

***Universidad Tecnológica Nacional - Facultad Regional Buenos Aires***

**Gestión de datos**

**2017**

**Trabajo Práctico Grupal: El monstruo del lago Maser**

**Curso: K**

**Profesor: Juan Zaffaroni - Enrique Reinosa**

|  |  |
| --- | --- |
| **Alumno** | **Legajo** |
| Fernando Matin Besteiro | 1558973 |
| Guillermo Perrone |  |
| Agustin Javier Lago | 1388782 |
| Alavaro De La Cruz | 1593730 |

Índice

**Decisiones generales aplicadas a las tablas del nuevo modelo............................................... 3**

**Descripción de las tablas del nuevo modelo............................................................................. 3**

**Descripciones de los Stored Procedures, Funciones y Triggers............................................ 6**

**Detalle de cada funcionalidad implementada.......................................................................... x**

Migración:....................................................................................................................... x

Detalles generales de los ABM.......................................................................................x

ABM Roles.....................................................................................................................x

ABM Usuarios………………………………………………………………………....x

**Decisiones generales aplicadas a las tablas del nuevo modelo**

Para la realización de este trabajo optamos por que todas las tablas de nuestro sistema contaran con una única clave primaria (a excepción de aquellas que fueron generados a causa de un "muchos a muchos"), todas ellas utilizando un identity y su nombre sigue el patrón de "id\_" seguido del nombre de la tabla en singular. Esto nos facilita el seguimiento de las claves foraneas y el manejo general de las tablas.

**Descripción de las tablas del nuevo modelo**

**Clientes:** Tabla que representa a los clientes (huéspedes) existentes en el sistema. El campo "correo\_cliente" es único por cada cliente y el campo "estado\_cliente" representa si este está habilitado o inhabilitado; En caso de no estarlo este no podrá realizar reservas. Dicho estado puede ser modificado por cualquier recepcionista.

**PK:** id\_cliente **FK:** id\_tipo\_documento, id\_pais.

**Cuentas:** Tabla que posee los datos necesarios para el proceso de log in. Teniendo en ella tanto su username como su respectiva contraseña y guardando los intentos de logeo fallidos junto a estos. El username es único en todo el sistema.

**PK/FK:** id\_usuario

**Usuarios:** Tabla que tiene la información que detalla a cada uno de los usuario representando a administradores o a recepcionistas. El campo "estado\_usuario" nos permite inhabilitar al susodicho en caso de una baja, convirtiendo a esta en solo una baja lógica.

**PK:** id\_usuario  **FK:** id\_tipo\_documento

**Tipos\_de\_documento:** Tabla que alberga los tipos de documento validos teniendo uno extra que representa un tipo de documento desconocido.

**PK:** id\_tipo\_documento

**Items\_factura:** Tabla que representa cada renglón de un factura determinada teniendo en este la información de un consumo determinado.

**PK:** id\_item\_factura **FK:** id\_consumo, id\_factura

**Facturas:** Tabla que posee un registro de todas las facturas del sistema guardando consigo su correspondiente total.

**PK:** id\_factura **FK:** id\_estadia, id\_forma\_de\_pago

**Formas\_de\_pago:** Tabla que contiene todas las formas de pago aceptadas por nuestro sistema. Teniendo una forma extra que representa un medio de pago desconocido.

**PK:** id\_forma\_de\_pago

**Consumos:** Esta tabla representa un consumo de un renglón de una determinada factura, guardando también la fecha y la hora en que el consumible fue ordenado.

**PK:** id\_consumo **FK:** id\_consumible, id\_estadia, id\_habitacion

**ClientesXEstadias:** Tabla que guarda todas las estadías de un cliente y todos los clientes que estuvieron en dicha estadía. Sirve para representar el "muchos a muchos" entre Clientes y Estadias.

**PK/FK:** id\_cliente, id\_estadia

**Cierres\_Temporales\_Habitacion:** Tabla que contiene los datos de la baja temporal de una habitación en concreto. Guardando consigo tanto la fecha de inicio como la de fin y el motivo en particular de la baja momentánea.

