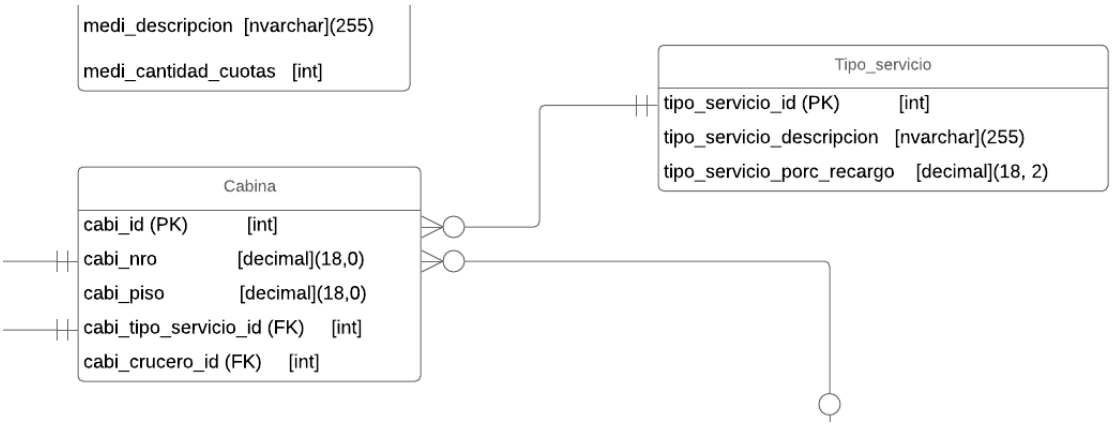
Desarrollo

Crucero, Baja Crucero y Compañía



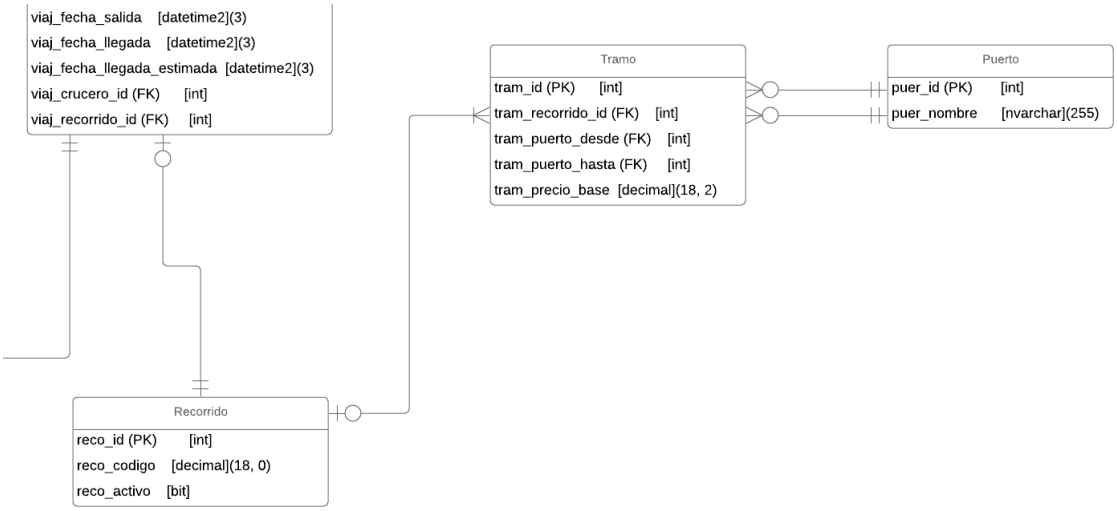
Para diseñar la tabla cruceros tomamos en cuenta que su PK es *cruc_id,* como FK está *cruc_compañía*. También decidimos guardar el nombre, el modelo, la cantidad de cabinas y el estado del crucero lo representamos con los atributos de crucero activo y crucero fecha de alta. En este punto se decidió modelar una entidad aparte para la *Baja de Crucero*, que tiene como PK *baja_id* y como FK a *baja_cruc_id,* además de guardar la fecha de fuera de servicio y fecha del reinicio de servicio como así también el motivo de la baja.

