UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE GUAYAQUIL

FACULTAD DE INGENIERIAS

CARRERA DE INGENIERÍA DE SISTEMAS

PROYECTO FINAL PREVIO A LA OBTENCIÓN DEL TÍTULO DE:

INGENIERO DE SISTEMAS CON MENCIÓN EN INFORMÁTICA PARA LA GESTIÓN

TEMA: *"SISTEMA DE GESTIÓN COMERCIAL PARA CONSECIONARIOS AUTOMATRICES"*

PROYECTO:
TERRASOFT

AUTORES:
Bogdan Gabriel Babici
Doris Cristina Tierra Montero
Ma. del Carmen Achig Calderón

Director del Proyecto:
Ing. Ricardo Naranjo S.

Guayaquil, Noviembre del 2010

DEDICATORIA

A Dios todopoderoso y eterno

A mi familia

A mis amigos y amigas.

La humildad trae gracia y felicidad a la vida; permite acomodarse a las situaciones difíciles sin pensar en lo que se está dejando ó renunciando, nos vuelve más sencillos y naturales, permite que nos concentremos en lo que estamos haciendo, y que lo hagamos correctamente.

La humildad hace que podamos ver los beneficios en cada escena de la vida, haciendo que nuestras interacciones giren en un ambiente más agradable, así logramos un lugar en el corazón de todos, eliminando en un segundo aquello que nos hiere y no nos deja crecer.

Siendo humildes comprenderemos que aún tenemos mucho por mejorar, mucho que aprender y que podemos ocuparnos en la tarea de crecer.

María del Carmen.

AGRADECIMIENTO

A Dios por ser mi mejor amigo, fortaleza, darme todo lo que tengo y no dejarme caer nunca.

Mi familia, en especial mis queridos padres, por su comprensión y ayuda, porque me han enseñado a encarar las adversidades sin perder nunca la dignidad ni desfallecer en el intento. Me han dado todo lo que soy como persona: valores, principios, perseverancia, empeño, y todo ello con una gran dosis de amor y sin pedir nunca nada a cambio.

Mi novio, por su amor, apoyo incondicional y motivación en la realización de mis anhelos por ser mi ángel y llegar en el momento que más lo necesitaba.

Mis distinguidos Maestros y tutor, modelos de valor y sabiduría, por su desinteresada y generosa labor de transmisión del saber, su inagotable entusiasmo y sus acertados consejos y sugerencias.

Mis amigos y compañeros de tesis, un reconocimiento sincero, sin la constancia y trabajo en equipo no hubiera podido llevar a cabo mis aspiraciones.

Un agradecimiento especial a la UNIVERSIDAD POLITÉCNICA SALESIANA, CARRERA DE INGENIERÍA DE SISTEMAS y a todas aquellas personas que me han apoyado incondicionalmente permitiéndome realizar mis estudios universitarios. A todos aquellos que han intervenido en mi formación. ¡MUCHAS GRACIAS!

María del Carmen.

TRIBUNAL DE GRADO

ING. XXXX
Miembro del Tribunal

ING. XXXX
Miembro del Tribunal

ING. XXXXX
Director de Tesis

ING. XXXXX
Sub-decano de la UPS

DECLARACIÓN EXPRESA

"La responsabilidad por los hechos, ideas y doctrinas expuestos en esta tesis, me corresponde exclusivamente; y, el patrimonio intelectual de la misma, a la

UNIVERSIDAD POLITÉCNICA SALESIANA".

(Reglamento de Exámenes y Títulos profesionales de la U.P.S.)

_____

Bogdan Gabriel Babici.

_____

Doris Cristina Tierra M.

_____

María del Carmen Achig C.

INDICE

1. INTRODUCCIÓN 7
2. PLANTEAMIENTO DEL PROBLEMA 9
3. RESUMEN DE LA PROPUESTA 11
3.1 Exposición del problema que se desea solucionar 11
3.2 Descripción de los límites del proyecto 13
3.3 Lenguaje de programación 13
4. OBJETIVOS 15
4.1 Objetivos Generales 15
4.2 Objetivo Específico 15
5. BENEFICIARIOS DE LA PROPUESTA DE INTERVENCIÓN 15
6. MARCO REFERENCIAL 16
6.1 MARCO TEORICO 16
CAPITULO 1 16
1.1 ANALISIS Y DIAGRAMACION DE PROCESOS EMPRESARIALES 16
1.2 Sistemas Indicadores de Gestión y BSC 16
CAPITULO 2 30
2.1 ANALISIS Y DISEÑO DE SISTEMAS ORIENTADO A OBJETOS 30
CAPITULO 3 30
3.1 INTRODUCCION A LA ADMINISTRACION DE BASE DE DATOS ORACLE 10G Y AL APLICATION SERVER 10G 30
CAPITULO 4 30
4.1 PROGRAMACION EN PL/SQL ORACLE 30
CAPITULO 5 35
5.1 Generalidades de la Base de Datos de ORACLE 10g. 35
6.6 MARCO CONCEPTUAL 52
7. MANUAL DE USUARIO

8. CRONOGRAMA DE ACTIVIDADES 59
9.PRESUPUESTO

10. RECOMENDACIONES

11. CONCLUSIONES

12.Bibliografía 197
13. REFERENCIAS WEB 198
14. ANEXOS.

INDICE DE TABLAS

TABLA 1: Activos 75
TABLA 2: Agenda 77
TABLA 10: Gastos 86
TABLA 11: Marcas 87
TABLA 12: Parientes 87
TABLA 13: Pasivos 88
TABLA 14: RefBancarias 90
TABLA 15: RefComerciales 91
TABLA 16: RefTarjetas 92
TABLA 17: Seguros 94
TABLA 18: TrabajoCliente 95
TABLA 19: TrabajoConyuge 96
TABLA 20: Usuarios 98
TABLA 21: Vehiculos 99
TABLA 22: Ventas 100
TABLA 23: Viviendas 101
TABLA 3: Bancos 77
TABLA 4: Categorias 78
TABLA 5: Clientes 79
TABLA 6: Conyuges 82
TABLA 7: Creditos 83
TABLA 8: Dealer 84
TABLA 9: Dispositivos 85

INDICE DE GRÁFICOS

Gráfico #01 Diagrama causa/efecto 10
Gráfico #02 Descripción del problema 13
Gráfico #03 Diagrama de Flujo - Simbología 19
Gráfico #04 Representación de Indicadores 22
Gráfico #05 De la organización vertical a la horizontal 25
Gráfico #06 Tipos de Diagramas Visio 28
Gráfico #07 Tipos de Diagramas UML 28
Gráfico #08 Esquema de las 4 acciones 40
Gráfico #09 Indicadores gráficos de medición 48
Gráfico #10 Estructura Oracle 10g 58
Gráfico #11 Base de Datos- Tablespace 60
Gráfico # 12 Relacion segmentos, extensiones y bloques 61
Gráfico #13 Arquitectura Oracle 63
Gráfico # 14 Instancia de Oracle 64
Gráfico # 15 Instancia de Oracle 67
Gráfico # 16 Estructura de Oracle 75
Gráfico #17 Tipos de Archivos 76
Gráfico # 18 Espacio System 77
Gráfico #19 DBA 78
Gráfico # 20 Esquema de Objetos 80
Gráfico #20 Extensión y Segmento de Tabla 79
Gráfico # 21 SGA 82
Gráfico # 22 Tipos de Datos Oracle 83
Gráfico # 23Presupesto Terrasoft 232

1. INTRODUCCIÓN

Una de las principales preocupaciones de la Gerencia General de todo concesionario automotriz, distribuidor autorizado de una reconocida marca de vehículos de nuestro medio, es la satisfacción de sus compradores de vehículos nuevos.

Aunque los Concesionarios Automotrices son grandes empresas regidas por Normas y Estándares de sus Marcas a nivel internacional, éstos cuentan con Sistemas Informáticos para su Gestión Financiera, Contable y de Facturación, pero desafortunadamente ninguno de ellos cuenta con un Sistema enfocado netamente en automatizar, optimizar y agilizar de forma más eficiente la Gestión realizada rutinariamente por el Área de Ventas.

Es muy común ver:

- A vendedores que llevan los datos de sus clientes en agendas.
- Que traspasan solicitudes de créditos para varios Bancos varias veces, a mano, siendo la Información del Cliente una sola.
- Que llenan sus escritorios de Papeles con las respuestas de créditos de varios Bancos y tienen que buscar entre todas ellas (en especial los menos ordenados) para saber ¿clientes aprobados o negados?, en que Institución Bancaria aplica el crédito?, en qué condiciones se encuentra el crédito?, etc.
- Que llevan las cotizaciones de sus clientes en un Bloc de Proformas y cuando quieren saber que Vehículo desea un Cliente tienen que buscar en todo el Bloc
- Que tienen que elaborar semanal o quincenalmente archivos en Excel con la información sobre su Gestión realizada diariamente

Todo esto conlleva a desperdiciar el recurso más importante dentro de toda organización, el tiempo, el mismo que los vendedores deberían emplear en atender más clientes, o hacer más seguimiento a los que ya tienen y registrar una mayor cantidad de ventas.

2. PLANTEAMIENTO DEL PROBLEMA

Descripción.-

El problema en el que nuestro grupo se ha enfocado es la falta de un sistema informático para la automatización en el Proceso de Gestión Comercial llevado a cabo por los Asesores de Ventas de un Concesionario.