**PK:** id\_cierre\_temporales\_habitacion **FK:** id\_habitacion

**Consumibles:** Tabla que contiene la información de todos los consumibles que se pueden pedir guardando su precio y descripción.

**PK:** id\_consumible

**Estadias:** Tabla que contiene la información de todo tipo de estadías que intervengan en el sistema.

**PK:** id\_estadia **FK:** id\_reserva, id\_usuario

**EstadiasXHabitaciones:** Tabla que contiene las habitaciones de una determinada estadía y todas las estadías que tuvo una habitación. Sirve para representar el "muchos a muchos" entre Estadías y habitaciones.

**PK/FK:** id\_estadia, id\_habitacion

**Habitaciones:** Tabla que contiene la información de todas las habitaciones de un hotel. No puede haber dentro de un mismo hotel 2 habitaciones con el mismo número.

**PK:** id\_habitacion **FK:** id\_hotel, id\_tipo\_habitacion

**ReservasXHabitaciones:** Tabla que contiene todas las reservas de una determinada habitacion y las habitaciones de una determinada reserva. Sirve para representar el "muchos a muchos" entre Reservas y Habitaciones.

**PK/FK:** id\_reserva, id\_habitacion

**Tipos\_habitacion:** Tabla que contiene los tipos posibles que puede adoptar una habitación. Guardando consigo cuantos huespedes pueden entrar y su porcentual.

**PK:** id\_tipo\_habitacion

**UsuariosXRoles:** Tabla que contiene que roles tiene que usuario y que usuarios tienen un determinado rol. Sirve para representar el "muchos a muchos" entre Usuarios y Roles.

**PK/FK:** id\_rol, id\_usuario

**Cancelaciones\_Reserva:** Tabla que guarda los datos de las cancelaciones de reservas. Guardando quien, cuando y porque realizo la cancelación.

**PK:** id\_cancelacion\_reserva **FK:**id\_usuario

**Reservas:** Tabla que guarda la información pertinente a las reservas, esta posee un campo de estado para poder determinar en que situación se encuentran cada una de ellas.

**PK:** id\_reserva **FK:** id\_usuario, id\_regimen, id\_estado\_reserva

**Estados\_reserva:** Tabla que guarda todos los posibles estados que pueda llegar a tener una reserva.

**PK:** id\_estado\_reserva

**UsuariosXHoteles:** Tabla que tiene que usuarios pertenecen a un determinado hotel y a que hoteles pertenece un usuario.Sirve para representar el "muchos a muchos" entre Usuarios y Hoteles.

**PK/FK:** id\_usuario, id\_hotel

**Paises:** Tabla que contiene el nombre de todos los países del mundo.

**PK:** id\_pais

**Roles:** Tabla que contiene todos los roles posibles que pueden tener los usuarios en el sistema, teniendo también un campo de estado para saber si este está activo o no.

**PK:** id\_rol

**RolesXFuncionalidades:** Tabla que contiene que funcionalidades tiene un determinado rol y que roles tienen una determinada funcionalidad. Sirve para representar el "muchos a muchos" entre Roles y Funcionalidades.

**PK/FK:** id\_rol, id\_funcionalidad

**Regímenes:** Tabla que contiene la información de los distintos regímenes aceptados por el sistema.

**PK:** id\_regimen

**HotelesXRegimenes:** Tabla que contiene que regímenes tiene un hotel y que hoteles tienen un determinado régimen. Sirve para representar el "muchos a muchos" entre Hoteles y Regímenes.

**PK/FK:** id\_hotel, id\_regimen

**Hoteles:** Tabla que sirve para registrar la información importante de todos los hoteles que alberga nuestro sistema.

**PK:** id\_hotel **FK:** id\_pais

**Funcionalidades:** Tabla que contiene todas las funcionalidades que pueden llegar a tener los distintos roles de este.

**PK:** id\_funcionalidad

**Cierres\_Temporales\_Hotel:** Tabla que contiene los datos de la baja temporal de un hotel en concreto. Guardando consigo tanto la fecha de inicio como la de fin y el motivo en particular de la baja momentánea.