Cabina y Tipo de Servicio



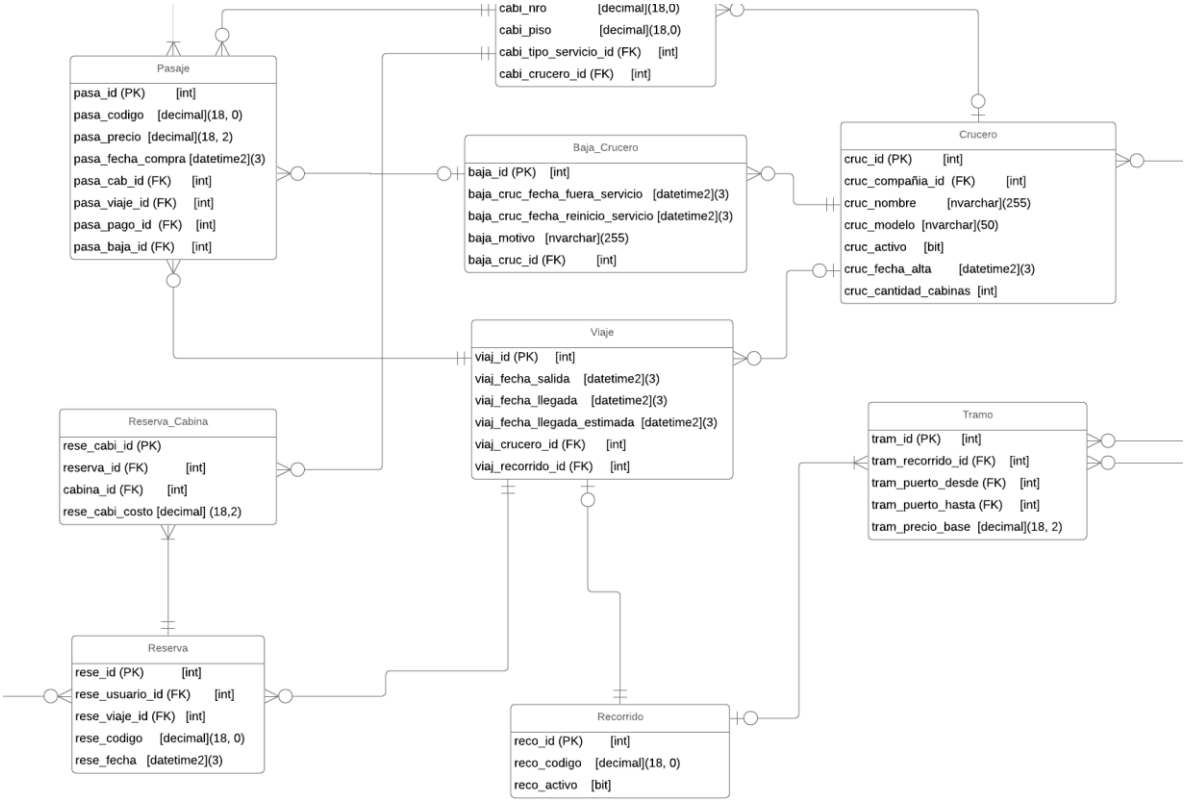
La tabla cabina tiene como PK a *cabi_id*, como FK a *cabi_crucero_id* y *cabi_tipo_servicio,* permite guardar además el número y piso la cabina. Acá se decidió que la tabla Tipo_servicio almacene los datos referidos a la descripción y porcentaje de recargo asociados al tipo de cabina.