Gráfico #01

Diagrama causa /efecto

Causa	Efecto
Los Vendedores actualmente no cuentan con un sistema de control y apoyo a sus tareas diarias.	• La información es registrada en Agendas, Papeles, Tarjetas de los Clientes, Proformas, lo cual hace difícil y lenta la Consulta de Información. • No se puede almacenar información histórica mayor a ciertos meses o un año debido a la falta de espacio en el escritorio para tantos papeles. • Se llenan las solicitudes de crédito a mano una y otra vez por cada Banco al que se quiere aplicar crédito al cliente. • Pérdida de Tiempo que podría emplearse en atender más clientes, hacer seguimientos y cerrar negocios
Los Asesores de Crédito de los Vendedores (en los concesionarios que tienen estos cargos) comúnmente conocidos como F&I's que	• Falta de un Control Eficiente y Rápido en la Gestión de Créditos • Los vendedores no obtienen sus Respuestas de Crédito a tiempo para poder seguir con el proceso de venta

tienen que gestionar los créditos de todos los clientes de cada Vendedor en cada Banco o Financiera tampoco cuentan con un Software que administre su Gestión	• Los clientes pueden marcharse a otro lugar donde gestionan su compra más rápido
Los Jefes de Ventas y otras Gerencias solo cuentan con un Sistema de Facturación que permite ver las Ventas pero no hay un Sistema que les brinde reportes de la gestión realizada por los vendedores, y en ciertas agencias que no están en red con la Matriz los Jefes de Ventas de las mismas no cuentan con ni siquiera eso	• No contar con reportes oportunos y en tiempo real de las gestiones de cada vendedor, lo cual no les permite tomar acciones a tiempo

Elaborado por: Bogdan Babici, Doris Tierra, María Achig

Fuente: Bogdan Babici.

3. RESUMEN DE LA PROPUESTA

Nuestra propuesta para el desarrollo de Terrasoft básicamente consisten en:

- ❖ Reducir el tiempo de la Gestión del Vendedor en tramitar los Créditos de sus Clientes.
- ❖ Facilitar el Ingreso de Solicitudes de Crédito a los Bancos llenando una sola vez la solicitud de crédito del cliente, almacenándola y luego poder traspasarla a cualquier Banco mediante un solo clic las veces que se desee.
- ❖ Facilitar su trabajo y contar con una herramienta de apoyo para el seguimiento de los clientes.
- ❖ Facilitar al Vendedor de una forma segura y eficaz de almacenar y consultar los datos de sus clientes.

- ❖ Facilitar al Vendedor la generación de reportes exigidos por la jefatura de ventas
- ❖ Permitir a los Jefes de Ventas contar con Información actual y disponible de forma rápida para conocer la Gestión de los Asesores Comerciales

3.1 Exposición del problema que se desea solucionar

Descripción.- El problema antes mencionado en el que nuestro grupo de tesis se ha enfocado es la falta de un sistema informático para la automatización en el Proceso de Gestión Comercial llevado a cabo por los Asesores de Ventas de un Concesionario.

Gráfico #02
Descripción del problema

Problema	Descripción
Los Vendedores actualmente no cuentan con un sistema de control y apoyo a sus tareas diarias.	• La información es registrada en Agendas, Papeles, Tarjetas de los Clientes, Proformas, lo cual hace difícil y lenta la Consulta de Información. • No se puede almacenar información histórica mayor a ciertos meses o un año debido a la falta de espacio en el escritorio para tantos papeles. • Se llenan las solicitudes de crédito a mano una y otra vez por cada Banco al que se quiere aplicar crédito al cliente. • Pérdida de Tiempo que podría emplearse en atender más clientes, hacer seguimientos y cerrar negocios.
Los Asesores de Crédito de los	• Falta de un Control Eficiente y

Vendedores (en los concesionarios que tienen estos cargos) comúnmente conocidos como F&I's que tienen que gestionar los créditos de todos los clientes de cada Vendedor en cada Banco o Financiera tampoco cuentan con un Software que administre su Gestión	Rápido en la Gestión de Créditos • Los vendedores no obtienen sus Respuestas de Crédito a tiempo para poder seguir con el proceso de venta • Los clientes pueden marcharse a otro lugar donde gestionan su compra más rápido
Los Jefes de Ventas y otras Gerencias solo cuentan con un Sistema de Facturación que permite ver las Ventas pero no hay un Sistema que les brinde reportes de la gestión realizada por los vendedores, y en ciertas agencias que no están en red con la Matriz los Jefes de Ventas de las mismas no cuentan con ni siquiera eso	• No contar con reportes oportunos y en tiempo real de las gestiones de cada vendedor, lo cual no les permite tomar acciones a tiempo

Elaborado por: Bogdan Babici, Doris Tierra, María Achig.

Fuente: Bogdan Babici.

3.2 Descripción de los límites del proyecto.

La funcionalidad e importancia del sistema será la siguiente:

- Permitir a Vendedores y F&I's registrar los clientes atendidos, así como los datos de sus proformas (Vehículos de Interés, Formas de Pago, etc.)
- Llevar un Registro de los Créditos ingresados por cada Vendedor de sus Clientes en cada Banco con información como:
  - Entrada
  - Plazo
  - Estado del Crédito

- Dispositivo de Seguridad
- Gastos Financiados
- Tasa de Interés
- Condiciones y Observaciones

- Llevar un registro similar a una Agenda Virtual con Recordatorios para hacer seguimiento al cliente, registrar las respuestas de las llamadas y/o citas.
- Ingresar las Facturas realizadas por Vendedor.
- Permitir a Jefes de Ventas y Gerencia contar con Información clara y oportuna de la Gestión de las Ventas a partir de una Base de Datos Centralizada.
- Este sistema funciona en intranet.
- El Sistema no realiza facturación, ni ayuda en la recuperación de carteras vencidas.
- No envía recordatorios a celular

3.3 Lenguaje de programación

El desarrollo del sistema esta realizado en;

- ✓ Desarrollado en Lenguaje de Programación: Oracle Forms
- ✓ Base de Datos: Oracle 10G.
- ✓ Con Reportes en Oracle Forms.

Requerimientos Minimos y Recomendados

Con respecto a los equipos:

- Mínimo Pentium 4
- Memoria de 512 Mb
- Acceso a la Red

4. OBJETIVOS:

4.1 Objetivo General.

Desarrollar un sistema informático que automatice la Gestión de Ventas en un Concesionario Automotríz.

4.2 Objetivos Específicos

En cuanto a los objetivos específicos que se espera lograr con la realización de este proyecto tenemos:

- ✓ Reducir el tiempo de la Gestión del Vendedor en tramitar los Créditos de sus Clientes.
- ✓ Facilitar el Ingreso de Solicitudes de Crédito a los Bancos llenando una sola vez la solicitud de crédito del cliente, almacenándola y luego poder traspasarla a cualquier Banco mediante un solo click las veces que se desee.
- ✓ Facilitar el trabajo de los agentes vendedores y contar con una herramienta de apoyo para el seguimiento de los clientes.
- ✓ Facilitar al Vendedor de una forma segura y eficaz el almacenamiento y consulta de los datos de sus clientes.
- ✓ Facilitar al Vendedor la generación de reportes exigidos por la jefatura de ventas.
- ✓ Permitir a los Jefes de Ventas contar con Información en Línea y disponible siempre de forma rápida sobre la Gestión de los Asesores Comerciales

5. BENEFICIARIOS DE LA PROPUESTA DE INTERVENCIÓN

Con nuestra propuesta se podrá beneficiar cualquier Concesionario Automotriz a nivel nacional brindando mayor apoyo a cuyos usuarios ejerzan los siguientes cargos dentro del Concesionario:

- Gerente General del Concesionario
- Gerentes de Ventas
- Jefes de Ventas
- Asistentes de Créditos (F&I's)

- Vendedores

6. MARCO REFERENCIAL

6.1 MARCO TEORICO

6.1.1 CAPITULO 1

6.1.1.1 ANALISIS Y DIAGRAMACION DE PROCESOS EMPRESARIALES

CONCEPTO DE PROCESO

Cualquier actividad, o conjunto de actividades secuenciales, que transforma elementos de entrada (inputs) en resultados (outputs) puede considerarse como un proceso. Los procesos utilizan recursos para llevar a cabo dicha transformación. Los procesos tienen un inicio y un final bien definidos.

En general en todo proceso se identifican los siguientes elementos:

1. **Elemento procesador**: Son las personas o máquinas que realizan el conjunto de actividades que constituye el proceso.

2. **Secuencia de actividades**: Es la secuencia ordenada de actividades que realiza el elemento procesador.

3. **Entradas (Inputs)**: Son los flujos que requiere el elemento procesador para poder desarrollar su proceso. Ejemplos de inputs son materiales, información, condiciones medioambientales, etc.

4. **Salida (Output)**: Es el flujo que genera el elemento procesador como consecuencia de efectuar la secuencia de actividades que constituyen el proceso. La salida es el flujo resultado del proceso. Ya sea interno o externo.

5. **Recursos**: Son los elementos fijos que emplea el elemento procesador para desarrollar las actividades del proceso. Ejemplos de recursos son las máquinas.

6. **Cliente del proceso**: Es el destinatario del flujo de salida del proceso. Si el destinatario es una persona de la organización se dice que es un cliente interno. Si el destinatario es el usuario final, entonces se trata de un cliente externo.