**PK:** Id\_cierre\_temporal\_hotel **FK:** id\_hotel

**Descripciones de los Stored Procedures, Funciones y Triggers**

**Procedures de migración**

**(CREAR\_TIPOS\_HABITACION, CREAR\_RESERVAS, CREAR\_CONSUMIBLES, CREAR\_FACTURAS):**

Estos procedures se utilizan al momento de migrar los datos de la tabla maestra a las distintas tablas de nuestro sistema y estos insertan distintos datos (detallados en sus nombres) en la tabla correspondiente, haciendo que la id que tenia previamente encaje con la nueva que tendrá en el nuevo sistema.

**Procedures**

**OBTENER\_USUARIO\_DUMMY:** Procedure que devuelve el id del usuario Dummy. Necesario a la hora de que entre un cliente sin usuario.

**OBTENER\_ROL\_GUEST:** Procedure que devuelve el id del rol Guest. Este procedure es necesario cuando entra algun cliente sin usuario.

**OBTENER\_FUNCIONALIDADES:** Procedure que devuelve la información que hay en la tablas de funcionalidades del sistema. Este se usa en el abm de roles, para poder cargar todas las funcionalidades disponibles.

**INSERTAR\_NUEVO\_ROL:** Procedure que teniendo el nombre del nuevo role y una tablas con las funcionalidades que este tendrá crea dicho Rol en la tabla de roles y agrega sus funcionalidades en RolesXFuncionalidades.

**OBTENER\_ROLES\_FILTRADOS:** Procedure que recibe un nombre y una funcionalidad el cual filtra y devuelve los roles que cumplan con dicho filtro (O sea que tengan esa funcionalidad o que su nombre se igual y/o contenga el nombre deseado). Usado principalmente tanto para la modificación como para la baja de roles. También posee un tercer campo el cual nos sirve para traer todos los roles o solo los que están activos que cumplan esa condición.

**OBTENER\_FUNCIONALIDADES\_DE\_ROL:** Procedure que recibe un id de un determinado rol y nos devuelve todas las funcionalidades que este posea (solo su id). Este se usa tanto para la modificación como para la baja de roles.

**MODIFICIAR\_ROL:** Procedure que modifica un rol en base a un nombre, funcionalidades y estado. Este setea el nuevo nombre, sus nuevas funcionalidades y su nuevo estado y se usa para la modificación de un rol.

**VALIDAR\_ROL\_USUARIO:** Procedure que valida que el rol activo del usuario tenga la funcionalidad que desea realizar. Esto nos permitirá tener un chequeo de seguridad extra, en caso de que por alguna razón algún usuario este usando una función que no le corresponda.

**OBTENER\_HOTELES:** procedure que devuelve todos los hoteles que hay en el sistema, junto a su correspondiente información.

**OBTENER\_ROLES:** Procedure que devuelve todos los roles que hay en el sistema, trayendo toda la información correspondiente.

**OBTENER\_ROLES\_DE\_UN\_USUARIO:** Procedure que recibe una id de un usuario y devuelve todos los roles que tenga el susodicho. Este procedure se usa para la ABM de usuarios.

**OBTENER\_HOTELES\_DE\_UN\_USUARIO:** Procedure que recibe una id de un usuario y devuelve todos los hoteles que tenga el susodicho. Este procedure se usa para la ABM de usuarios.

**OBTENER\_PAIS:** Procedure que recibe una id de un país y devuelve el nombre de dicho país.

**OBTENER\_TIPOS\_DOCUMENTO:** Procedure que devuelve toda la información que se encuentra en la tabla tipos\_documento.

**INSERTAR\_USUARIO:** Procedure que recibe los datos de un usuario e inserta este en la tabla usuarios. Agregando también la información correspondiente en las tablas UsuariosXRoles, UsuariosXHoteles y Cuentas. El mismo es usado para la creación de un nuevo usuario.

**Triggers**

**Add\_dummy\_user\_for\_new\_hotel:** Trigger que se activa al insertar un nuevo hotel al sistema. Este le agrega al usuario dummy el nuevo hotel que fue insertado, permitiéndole a este usuario tener al nuevo hotel como uno de los hoteles donde el se desempeña.