Puerto, Tramo y Recorrido



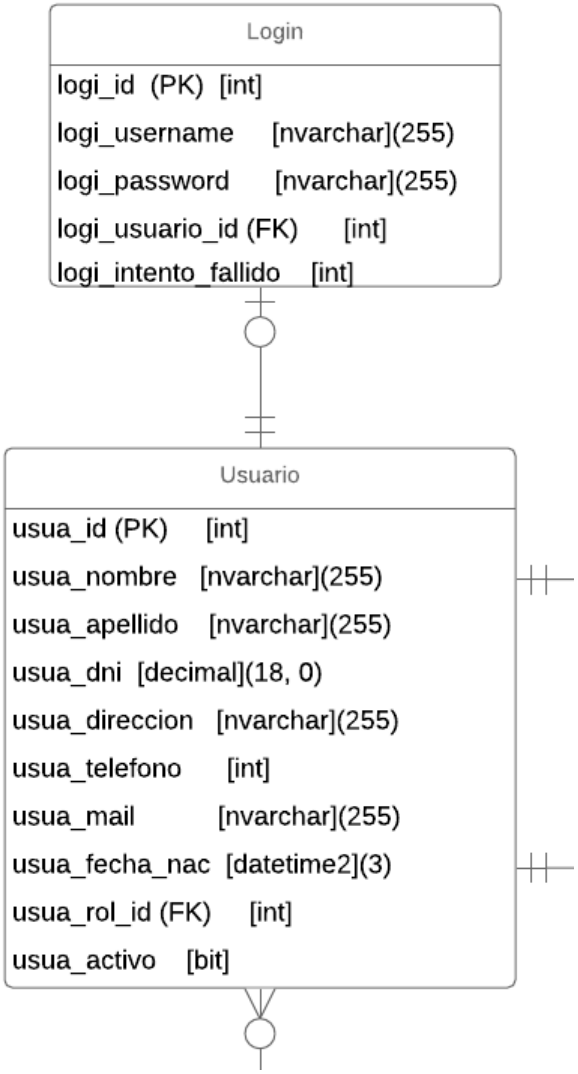
En la tabla Recorrido la PK es reco_id, guarda también el código del recorrido y si está o no activo. Del Puerto, su PK es puer_id, guardando el nombre de este. Creamos tramos para poder persistir las distintas partes que conforman un recorrido. Puerto tiene dos relaciones con tramo debido a que todos los tramos tienen puerto destino y puerto de llegada

Viaje



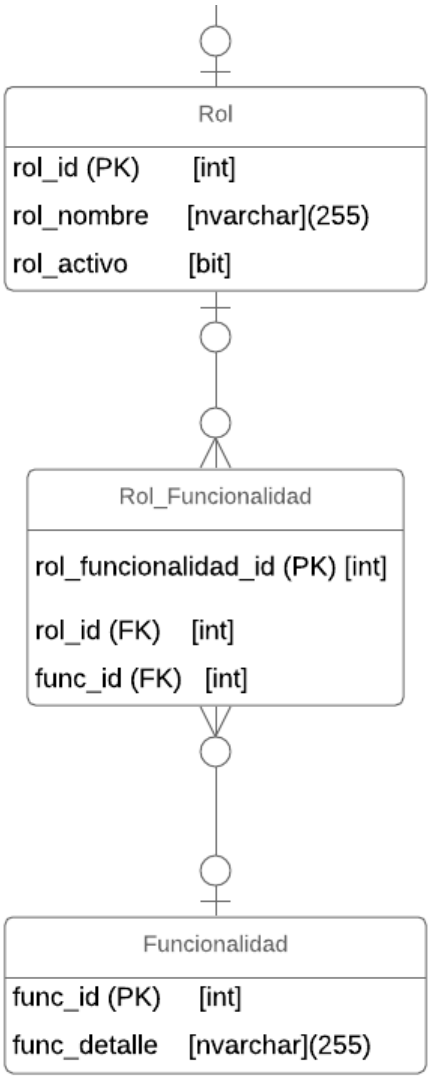
Para los viajes mencionamos que su PK es *viaj_id*, sus FK son *viaj_cruc_id* y *viaj_recorrido*, permitiendo persistir la fecha de salida, la fecha de llegada y la fecha de llegada estimada. Esta tabla se relaciona con las tablas Crucero, Pasaje, Reserva y Recorrido.