7. **Expectativas del cliente del proceso con relación al flujo de salida**: Son conceptos que el cliente del proceso espera ver incorporados al flujo de salida del proceso y que si no aparecen será capaz de detectar. Condicionan su satisfacción.

8. **Indicador**: Es la medición de una característica de un proceso.

9. **Responsable del proceso**. Es el propietario del proceso.

TIPOS DE PROCESOS

Toda organización puede representarse como una compleja red de elementos que realizan actividades que les permiten interrelacionarse unas con otras para alcanzar los fines (misión) del conjunto. Cada una de estas interrelaciones puede representarse y gestionarse como un proceso.

En función de la finalidad, los procesos se pueden clasificar en tres categorías: Procesos estratégicos, procesos operativos y procesos de soporte.

- **Procesos estratégicos**

Procesos destinados a definir y controlar las metas de la organización, sus políticas y estrategias.

Son procesos destinados a definir y controlar las metas de la organización, sus políticas y estrategias. Permiten llevar adelante la organización. Están en relación muy directa con la misión/visión de la organización. Involucran personal de primer nivel de la organización.

Afectan a la organización en su totalidad. Ejemplos: Comunicación interna/externa, Planificación, Formulación estratégica, Seguimiento de resultados, Reconocimiento y recompensa, Proceso de calidad total, etc.

- **Procesos operativos**:

Son procesos que permiten generar el producto/servicio que se entrega al cliente, por lo que inciden directamente en la satisfacción del cliente final. Generalmente atraviesan muchas funciones. Son procesos que valoran los clientes y los accionistas. Ejemplos:
Desarrollo del producto, Fidelización de clientes, Producción, Logística integral, Atención al cliente, etc. Los procesos operativos también reciben el nombre de procesos clave.
Son Procesos que permiten generar el producto/servicio que se entrega al cliente. Aportan valor al cliente.

- **Procesos de soporte**:

Apoyan los procesos operativos. Sus clientes son internos. Ejemplos:
Control de calidad, Selección de personal, Formación del personal, Compras, Sistemas de información, etc. Los procesos de soporte también reciben el nombre de procesos de apoyo.

Cuando ya se han identificado todos los grandes procesos de la organización, éstos se representan en un mapa de procesos. Téngase en cuenta que la clasificación de los procesos de una organización en estratégicos, operativos y de soporte, vendrá determinada por la misión de la organización, su visión, su política, etc. Así por ejemplo un proceso en una organización puede ser operativo, mientras que el mismo proceso en otra organización puede ser de soporte.
Son Procesos que abarcan las actividades necesarias para el correcto funcionamiento de los procesos operativos.

MAPA DE PROCESOS

Los procesos identificados en el mapa de procesos son procesos principales, son procesos muy grandes, macroprocesos, que a su vez están formados por subprocesos o microprocesos. El grado de detalle al que debe llegarse, es decir, el número de niveles de subprocesos que debe considerarse depende del tamaño y complejidad de la empresa. Habrá empresas que sólo precisen de la identificación y detalle de los macroprocesos y habrá otros que precisarán un elevado grado de detalle dentro de los subprocesos.

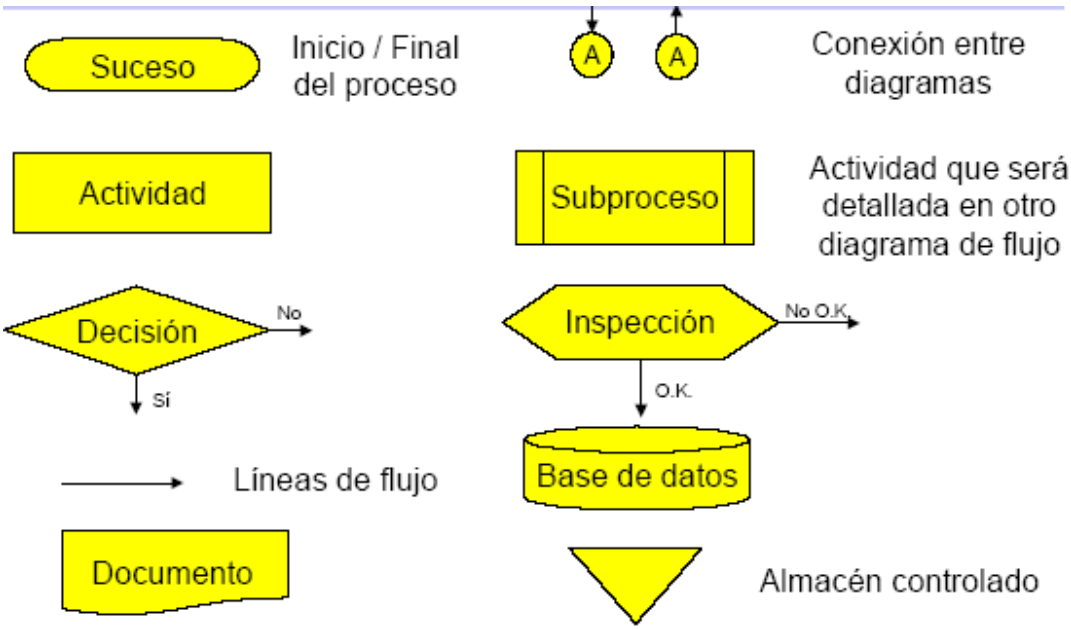
REPRESENTACIÓN DEL PROCESO

Una vez obtenido el mapa de procesos, la organización deberá representar éstos, o sus sub-procesos.

La representación de los procesos, que consiste en desglosar los procesos en sus actividades, posibilita la estandarización de los procesos y la identificación de oportunidades de mejora. Esta representación se puede realizar mediante un diagrama de flujo empleando la simbología mostrada en el gráfico#01.

Gráfico #03

Diagrama de Flujo - Simbología



Fuente: http://uproprod.blogspot.com/2007/08/aprenda-crear-diagramas-de-flujo.html

Elaborado: http://uproprod.blogspot.com/2007/08/aprenda-crear-diagramas-de-flujo.html

Inicio o final de proceso: En su interior situamos materiales, información o acciones para comenzar el proceso o para mostrar el resultado en el final del mismo

Actividad: Tarea o actividad llevada a cabo durante el proceso. Puede tener muchas entradas, pero solo una salida

Decisión: Indicamos puntos en que se toman decisiones: sí o no, abierto o cerrado.

Lineas de flujo: Muestran dirección y sentido del flujo del proceso, conectando los símbolos.

Documento: Se utiliza este símbolo para hacer referencia a la generación o consulta de un documento específico en un punto del proceso.

Conexión entre diagramas: Como su nombre lo indica permite enlazar o conectar diagramas entre si.

Base de Datos: Permite el almacenamiento de información en una base de Datos.

USO DE DIAGRAMAS DE FLUJO DE DATOS

Los diagramas de flujo de datos son útiles a lo largo del proceso de análisis y diseño. Existen compromisos para decidir que tanto deben ser explotados de los flujos de datos. Se desperdiciara tiempo y se sacrificara complusibilidad si los diagramas de flujo de datos son exclusivamente complejos. Por otro lado, si los diagramas de flujo de datos están muy poco explotados, pueden ocurrir errores u omisiones que pueden eventualmente afectar el sistema que esta en desarrollo.

Por ultimo, recuerde que los diagramas del sistema de flujo pueden ser usados para documentar niveles altos o bajos del análisis y para ayudar a sustentar la lógica subyacente en los flujos de datos de la organización.

CARACTERÍSTICAS DE LOS DIAGRAMAS DE FLUJO DE DATOS

Muestran que debe hacer el sistema sin referencias.

Son diagramas explícitos y comprensibles.

Dan la posibilidad de representan el sistema a diferentes niveles de complejidad, desde lo mas global a lo mas detallado solo requieren de 4 símbolos.

Son de fácil mantenimiento, pues los cambios afectan solo algunos de sus elementos y no al todo.

INDICADORES

Un indicador es la medición de una característica de un proceso. Asociar indicadores a un proceso sirven para:

- Analizar la situación actual del proceso en base a hechos y datos.
- Establecer objetivos y planes de futuro consistentes.
- Evaluar y reconocer, con objetividad, el trabajo de las personas y equipos de mejora

implicados en el proceso.
- Gestionar con mayor eficacia los recursos que necesita el proceso.

Los indicadores en una organización deben ser **fiables**, es decir, en idénticas situaciones deben proporcionar el mismo resultado, y **válidos**, es decir, medir aquello que se quiere medir. Además deben ser **pocos** para facilitar su seguimiento.

Existen dos tipos de indicadores:

1. **Indicadores de eficacia**. Miden la manera en la que un proceso cumple sus objetivos. Ejemplos:
Nivel de satisfacción del cliente, % de aumento de ventas, conocimiento de la marca.

2. **Indicadores de eficiencia**. Miden la cantidad de recursos que necesita el proceso para conseguir un determinado nivel de eficacia. Ejemplos: Minutos dedicados a cada paciente, Número de enfermeras en una unidad.

Para definir un indicador, es necesario tener en consideración los siguientes campos:

- ✓ Nombre del indicador: descripción del indicador.
- ✓ Fórmula: modo en que se realizará la medición concreta del mismo.
- ✓ Responsable de recogida: quién se encargará de recoger los datos para el cálculo del indicador.
- ✓ Periodicidad de recogida: cada cuánto tiempo se llevará a cabo la medición del indicador.
- ✓ Responsable de actuación. Es la persona que se encarga de tomar medidas en función de los valores que presente el indicador.
- ✓ Valor objetivo. Es el valor que se pretende que tome el indicador. Si no se consigue este valor, el responsable de actuación debe llevar a cabo acciones de mejora.