**Detalle de cada funcionalidad implementada**

**Migración:**

Para realizar esta tomamos las siguientes consideraciones:

* Crear el Schema a utilizar y luego las nuevas tablas que habíamos diseñado previamente. En esta primera instancia no asignamos las claves foraneas ya que eso nos permitirá una migración más sencilla.
* Poblar las tablas que no tengan ningun dato que venga del esquema maestro que nos suministraron. Nos referimos a poblar las tablas "Paises", "EstadosReserva", "FormasDePago","Funcionalidades","Roles" ,"RolesXFuncionalidades" y "TiposDeDocumentación" con la información que queremos que soporte el sistema; Por ejemplo en la tabla países la poblamos con todos los países del mundo, la de formas de pago con los medios de pago que queremos que soporte le nuevo sistema, etc.
* Crear un usuario Dummy. Este usuario le permitirá a cualquier visitante sin cuenta poder hacer, modificar o borrar reservas.
* Migrar aquellas tablas que ya tenían un id mediante procedures los cuales las insertan en bloques usando "INSERT INTO" en la tabla correspondiente. Esto nos permitió aseguramos de que cada nueva id empezara desde el mismo número que el dato original (Por ejemplo id\_tipo\_habitacion arranca desde el 1001 al igual que lo hace Habitacion\_tipo\_codigo) para que no haya discrepancias. Para poder ingresarlos estos tienen que venir ordenados y sin ningún valor faltante, al ser este el caso que cumple el esquema maestro decidimos ir por esta estrategia para poblar algunas tablas.
* Utilizamos un cursor para la migración de las tablas restantes aprovechando de que por cada columna tuviéramos las claves primarias de todas las tablas de interés facilitando la inserción de las futuras claves foraneas y de las claves primarias compuestas de algunas tablas.
* Finalmente modificamos las tablas que ya fueron creadas y pobladas, agregándoles las claves foraneas correspondientes.

**Detalles generales de los ABM:**

En general todos los ABM van a estar distribuidos de una forma parecida: El alta y la modificación irán juntas permitiéndonos crear y modificar al mismo tiempo de una forma sencilla y cómoda. Mientras que la baja estará en otra ventana aparte, dejándonos ver un listado de los datos existentes y borrarlos a placer.

Pasada la ventana de logueo y la general, todas las ventanas poseen 2 listados y 2 pequeños botones (un "<" y un ">") los cuales se encargan de guiarnos a la hora de crear, modificar o dar de baja (Por ejemplo a un rol) para luego con un botón confirmar y ejecutar la acción deseada.

En caso de error se mostrara un mensaje con todos los campos que no hayan cumplido los criterios de validación requeridos.

**ABM Roles:**

La ventana de creación y modificación de roles nos muestra 2 listados: A la izquierda el listado con los roles existentes en el sistema y a la derecha las funcionalidades disponibles. El usuario escribirá el nombre del nuevo rol en la text box de arriba y seleccionara las funcionalidades deseadas, agregándolas al presionar el botón "<“. Finalmente al presionar "agregar nuevo rol" se confirmara la acción y se ejecutara creando en nuestra base de datos el nuevo rol en la tabla de Roles y agregando sus funcionalidades en la tabla RolesXFuncionalidades. Si el usuario desea modificar algún rol le aparecerán los roles existentes y al apretar "modificar" lo llevara a la misma ventana de creación de rol solo que en este caso solo podrá agregar o sacar funcionalidades a dicho rol. Por último la ventana de borrado nos permitirá dar una baja lógica de un rol en especifico con tan solo 2 clicks, en su proceso nos mostrara también las funcionalidades que este posee.

**ABM Usuarios:**

La ventana de creación de usuario nos permite setear su contraseña, nombre de usuario, nombre, apellido, año de nacimiento (desplegando un calendario a la hora de elegir dicha fecha y no permitiendo que se ingresen fechas futuras a la actual), su tipo y numero de documento (Desplegando una lista mostrando los tipos de documento soportados por el sistema), su correo, sus roles y los hoteles donde se desempeñara (Mostrando en el caso de estos últimos una lista con los roles y hoteles disponibles y a su izquierda los que tendrá el nuevo usuario). Finalmente, clickeando el botón "agregar nuevo usuario", se creara al usuario con las características pedidas.