Usuario y Login



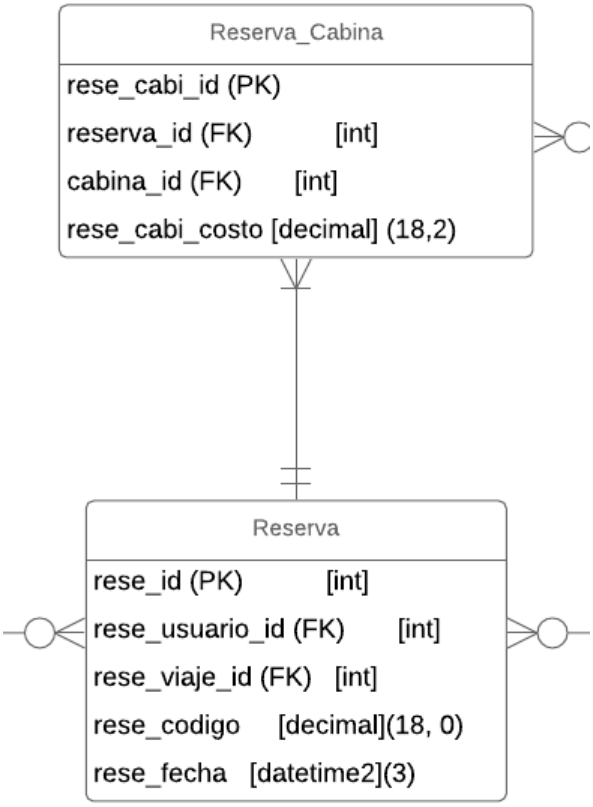
La tabla Usuario tiene PK *usua_id,* como FK a *usua_rol_id*. Guarda nombre, apellido, DNI, dirección, teléfono, mail, fecha de nacimiento y si está activo el usuario. Mientras que la tabla de Logintiene PK *logi_id*y cómo FK a *logi_usuario_id.* Aquí se decidió utilizar el mecanismo de encriptación de hashing para los datos sensibles como lo son las contraseñas de los usuarios. Otro punto que consideramos para reforzar la seguridad es implementar un contador de intentos fallidos que bloquea un usuario cuando éste ingresa 3 veces mal su contraseña, cumpliendo así con uno de los requerimientos establecidos en el trabajo práctico.

Rol, Funcionalidad y Rol x Funcionalidad



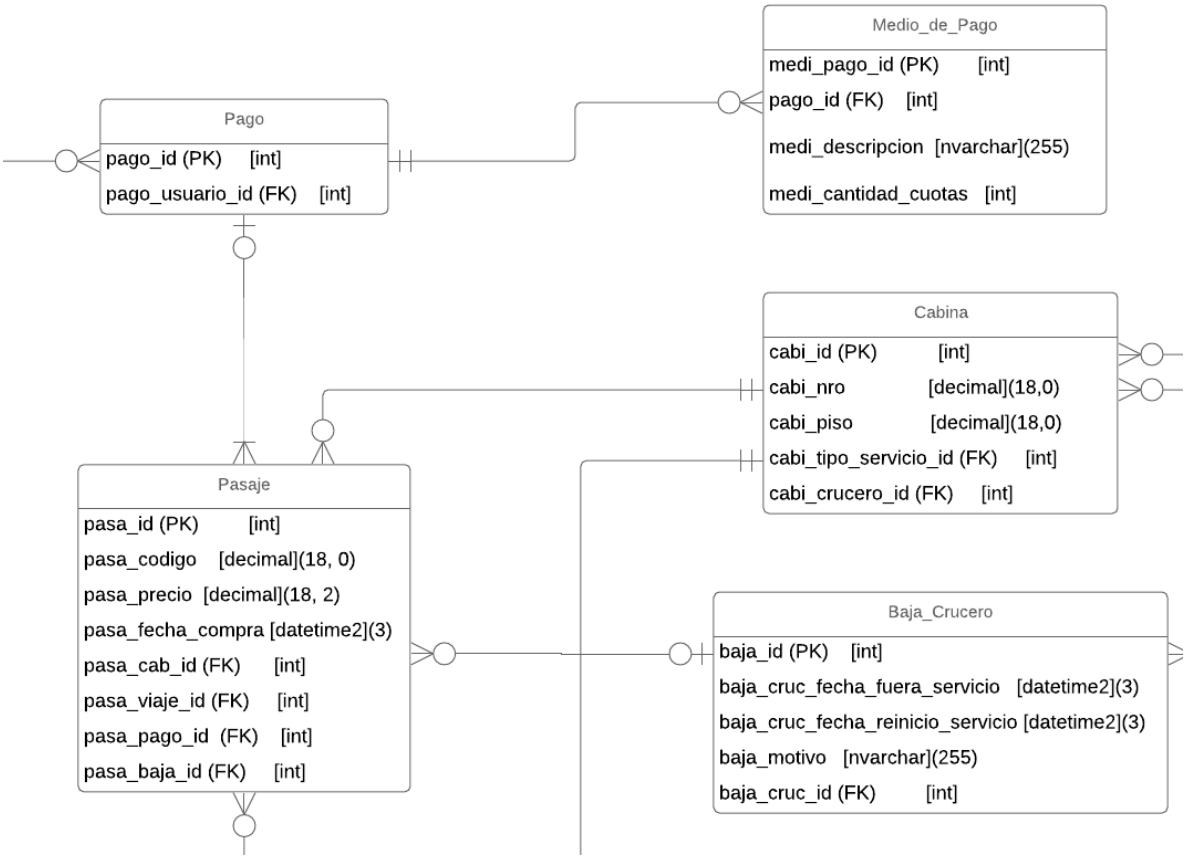
Decidimos modelar a los usuarios con distintas funcionalidades de acuerdo con el rol que ocupen dentro del sistema, pudiendo ser *administradores* o *clientes*. La tabla Rol tiene PK a*rol_id*, guarda el nombre y si está activo o no el rol. La tabla Funcionalidad tiene PK *func_id* y guarda el detalle de la misma. Para romper la relación de muchos a muchos se crea la entidad Rol x Funcionalidad, con su PK *rol_funcionalidad_id* y hereda como FK a las PKs de las 2 entidades.