Ejemplo:
Nombre del indicador: Quejas mensuales.
Fórmula: Número total de quejas que se reciben durante un mes.

Responsable de recogida: Atención al cliente.

Periodicidad de recogida: Mensual.

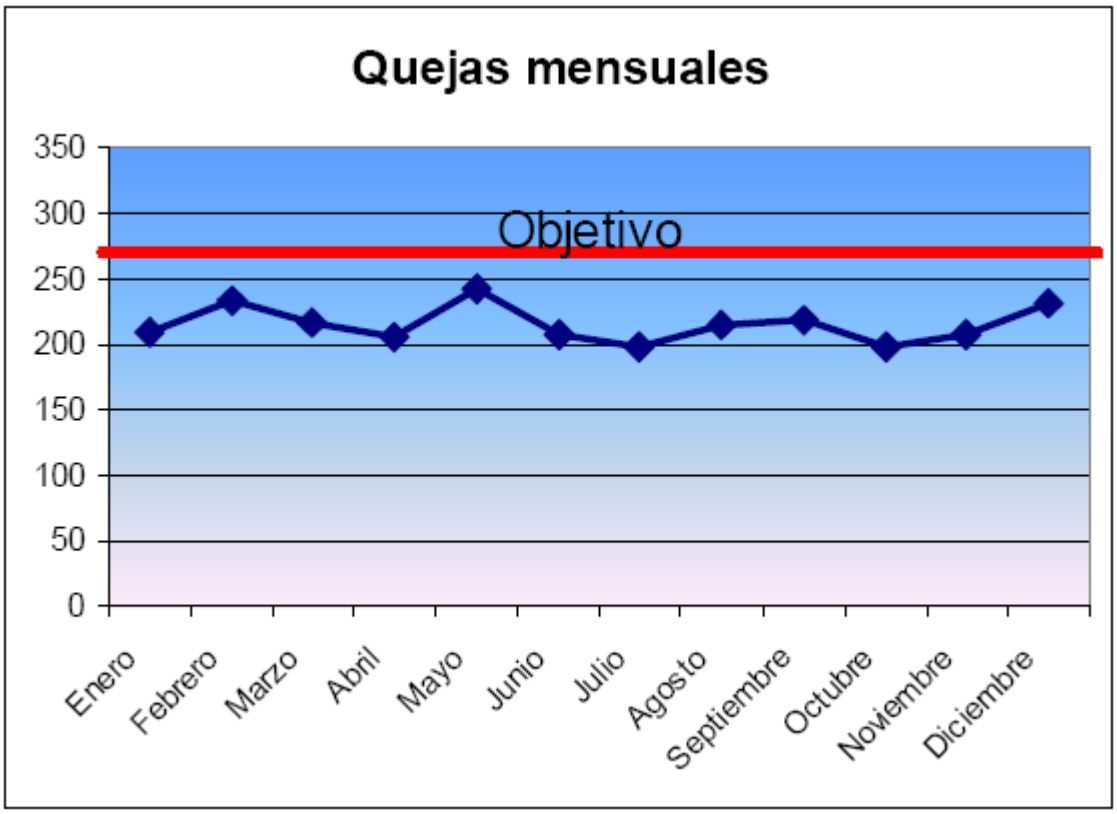
Responsable de actuación: Director de calidad.

Valor objetivo: Menos de 260.

Los indicadores se suelen representar en gráficos para observar su evolución:

Gráfico #04

Representación de Indicadores



Fuente: http://www.gestion-calidad.com/indicadores.html

Elaborado: http://www.gestion-calidad.com/indicadores.html

DE LA ORGANIZACIÓN VERTICAL A LA HORIZONTAL:

GESTIÓN POR PROCESOS.

La **gestión de las organizaciones** ha ido evolucionando a lo largo de toda la época de desarrollo industrial.

La mayor parte de los directores de empresas las consideran todavía desde el punto de vista funcional (departamentos), por tanto tenderán a dirigirlas también funcionalmente, considerando cada departamento como independiente de los demás. Los objetivos se establecerán para cada función por separado. Las reuniones entre los distintos departamentos se limitarán a informar de cómo van las cosas.

En esta atmósfera, los responsables de cada área funcional tienden a considerar a las otras áreas funcionales como enemigos, en lugar de verlas como miembros de un mismo equipo. Alrededor de los departamentos se levantan unos búnkers. Estos búnkers normalmente impiden que las cuestiones interdepartamentales se puedan resolver entre empleados de igual rango de los niveles inferiores.

Todo asunto que afecta a varias áreas funcionales, como el establecimiento de calendarios o normas de producción y entregas, asciende hasta la cima del búnker, para que el responsable de un área trate de esos asuntos con el responsable de las otras, y busquen una solución. Después, ambos responsables comunicarán la resolución en sentido descendente hasta el nivel en el que se debe realizar el trabajo.

La realidad actual obliga a la mayoría de las empresas a competir en un mercado de compradores.
Ahora se necesita una forma diferente de enfocar, de analizar, y de dirigir las empresas. Se debe dirigir una organización considerándola como un sistema integrado de procesos.

La **gestión por procesos** es esta nueva forma diferente de dirigir las organizaciones. Se pasa de una visión vertical de la organización a una visión horizontal que permite gestionar la organización no como un grupo de funciones heterogéneas (departamentos), sino como un **sistema** formado por flujos, **procesos**, que satisfacen

y superan las necesidades y expectativas “razonables” de los destinatarios de los mismos, los clientes.

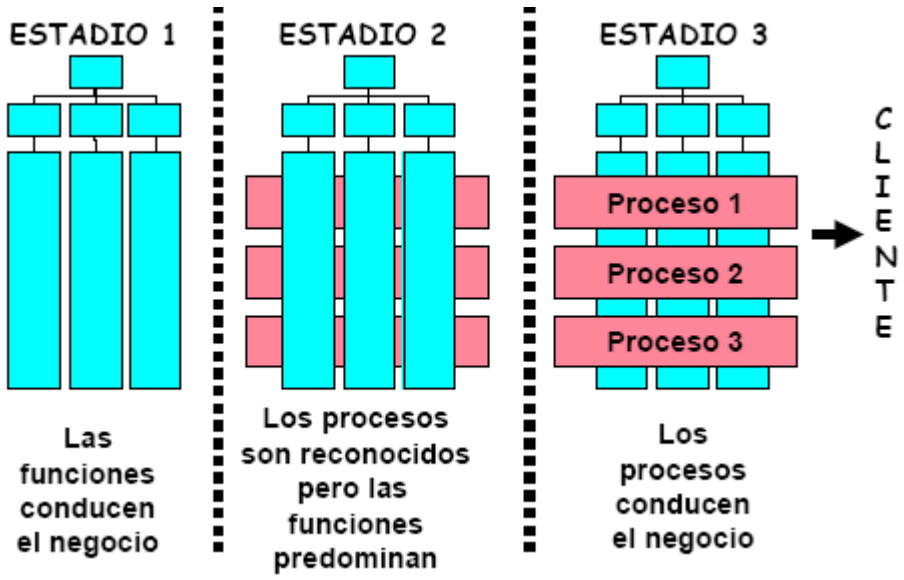
EVOLUCIÓN DE UNA ORGANIZACIÓN VERTICAL A UNA ORGANIZACIÓN HORIZONTAL

El paso de una visión vertical de la organización a una visión horizontal no se realiza de forma brusca, si no que pasa por diferentes estadios. Existe un primer estadio en el que dominan las funciones, la organización por departamentos. En un segundo estadio se comienza a reconocer que los procesos dentro de la organización son importantes, pero todavía dominan las funciones.

Finalmente, en el tercer estadio, se reconoce la supremacía de los procesos sobre las funciones y se trabaja con una visión horizontal de la organización.

Gráfico #05

De la organización vertical a la horizontal



Fuente: Topico de Graduación UPS.

La **gestión por procesos** consiste en entender la organización como un conjunto de procesos que traspasan horizontalmente las funciones verticales de la organización. Permite asociar objetivos a estos procesos, de tal manera que se cumplan los objetivos de los departamentos para conseguir finalmente los objetivos de la

organización. Los objetivos de los procesos deben corresponderse con las necesidades y expectativas de los clientes.

PROPIETARIO DEL PROCESO

Cada uno de los procesos que se hayan identificado en la organización debe tener un responsable del mismo. Este responsable recibe el nombre de propietario del proceso. Normalmente, el propietario de un proceso suele ser el director de alguno de los departamentos de la organización.

Las funciones principales que desempeña el propietario de un proceso son:

- ✓ Comprender y diseñar el proceso para alcanzar los objetivos de la organización.
- ✓ Establecer un conjunto de medidas que permitan controlar el proceso e identificar oportunidades de mejora.
- ✓ Dotar a las personas que operan el proceso de la información, herramientas y sistemas que necesitan para dar servicio a los clientes.
- ✓ Revisar el rendimiento del proceso y acometer proyectos de mejora.

Herramientas para el modelado de procesos

Herramienta BPwin

Es una herramienta de modelamiento de procesos de negocios que te permite capturar las actividades (léase procesos o funciones) del negocio para entenderla mejor, modelarla y cambiarla.

Utiliza dos metodologías:

IDEF0 (ICAM DEFinitions 0,Definiciones del Programa de Manufactura

Es una representación gráfica y representación del lenguaje natural de las interacciones de las actividades del negocio y los recursos necesarios para producir salida. Soporta la descripción gráfica de funciones de negocios como un conjunto de actividades interrelacionadas y la información o recursos requeridos por cada actividad.

El propósito de un modelo **IDEF0** es efectuar la documentación y reestructuración de las funciones que sirvan para una mejor eficiencia y efectividad).

El BPwin utiliza las siguientes definiciones:

ACTIVITY (ACTIVIDAD): Una Actividad es un proceso, función o tarea nombrada que ocurre sobre un periodo y, produce resultados reconocidos.

ARROWs o FLECHAS: Se define a una actividad por la representación de parámetros que actúan hacia la actividad o desde la actividad hacia afuera.
También, se puede definir a una ACTIVIDAD.

Básicamente, lo que hace es conducir el modelo por las actividades funcionales, y determinar : ¿ cuál de las actividades son la más eficientes ?, ¿ cuáles son las más efectivas en términos de costos y cuáles toman menos tiempo o cuánto tiempo ?. Permitiendo mostrar el costo por cada actividad.

Se pueden calcular los costos basados en actividades y ver cuan eficiente es la organización, empleando la herramienta de costeo EasyABC (Activity Based Costing, Costeo basado en actividades) de la empresa ABC Technologies. El ABC es una técnica para capturar y analizar costos de actividades, en el trabajo de los Centros de Costo (como Ventas, Marketing, Producción, Contabilidad, RR.PP., Administración, Créditos, Almacén y otros) son identificados y estos son asignados a las actividades. En caso, de no disponer de una herramienta de costeo se puede recurrir al MS Excel o MS Word for Windows 6.0c y versiones posteriores, para importar o exportar información.

Además, BPwin trabaja estrecha-mente con el ERwin a través de un enlace bidireccional donde el BPwin realiza la interfase con el archivo: *.BPX y le envía información al ERwin, concerniente a las

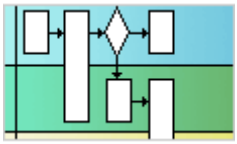
entidades y atributos identificados; en forma viceversa, el ERwin le envía el archivo: *.EAX con información de entidades y atributos al BPwin.

Microsoft Visio 2003

Las herramientas que lo componen permiten realizar diagramas de oficinas, diagramas de bases de datos, diagramas de flujo.

Los diagramas de flujo de Visio tienen como finalidad ver los procesos empresariales completos, así podrá obtener una visión clara de qué funciona y qué no funciona en el flujo de tareas e información de su equipo e identificar áreas de posibles mejoras. También puede documentar sus soluciones a esos problemas, explicando los pasos de sus procesos con el nivel de detalle necesario.

Gráfico #06
Tipos de Diagramas Visio

Tipo de Diagrama	Demostración	Objetivo
Diagrama de flujo de		Describir, analizar y documentar el flujo de información y las tareas de automatización de los procesos, contabilidad, administración y recursos humanos.
Diagrama de flujo de referencias cruzadas		Mostrar la relación entre un proceso empresarial y las unidades organizativas o funcionales responsables del mismo.
Diagrama de causa y efecto Ishikawa		Documentar los factores que intervienen en un resultado determinado. Se utiliza para revisar los factores que contribuyen a una situación concreta.

Fuente: http://webdeinformatica.blogspot.com/
Elaborado por: Tierra Doris, Babici Bogdan, Achig María.

VISUAL PARADIGM UML

El paradigma visual para UML es un Lenguaje de Modelado Unificado (UML) herramienta de diseño que soporta todos los diagramas UML, SysML esquemas y diagrama de entidad-relación. Visual Paradigm for UML permite el uso de modelado caso amplias funciones incluyendo la función completa de uso UML Diagrama de casos , editor de flujo de eventos , casos de uso / rejilla actor y la generación de un diagrama de actividad . Visual Paradigm for UML genera código Java.

Gráfico #07
Tipos de Diagramas UML

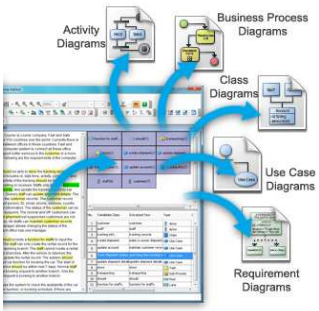
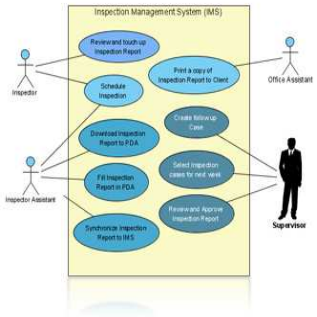
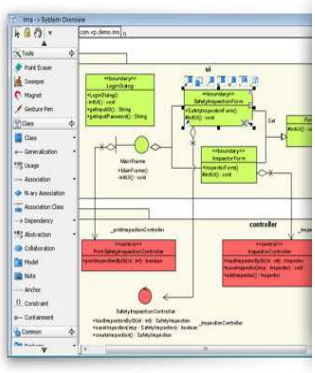
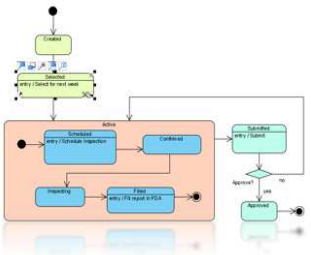
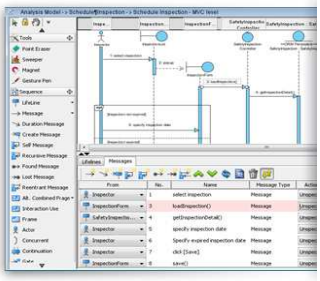
Nombre	Modelo	Descripción
Análisis textual		El proceso normal del análisis textual es recoger o preparar el documento de texto. El análisis textual admite la extracción de casos de uso, actor, la clase, acción, tareas, sub-proceso y el evento.
Diagrama de casos de uso		Está diseñado para visualizar y documentar el comportamiento de un sistema. Las notaciones principales del diagrama de casos de uso incluyen un sistema (objeto), casos de uso, Actor de la Asociación (enlace de comunicación), dependencia y generalización. Diagrama de casos de uso se utiliza generalmente en modelar el contexto de un sistema y los requisitos de sistema. Diagrama de casos de uso se utiliza para especificar lo que el sistema debe hacer pero no cómo el sistema debe hacer.

Diagrama de clases		Un diagrama de clases muestra un conjunto de clases, interfaces, asociaciones y generalizaciones. Paquete se utiliza comúnmente elemento del modelo para la organización de los elementos en el diagrama de clase. Los diagramas de clases no son sólo para visualizar y documentar los modelos de estructura, sino también para la construcción del sistema ejecutable con avance, retroceso y la ingeniería de ida y vuelta . Hay también un motor de sincronización para generar y actualizar la relación diagrama entidad a partir del diagrama de clase.
Diagrama de estado		Diagrama de máquina de estados representa una máquina de estados. Diagrama de máquina de estados muestra el flujo de control de estado a estado dentro del objeto único. Diagrama de estado común contiene los estados simples, los Estados compuestos, compuestos estados, transiciones, eventos y acciones.
Diagrama de secuencia		Diagrama de secuencia muestra la interacción entre usuarios, sistemas y subsistemas, y hace hincapié en la ordenación del tiempo de los mensajes. Usted puede dibujar diagramas de secuencia exclusivamente con el ratón o con atajos de teclado.
Diagrama entidad relación		Diagrama de entidad-relación (ERD) se utiliza para presentar el Modelo de Entidad-Relación en la industria de software de ingeniería. ERD se puede utilizar para visualizar la estructura conceptual de datos y el esquema de base de datos física. Visual Paradigm for UML apoya conceptual, modelado de datos lógicos y físicos.

Fuente: http://mygnet.net/manuales/uml//manual_de_uml.127

Estructura estática de los diagramas

Clase: Representa un conjunto de objetos que comparten los mismos atributos, operaciones, relaciones y semántica.

Objeto: Se caracteriza por tener una identidad única, un estado definido por un conjunto de valores de atributos y un comportamiento representado por sus operaciones y métodos.

Asociaciones.- Las asociaciones entre dos clases se representan mediante una línea

que las une. La línea puede tener una serie de elementos gráficos que expresan características particulares de la asociación.

El análisis textual

Podemos ver que las palabras que pueden asignar al análisis textual son las siguientes:

Actor: Es un usuario del sistema, que necesita o usa alguno o algunos de los casos de uso. Un usuario puede jugar más de un rol.

Un caso de uso puede tener varios actores. Los actores no necesitan ser humanos pueden ser sistemas externos que necesitan alguna información del sistema actual.

Clase: Dentro de este diagrama las clases son los procedimientos a realizar, sea este uno o más actores.

Diagrama de casos de uso

Podemos ver que las palabras que pueden asignar al diagrama de casos de uso son las siguientes:

Caso de Uso: Conjunto de secuencia de acciones que se ejecutan y el resultado es de interés para un actor en particular.

Inclusión / <<include>>.- Un caso base de include incorpora explícitamente el comportamiento de otro caso de uso en lugar especificado en el caso base, se utiliza para evitar describir el mismo flujo de eventos repetidas, poniendo comportamiento común en un caso de uso aparte.