Reserva y Reserva Cabina



Para verificar cuándo vence una reserva decidimos que en la tabla se guarde la fecha en la que fue efectuada la misma. Se asigna también un código para su posterior pago ya que es un compromiso de compra por parte del usuario respaldado por las reglas vigentes del negocio. La tabla reserva cabina guarda las distintas cabinas que el usuario eligió al momento de realizar la reserva. La entidad Reserva también se relaciona con Viaje.

Pasaje, Pago y Medio de Pago



Cuando una reserva necesita ser pagada se genera un pasaje con código de pasaje, el precio, la fecha de compra, la identificación de la cabina, la identificación del viaje, la identificación del pago y en el caso de que se de baja un crucero también se guarda la baja identificación. El pago lo asocia a un usuario qué puede elegir diferentes medios de pago qué debe especificar junto con la cantidad de cuotas.

También se puede realizar una compra directa de pasajes sin tener que pasar por la etapa de reserva.

Script Inicial, Stored Procedures, Triggers y Funciones

Procedures

PROCEDURE[MACACO_NOT_NULL].AltaRol:Este procedure da de alta un rol con una funcionalidad.

PROCEDURE[MACACO_NOT_NULL].BuscarRol: Este procedure trae una lista de roles los cuales cumplan con los filtros elegidos.

PROCEDURE[MACACO_NOT_NULL].BajaRol: Este procedure establece en inactivo un rol.

PROCEDURE[MACACO_NOT_NULL].ModificarNombreRol: Se llama este procedure primero al querer modificar un rol

PROCEDURE[MACACO_NOT_NULL].AgregarFuncionalidadRol: Si en la tabla de funcionalidad pantalla de modificacion de Rol hay N filas, llamar a este procedure N veces

PROCEDURE[MACACO_NOT_NULL].HabilitarRol: Este procedure establece en activo un rol inactivo.

PROCEDURE[MACACO_NOT_NULL].LogearUsuario: Este procedure permite el ingreso de un usuario al sistema verificando su contraseña.

PROCEDURE[MACACO_NOT_NULL].GetCruceros: Obtiene una lista de cruceros de acuerdo a los que cumplan los filtros seteados.

PROCEDURE[MACACO_NOT_NULL].CreateCrucero: Crea un nuevo crucero.

PROCEDURE[MACACO_NOT_NULL].UpdateCrucero: Modifica los valores de un crucero.

PROCEDURE[MACACO_NOT_NULL].GetCabinasXPisoYServicio: Obtiene las cabinas con su tipo de servicio y piso de acuerdo al id del crucero ingresado.

PROCEDURE[MACACO_NOT_NULL].GetTramos: Obtiene una lista de tramos de acuerdo al codigo del recorrido ingresado.

PROCEDURE[MACACO_NOT_NULL].getRecorridos: Obtiene una lista de recorridos de acuerdo al codigo del recorrido ingresado y las ciudad de origen y destino.

PROCEDURE[MACACO_NOT_NULL].InsertRecorrido:Inserta todos los recorridos que tengan el codigo ingresado en una tabla de tramos.

PROCEDURE[MACACO_NOT_NULL].BajaRecorrido: Da de baja un recorrido salvo que el recorrido se encuentre en un viaje.

PROCEDURE[MACACO_NOT_NULL].GetPuertoByName: Obtiene todos los puertos que tengan el nombre ingresado.

PROCEDURE[MACACO_NOT_NULL].ModificarRecorrido: Modifica los valores de un recorrido.

PROCEDURE[MACACO_NOT_NULL].GetFuncionalidades: Obtiene una lista de funcionalidades de acuerdo a los filtros seleccionados.

PROCEDURE[MACACO_NOT_NULL].UpdateRol: Modifica los valores de un rol.

PROCEDURE[MACACO_NOT_NULL].GenerarViaje: Genera un nuevo viaje con la fecha, el crucero y el recorrido ingresado.

PROCEDURE[MACACO_NOT_NULL].OrdenarTramosRecorridos: Procedure que devuelve los tramos de los recorridos ordenados lógicamente.

PROCEDURE[MACACO_NOT_NULL].AgregarBajaCrucero: Se realiza la baja de un crucero de manera temporal.

PROCEDURE[MACACO_NOT_NULL].AgregarBajaCruceroDefinitivo: Realiza la baja de un crucero de manera definitva sin fecha de recuperacion y sin motivo.

PROCEDURE[MACACO_NOT_NULL].CancelarPasajes: Se ejecutar luego de agregar una baja a un crucero (siempre y cuando la accion posterior elegida por el admin era cancelar los pasajes vendidos del viaje) .

PROCEDURE[MACACO_NOT_NULL].PosponerViajes: Para posponer, una cierta cantidad de dias, los viajes del crucero que se acaba de dar de baja.

PROCEDURE[MACACO_NOT_NULL].ReemplazarCrucero: Reemplaza un crucero con otro el cual no se le superpongan los viajes que posee con los nuevos que tendria que realizar.

PROCEDUREMACACO_NOT_NULL.IdCruceroRemplazante: Obtiene el id de un crucero el cual puede reemplazar al crucero ingresado, si devuelve -1 no se encontro ninguno.

PROCEDURE[MACACO_NOT_NULL].CrearViaje: Creacion de un nuevo viaje

PROCEDURE[MACACO_NOT_NULL].ComprobarVigenciaReservasDelSistema: Comprobacion del vencimientos de todas las reservas.

PROCEDURE[MACACO_NOT_NULL].ComprobarVigenciaReserva:Comprobacion de vencimiento de reserva.

PROCEDUREMACACO_NOT_NULL.CabinasDisponiblesViaje: Obtiene la lista de las cabinas disponible para un viaje.

PROCEDURE[MACACO_NOT_NULL].[GenerarReserva]: Generacion reserva

PROCEDURE[MACACO_NOT_NULL].AgregarCabina_Reserva: Agregar 1 cabina a 1 reserva

PROCEDURE[MACACO_NOT_NULL].AgregarPasajeA_Cliente: Procedure que agrega 1 pasaje, se ejecuta en el caso que se compre directamente un pasaje, sin pasar por la reserva

PROCEDURE[MACACO_NOT_NULL].AgregarPagoReserva_Y_PasajesAlCliente: Agrega los pasajes y el pago de la reserva al cliente de acuerdo al codigo de reserva.

PROCEDURE[MACACO_NOT_NULL].AgregarMedioDePago_Al_NuevoPago: Agrega los medios de pago a un nuevo pago.

PROCEDURE[MACACO_NOT_NULL].EliminarReserva: Elimina un reserva.

PROCEDURE[MACACO_NOT_NULL].[CrucerosConMasReparaciones]: Obtiene una lista de cruceros con mas reparaciones.

PROCEDURE[MACACO_NOT_NULL].[RecorridosConMasCabinasLibresEnSusViajes]:Obtiene una lista con los recorridos que posean mas cabinas libres en sus viajes.

PROCEDURE[MACACO_NOT_NULL].GetViajes: Devuelve todos los viajes que 'pasen' por el puerto origen o el destino.

PROCEDURE[MACACO_NOT_NULL].GetTipoServicioByDescription: Obtiene una lista con los tipos de servicio que tengan la descripcion ingresada.

procedure[MACACO_NOT_NULL].ObtenerCabinasDelCrucero: Obtiene una lista de cabinas del crucero ingresado.

PROCEDURE[MACACO_NOT_NULL].CreateOrUpdateCliente: Crea o modifica los valores de un cliente.