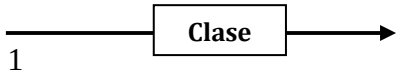
Extensión / <<extend>>.- Significa que un caso de uso incorpora implícitamente el comportamiento de otro caso de uso en el lugar especificado indirectamente por el caso de uso que extiende al base, se usa está relación cuando se tiene un caso de uso que es similar a otro, pero hace un poco más.

Diagrama de clase

Podemos ver que las palabras que pueden asignar al diagrama de clase son las siguientes:

Multiplicidad

La multiplicidad es una restricción que se pone a una asociación, que limita el número de instancias de una clase que pueden tener esa asociación con una instancia de la otra clase. Puede expresarse de las siguientes formas:



1 Exactamente uno

*	Clase	Cero a más
0….1	Clase	Cero a uno
m..n	Clase	Especificada numéricamente

Roles

Para indicar el papel que juega una clase en una asociación se puede especificar un nombre de rol, se representa en el extremo de la asociación junto a la clase que desempeña dicho rol.

Agregación

El símbolo de agregación es un diamante colocado en el extremo en el que está la clase que representa el “todo”. 

Herencia

La relación de herencia se representa mediante un triángulo en el extremo de la relación que corresponde a la clase más general o clase “padre”. 

Si se tiene una relación de herencia con varias clases subordinadas, pero en un diagrama concreto no se quieren poner todas, esto se representa mediante puntos suspensivos.

Diagrama de estado

Podemos ver que las palabras que pueden asignar al diagrama de estado son las siguientes:

Inicio: El inicio de un diagrama de estado es representado por un círculo de color negro sólido.

Actividad: Una actividad representa la acción que será realizada por el sistema la cual es representada dentro de un ovalo.

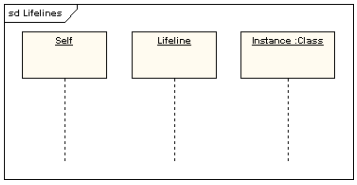
Transición: Una transición ocurre cuando se lleva a cabo el cambio de una actividad a otra, la transición es representada simplemente por una línea con una flecha en su terminación para indicar dirección.

Fin: El fin de un diagrama de estado es representado por la diana.

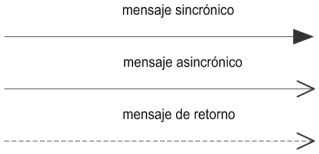
Diagrama de secuencia

Podemos ver que las palabras que pueden asignar al diagrama de secuencia son las siguientes:

Línea de Vida.- Una línea de vida usualmente tiene un rectángulo que contiene el nombre del objeto. Si el nombre es self entonces eso indica que la línea de vida representa el clasificador que posee el diagrama de secuencia.



Mensajes.- Los envíos de mensajes se representan mediante flechas horizontales que unen la línea de vida del objeto emisor con la línea de vida del objeto destinatario.



El mensaje sincrónico es el utilizado con mayor frecuencia. Su uso significa que el expedidor del mensaje espera que la activación del método mencionado por el destinatario finalice antes de continuar su actividad.

En los mensajes asincrónicos, el expedidor no espera el término de la activación invocada por el destinatario, los objetos pueden enviarse mensajes a sí mismos.

Diagrama entidad relación

Podemos ver que las palabras que pueden asignar al diagrama entidad relación son las siguientes:

Dominio.- conjunto de valores permitidos para un atributo. Ej: indicando el tipo de dato por extensión, sexo: M o F

Entidad.- Se trata de un objeto del que se recoge información de interés de cara a la base de datos. Gráficamente se representan mediante un rectángulo. Un ejemplo seria la entidad banco, donde se recogerían los datos relativos a ese banco, como puede ser el nombre, el número de sucursal, la dirección, etc.
Dentro de las entidades pueden ser fuertes o débiles. Las fuertes son las que no dependen de otras entidades para existir, mientras que las entidades débiles siempre dependen de otra entidad sino no tienen sentido por ellas mismas.
Relación.- Podemos definir la relación como una asociación de dos o más entidades. A cada relación se le asigna un nombre para poder distinguirla de las demás y saber su función dentro del modelo entidad-relación.

Las relaciones se representan gráficamente con rombos, dentro de ellas se coloca el

nombre de la relación.

Para finalizar las características de la relación tenemos la cardinalidad que define el número máximo y mínimo de ocurrencias de cada tipo de entidad, puede ser:

1:1. Uno a uno, a cada ocurrencia de una entidad le corresponde como máximo una ocurrencia de la otra entidad relacionada.

1:N. Uno a Mucho, a cada ocurrencia de la entidad A le pueden corresponder varias de la entidad B.

N:M. Muchos a muchos, cada ocurrencia de una entidad puede contener varias de la otra entidad relacionada y viceversa.

Claves.- Las claves se usan para acceder a las tablas. Las claves primarias identifican de manera única a una fila de una tabla, mientras que las claves ajenas son un acceso a datos en otras tablas relacionadas.

Las claves se representan como una restricción y como un valor marcado sobre una columna.

En un diagrama, las marcas PK representan **las claves primarias**, y la operación estereotipada PK es la restricción de la clave primaria.

Una clave ajena representa una columna, la cual es una parte de una relación con otra tabla.

Una marca FK representa la clave ajena. Esto genera la restricción de clave ajena, la cual se representa por un estereotipo FK sobre una operación.

6.1.1.2 Sistemas Indicadores de Gestión y BSC

Concepto del Balanced Scorecard

El **Balanced Scorecard** es un modelo de gestión que traduce la estrategia en objetivos relacionados, medidos a través de indicadores y ligados a unos planes de acción que permiten alinear el comportamiento de los miembros de la organización.

A través de un sistema coherente de elementos –como los mapas estratégicos- el Cuadro de Mando Integral ayuda a ensamblar piezas normalmente descoordinadas en nuestras organizaciones, para adecuar el comportamiento de las personas a la estrategia empresarial.

Podríamos decir que el Balanced Scorecard nos proporciona una "fotografía" que nos permite examinar como estamos llevando a cabo hoy nuestra estrategia al mediano y largo plazo. Para enfocar esta "fotografía" previamente, es necesario concretar la visión del negocio en objetivos estratégicos relacionados entre si según diferentes perspectivas.

Con este ejercicio se consigue hacer que la estrategia sea entendible y, por lo tanto fácilmente comunicable. Ese esfuerzo también nos permite organizar todos los elementos de gestión de la empresa entorno a sus verdaderos objetivos.

Elementos del Balanced Scorecard

Los elementos que componen esta metodología son presentados a continuación:

1. Misión, visión y valores
2. Perspectivas, mapas estratégicos y objetivos
3. Propuesta de valor al cliente
4. Indicadores y sus metas
5. Iniciativas estratégicas
6. Responsables y recursos
7. Evaluación subjetiva.

Etapas del Balanced Scorecard

Las etapas del Balanced Scorecard son las siguientes:

1. Enfoque estratégico
2. Traslado al Balanced Scorecard
3. Sincronización y Despliegue
4. Gestión por procesos
5. Implementación de un sistema de Gestión de Indicadores
6. Aprendizaje Estratégico y Toma de Decisiones

Todas estas etapas serán desarrolladas en los siguientes capítulos.

Enfoque estratégico

Definición de Estrategia

La estrategia debe entenderse como un cuerpo de fenómenos objetivos recurrentes que surgen del conflicto humano. La mayoría de las definiciones de estrategia son exclusivamente normativas, como si se asumiera que ese fenómeno objetivo no existiera o que es tan obvio que no vale la pena definirlo.

Planeación estratégica

Toda empresa diseña planes estratégicos para el logro de sus objetivos y metas planteadas, esto planes pueden ser a corto, mediano y largo plazo, según la amplitud y magnitud de la empresa, es decir, su tamaño, ya que esto implica que cantidad de planes y actividades debe ejecutar cada unidad operativa, ya sea de niveles superiores o niveles inferiores.

Ha de destacarse que el presupuesto refleja el resultado obtenido de la aplicación de los planes estratégicos, es de considerarse que es fundamental conocer y ejecutar correctamente los objetivos para poder lograr las metas trazadas por las empresas.

También es importante señalar que la empresa debe precisar con exactitud y cuidado la misión que va regir la empresa, siendo esta fundamental, ya que representa las funciones operativas que va ha ejecutar en el mercado y va ha suministrar a los consumidores.

Definición del negocio

Lo primero que hay que hacer es una definición de la actividad a la que se dedica la empresa, cómo la lleva a cabo y qué la diferencia de las demás.

Aunque esto puede parecer un paso simple, es sorprendente la cantidad de empresas que no saben a que se dedican o que definen su actividad de una manera muy ligera, dando como resultado objetivos difusos que se diluyen manteniéndose en una indefinición constante.

¿Cómo definir cual es su negocio?

Para definir nuestro negocio podemos valernos de 3 escuelas del pensamiento:

1. Según el producto.
2. Según el beneficio hacia el cliente.
3. Según las capacidades de la empresa.

Análisis FODA

El análisis FODA es una herramienta que permite conformar un cuadro de la situación actual de la empresa u organización, permitiendo de esta manera obtener un diagnóstico preciso que permita en función de ello tomar decisiones acordes con los objetivos y políticas formulados.

El término FODA es una sigla conformada por las primeras letras de las palabras Fortalezas, Oportunidades, Debilidades y Amenazas (en inglés SWOT: Strenghts, Weaknesses, Oportunities, Threats). De entre estas cuatro variables, tanto fortalezas como debilidades son internas de la organización, por lo que es posible actuar directamente sobre ellas. En cambio las oportunidades y las amenazas son externas, por lo que en general resulta muy difícil poder modificarlas.

Cadena de valor

Cada empresa es un conjunto de actividades que se desempeñan para diseñar, producir, llevar al mercado, entregar y apoyar a sus productos. La cadena de valor y la forma en que desempeñan sus actividades individuales son un reflejo de su historia, de su estrategia, de su enfoque y las economías fundamentales para las actividades mismas.

Stakeholders

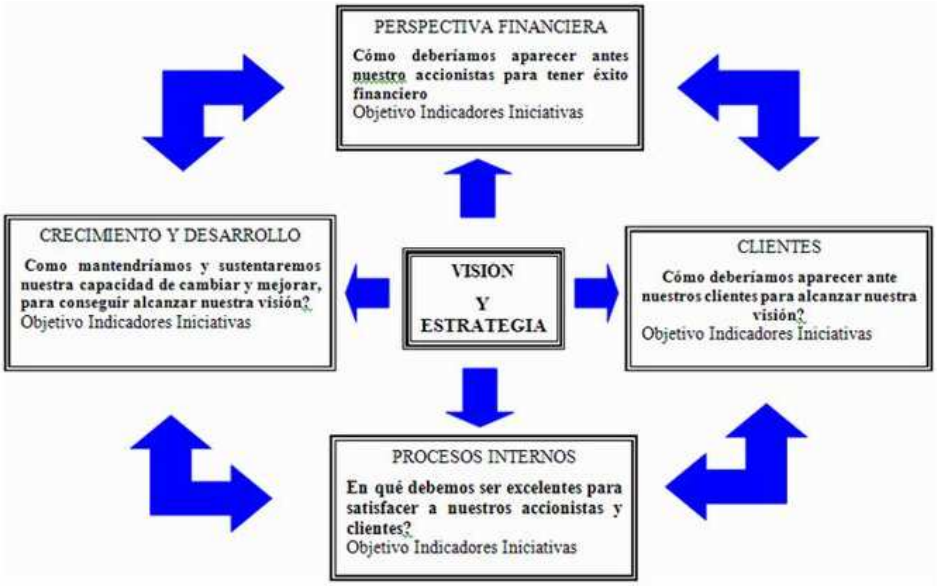
Son aquellos grupos de individuos, entes e instituciones cuyos objetivos y logros dependen de lo que haga la organización, y de los que a su vez dependen los de la organización.

Principales Stakeholders

- ✓ Accionistas
- ✓ Clientes
- ✓ Proveedores
- ✓ Empleados
- ✓ Gobierno

Gráfico #08

Esquema de las 4 acciones



Fuente: http://www.monografias.com/trabajos70/diseno-sistema-indicadores-empresa.html

Elaborado: Kaplan y Norton 2000

¿Qué es una propuesta de valor?

La propuesta de valor se puede considerar como la infraestructura necesaria que le dará vida a la estrategia. Una propuesta de valor depende de tres dimensiones:

- **Los atributos de los productos y/o servicios:** Está relacionada con la funcionalidad, oportunidad, la calidad y el precio.
- **La relación con los clientes:** Incluye la respuesta a los clientes, plazos de entrega y sensación del cliente, así como la experiencia de compra.
- **Imagen y prestigio:** Refleja los factores intangibles que atraen a un cliente hacia una empresa, permitiendo definirse así misma de manera proactiva para sus clientes.

Misión y visión organizacional

La Misión de una organización es una frase concisa que da la razón de la existencia de esta, su propósito básico, hacia donde apuntan sus actividades, y los valores que guían las actividades de sus empleados.

La misión está vinculada con los ***valores centrales***. También describen cómo competir y generar valor al cliente.

La Visión de una organización es una frase concisa que describe las metas de mediano y largo plazo. La Visión es "externa", orientada al mercado, y debería expresar de una manera colorida y visionaria cómo quiere la organización ser percibida por el mundo. Las principales diferencias entre Misión y Visión son:

• La Misión mira hacia "adentro" de la organización, mientras que la Visión lo hace hacia "afuera".

• La Misión se orienta a largo plazo, en el sentido de rescatar la identidad, los aspectos inmutables esenciales, mientras que la Visión lo hace en el mediano a largo plazo, acentuando los aspectos que deben cambiar.

Temas estratégicos

Representa los componentes claves que formaran la estrategia empresarial.

Son líneas básicas de desarrollo de la organización. Estos se basan en el resumen del análisis FODA, necesidades de los stakeholders, propuesta de valor y la misión y visión.

Valores organizacionales

Es la manera de ser de la organización. Ejemplo de valores

- Calidad
- Defensa del ambiente
- Equidad
- Honestidad
- Respeto
- Creatividad

Traslado al Balanced Scorecard

Perspectiva estratégica

Kaplan y Norton no definen explícitamente lo que significa una perspectiva, pero enumeran las cuatro perspectivas principales que una organización (con o sin fines de lucro) debe tener:

- **Perspectiva financiera:** En organizaciones con fines de lucro, esto implica a accionistas, mientras que en organizaciones sin fines de lucro, esto implica a las organizaciones financieras o subsidiarias.
- **Perspectiva del cliente**
- **Perspectiva del proceso (de negocio) interno**: Implica:
  - ✓ El proceso de gestión de las operaciones

- ✓ El proceso de gestión de clientes
- ✓ El proceso de innovación
- ✓ El proceso social y regulatorio

❖ **Perspectiva del aprendizaje y crecimiento:** Esto implica el desarrollo del capital humano, de la información y organización. Algunos datos importantes sobre las perspectivas y su orden:

Las perspectivas se ordenan de forma descendente por su:

Conmensurabilidad: Capacidad de ser medida, característica que vuelve a la perspectiva más objetiva. En la medida que la perspectiva no pueda medirse, ésta se vuelve subjetiva, y por este motivo se desplaza hacia abajo en la lista.

Urgencia

Tangencia: cantidad de puntos de contacto de la perspectiva con el rumbo que se quiere dar a la organización

Visibilidad: La misión, visión, valores centrales y metas principales de la organización se expresan en términos de perspectivas más altas.

Las estrategias de detalle están en términos de perspectivas más bajas.

Mapas estratégicos

Tanto el mapa de procesos como el mapa estratégico debería ser una representación gráfica de cómo la empresa espera alcanzar los resultados planificados para el logro de su estrategia o política de calidad. Algunos de los mapas de procesos analizados, en muchos casos, reflejan una descripción de los requisitos de la normativa en lugar de presentar como la empresa ha planificado alcanzar los resultados. En otros casos es una descripción de la interacción de procesos, parecido a una distribución de planta, otros asemejan organigramas funcionales

Objetivos estratégicos

Los objetivos estratégicos son resultados específicos de mediano plazo (más de un año) que la organización busca lograr a través del esfuerzo intencionado, y en el contexto de los desafíos que plantea la visión.

Son un enlace que le da coherencia a los retos organizacionales.

Matriz Balanced Scorecard y las 3'M

Es una tabla que presenta los diversos componentes del mapa estratégico:

- ✓ Meta: que tenemos que hacer para alcanzarla
- ✓ Medida: como me aseguro que logré la meta
- ✓ Medio: que mecanismo debo usar para lograr el objetivo

¿Que son los indicadores o KPI's?

Los Indicadores clave de rendimiento provienen del inglés Key Performance Indicators (KPI), estos miden el nivel de desempeño de unproceso, enfocándose en el "cómo" e indicando la evaluación de los procesos, de forma que se pueda alcanzar el objetivo fijado.

En otras palabras, Key Performance Indicators (KPI) son métricas financieras o no financieras utilizadas para cuantificar objetivos para reflejar el rendimiento de una organización. El acto de monitorizar los Indicadores Clave de Rendimiento en tiempo real se conoce como Monitorización de Actividad de Negocio. Los Indicadores de Rendimiento son frecuentemente utilizados para "valorar" actividades complicadas de medir, tales como los beneficios de desarrollos líderes, compromiso de empleados, servicio o satisfacción.

Los KPIs suelen estar atados a la estrategia de la organización (ejemplificadas en las técnicas como la del Balanced Scorecard). Los KPIs son "vehículos de comunicación". Permiten que los ejecutivos de alto nivel comuniquen la misión y visión de la empresa a los niveles jerárquicos más bajos, involucrando directamente a todos los colaboradores en realización de los objetivos estratégicos de la empresa.

- ✓ Tiempo que se utiliza en mejorar los niveles de servicio en un proyecto dado.
- ✓ Nivel de la satisfacción del cliente.
- ✓ Tiempo de mejoras de asuntos relacionados con los niveles de servicio.

- ✓ Impacto de la calidad de los recursos financieros adicionales necesarios para realizar el nivel de servicio definido.
- ✓ Para una organización es necesario al menos que pueda identificar sus propios KPI's. La clave para esto es:
- ✓ Tener predefinido de antemano un proceso de negocio.
- ✓ Tener claros los objetivos requeridos en el proceso de negocio.
- ✓ Tener una medida cuantitativa/cualitativa de los resultados y que sea posible su comparación con los objetivos.

Sincronización y Despliegue

Esta etapa consiste en alinear el Balance Scoredcard de la organización con los diferentes procesos de la misma. El **alineamiento horizontal** consiste en sincronizar todas las áreas y procesos de la organización. El **alineamiento vertical** consiste en acoplar todas las funciones y/o niveles de la organización.