PROCEDURE[MACACO_NOT_NULL].VerificarViajeYaRerservadOComprado: Verifica si un viaje ya fue reservado o comprado.

Funciones

FUNCTION[MACACO_NOT_NULL].GetPuertoId: Obtiene el id del puerto.

FUNCTION[MACACO_NOT_NULL].GetRecorridoIdByRecoCodigo: Obtiene los recorridos de acuerdo al código ingresado.

FUNCTION[MACACO_NOT_NULL].PrecioRecorrido: Obtiene el precio del recorrido.

FUNCTION[MACACO_NOT_NULL].ComprobarExistenciaReserva: Al ingresar el codigo de una reserva se debe verificar que exista alguna con ese numero.
FUNCTIONMACACO_NOT_NULL.ciudad_origen: Obtiene la ciudad de origen de acuerdo al código de recorrido ingresado.

FUNCTIONMACACO_NOT_NULL.ciudad_destino: Obtiene la ciudad de destiono de acuerdo al código de recorrido ingresado.

FUNCTION[MACACO_NOT_NULL].PrecioRecorrido: Obtiene el precio del recorrido.

FUNCTION[MACACO_NOT_NULL].DetallesReserva: Retorna toda la informacion asociada a una reserva.

FUNCTION[MACACO_NOT_NULL].ObtenerCabinas: Obtiene la lista de cabinas de acuerdo a los parámetros ingresados.

Triggers

TRIGGER[MACACO_NOT_NULL].TRIGGER_BLOQUEAR_USUARIO_POR_LOGIN_FALLIDO: Se ejecuta post login fallido.

TRIGGER[MACACO_NOT_NULL].DeleteReservasCabinas: Es ejecutado cuando se borra delete una reserva.

Tipos de Datos

Cabina_Piso: (atributos, nro_cabina,tipo_servicio,descripcion) Se utiliza al momento del alta del crucero para facilitar la inserción de las cabinas del mismo

`Funcionalidades_Rol_List: (atributos: rol,funcionalidadID,func_detalle): Se utiliza para facilitar la inserción de un nuevo rol con sus funcionalidades.`

Decisiones de Diseño en la aplicación Desktop

Paralelamente se diseñaron las vistas del programa teniendo en cuenta los lineamientos pautados en el *trabajo práctico, para el desarrollo de los ABM.*

Pantalla principal: se visualizan los botones correspondientes a las funcioanlidades del tipo de usuario que ha sido logeado.

En el caso de que un Crucero fuera dado de BAJA:

El sistema procede a comportarse según un parámetro "Estrategia_Baja_Crucero" del Archivo de Configuración "App.config", el cual posee, además, un String de conexión a la base de datos y la fecha que tomará el sistema para funcionar.

Al momento de dar de ALTA un Crucero, hemos decidido para simplificar la entrada de datos al usuario (consideramos poco práctico el ingresar cada cabina individualmente con su tipo de servicio) que se ingrese el piso y la cantidad de cabinas con determinado servicio. Por ejemplo, 20 cabinas en el piso 1 de tipo servicio "Ejecutivo", y así sucesivamente hasta completar el alta. El procedure encargado de este funcionamiento es "CreateCrucero", el cual se encarga de insertar individualmente las cabinas.

Con respecto a las pantallas para elegir el medio de pago: agregamos distintas tarjetas con la cantidad de cuotas habilitadas, obviando las validaciones de número de tarjeta, debido a que consideramos que lo correcto sería comunicarse con una API externa pero esto está más allá del alcance de la materia.

Con respecto al listado estadístico: si no hay datos para el año ingresado, hemos decidido mostrar un "Pop Up" indicando que no hay datos disponibles. La opción descartada era mostrar 5 cruceros con las estadísticas en cero.

En el alta de recorridos: se insertan en orden los tramos para luego obtenerlos en el orden en que se desarrolla el viaje. Esto agiliza las consultas posteriores, ya que nos ahorra ordenar los tramos en otro momento del programa.

En la baja de Rol, se busca por un Rol Exacto. Este rol traído de la base de datos se muestra con un checkbox, el cual indica si está activo o no. Si se desmarca el checkbox al guardar, se da de baja el rol. Esta pantalla se hizo exclusivamente para dar de baja un rol, por lo que para la alta o modificación, se debe ingresar a la pantalla correspondiente.