Lo importante de esta etapa es de innovar una organización que se encuentre basada en un enfoque funcional, ha que se base en un enfoque por procesos. De esta manera sincronizar todos los esfuerzos de los diferentes departamentos y cumplir la misión, visión y objetivos estratégicos trazados.

Para la ejecución de esta etapa es importante determinar los macroprocesos que posee la compañía, los mismos que deben clasificarse en procesos estratégicos, procesos de apoyo, procesos operativos y procesos de medición, análisis y mejora. Luego de determinado estos macroprocesos, se determina la secuencialidad de los procesos a través de la matriz SIPOC, donde se relaciona los departamentos con los procesos que llevan a cabo. A continuación se realiza la sincronización con el BSC, tomando en cuenta los principios de: enfoque, consistencia, sincronización, balance, los mismos que se generan de la matriz de contribución crítica. Se eligen aquellos procesos que se encuentren mayormente relacionados con los objetivos estratégicos para aplicar el alineamiento horizontal con el Balance Scorecard.

Esta sincronización se traduce a establecer la consistencia de cada unos de los KPI´s con los procesos de la organización, así como también en el ajuste de metas. Luego

se determina los mapas y BSC por cada área funcional de las distintas Unidades de Negocio.

La siguiente subetapa es trasladarse de la sincronización horizontal al despliegue vertical de la organización, es decir trasladar a cada uno de los niveles el horizonte que ha seguir para la consecución de lo antes mencionado. Este despliegue vertical consiste en buscar la participación de todos los miembros de la organización, involucrando desde el estrato superior (BSC de la empresa) hasta el estrato inferior (BSC individual). Este despliegue no significa una comunicación hacia abajo en todos los niveles, es buscar la adecuada comunicación entre cada uno de ellos, buscar el trabajo en equipo y compromiso de todos los miembros. Así como en el alineamiento horizontal, en esta subetapa se efectúa la matriz de contribución crítica, para enlazar cada uno de los componentes de la metodología.

Gestión por Procesos

La etapa de ENFOQUE O GESTION POR PROCESOS, consiste en analizar la secuencialidad de actividades que se llevan a cabo dentro de las organizaciones, para de esta manera conseguir los resultados deseados.

Los pasos ha seguir para enfocar a las organizaciones por una metodología basada en procesos es determinar los macroprocesos que poseen y la relación que llevan entre ellos. Luego se hace un despliegue de estos macroprocesos mostrando los procesos y actividades que los engloban. Lo importante en este tema, es determinar la mejor manera de realizar estos procesos, su cumplimiento en cada paso, y el monitoreo de dicho procesos, para buscar la mejora continua.

Para describir un proceso se puede aplicar la diagramación o también llamada representación gráfica, la misma que dentro de la metodología que vamos a desarrollar es la elaboración de matrices SIPOC, donde se muestra la interacción entre cada uno de los procesos con la secuencialidad de actividades que poseen, luego se realizan diagramas de flujo para extender la secuencia interna de los mismos. A continuación se define por cada proceso los indicadores que nos van a

permitir dar el seguimiento adecuado para la evaluación de eficiencia, tiempo de ejecución, calidad, productividad, impactos internos y externos, y la cultura de capacitación y satisfacción de estos.

Implementación e Implantación de un Sistema de Gestión de Indicadores,

La Creación y Gestión de Indicadores es la base fundamental para crear un sistema de mejora de cada una de las actividades dentro de la Empresa.

La importancia de este tema va más allá de la simple creación de indicadores de forma empírica y su posterior gestión. Como hemos venido analizando, se debe seguir un procedimiento extenso para la determinación de los indicadores precisos y su correcta forma de medición.

Por otro lado, estos indicadores obtenidos deben ser monitoreados periódicamente, y deben tomarse decisiones concretas en base a los resultados de dichas mediciones. Veamos algunas definiciones relacionadas con los indicadores.

Indicadores como la base para la medición

Los objetivos y tareas que se propone alcanzar una organización deben expresarse en términos medibles, que permitan evaluar el grado de cumplimiento o avance de los mismos. Es aquí que el uso de los indicadores tiene su mayor fortaleza.

El término "Indicador" se refiere a datos esencialmente cuantitativos, que nos permiten percibir cómo se encuentran las cosas en relación con algún aspecto de la realidad que nos interesa conocer. Los Indicadores pueden ser medidas, números, hechos, opiniones o percepciones que señalen condiciones o situaciones específicas.

Tipos de indicadores

- ***Indicadores Cuantitativos***: Son los que se refieren a medidas en números o cantidades. Se los usa para medir la efectividad de los OBJETIVOS ESTRATEGICOS.

- ***Indicadores Cualitativos***: Son los que se refieren a cualidades. Se trata de aspectos que no son cuantificados directamente, tales como opiniones,

percepciones o juicios. Se los usa para analizar algunos aspectos de las INICIATIVAS ESTRATEGICAS.

- ***Indicadores Positivos*:** Son aquellos en los cuales un AUMENTO en su valor o tendencia, estarían indicando un avance hacia la situación deseada. Por ejemplo: Nivel de ventas, participación de mercado, % de competencias del recurso humano, etc.

- ***Indicador Negativo*:** Son aquellos en los cuales una DISMINUCIÓN de su valor o tendencia, estarían indicando un avance hacia la situación deseada. Por ejemplo: Nivel de reclamos, nivel de costos, números de errores cometidos, etc.

- ***Indicador Centrado*:** Son aquellos en los cuales se espera que se mantenga CENTRADO alrededor de un valor meta para mantener una situación deseada. Por ejemplo: El cumplimiento de un estándar requerido como longitud, grosor, etc.

Definiciones Iniciales para un Sistema de Gestión

- **Nivel Base:** Se refiere a la medición inicial o nivel estándar que toma el indicador, y representa el desempeño logrado antes del efecto de mejora de las iniciativas estratégicas.

- **Valor actual:** Representa las mediciones período a período del indicador, las cuales se ven afectadas por los efectos de las iniciativas estratégica.

- **Meta:** Es el nivel esperado del indicador que la organización desea lograr luego de ejecutar exitosamente las iniciativas de mejora (iniciativas estratégicas)

Nivel de Efectividad de un Indicador

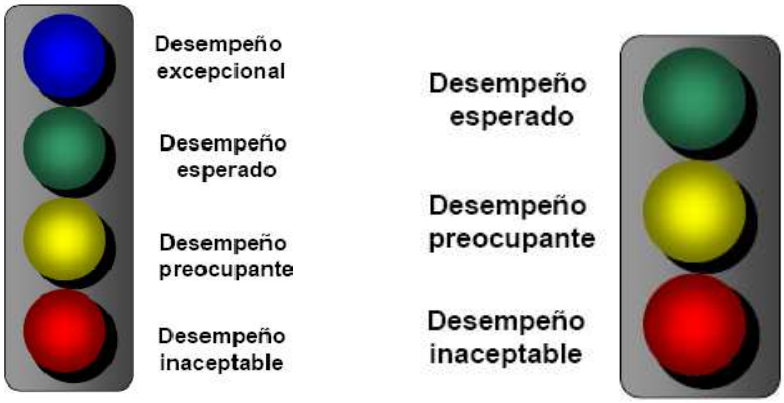
"Efectividad" es la relación entre los resultados logrados y las metas propuestas, es decir, nos permite medir el grado de cumplimiento de los objetivos planificados.

La "Efectividad" por si sola no facilita el proceso de toma de decisiones, pues esta debe ir acompañada con información accionable, que dirijan las acciones de la alta gerencia.

Para lograr esto, es necesario el uso de "semáforos" u "odómetros" que muestren rápidamente que tan Óptimo es el grado de cumplimiento alcanzado en el período. Estos sistemas gráficos de medición son de libre decisión de la Empresa.

En el siguiente gráfico se muestra un ejemplo del esquema comúnmente utilizado para medir los Indicadores.

Gráfico #09
Indicadores gráficos de medición.



Fuente: Seminario de Graduación UPS – Metodología BSC
Elaborado: Babici Bogdan, Achig María, Tierra Doris.

La gerencia general debe definir claramente los límites de actuación de los semáforos, a fin de garantizar un pleno conocimiento de sus expectativas al resto de usuarios del sistema.

Herramientas para el Análisis de Datos

-Análisis de series de tiempo

Con frecuencia se realizan observaciones de datos a través del tiempo. Cualquier variable que conste de datos reunidos, registrados u observados sobre incrementos sucesivos de tiempo se denomina serie de tiempo.

Una serie de tiempo es un conjunto de observaciones producidas en determinados momentos durante un periodo, semanal, mensual, trimestral o anual, generalmente a intervalos iguales.

Si bien el comportamiento de cualquier serie de tiempo puede observarse gráficamente, no en todos los casos es posible distinguir las particularidades que cada una puede contener. La experiencia basada en muchos ejemplos de series de tiempo, ha revelado que existen ciertos movimientos o variaciones características que pueden medirse y observarse por separado.

Estos movimientos, llamados a menudo componentes, de una serie de tiempo y que se supone son causados por fenómenos distintos. El primer paso para analizar una serie de tiempo es graficarla, esto permite: identificar la tendencia, la estacionalidad, las variaciones irregulares (componente aleatoria). Un modelo clásico para una serie de tiempo, puede ser expresada como suma o producto de tres componentes: tendencia, estacional y un término de error aleatorio.

Son innumerables las aplicaciones que se pueden citar, en distintas áreas del conocimiento, tales como, en economía, física, geofísica, química, electricidad, en demografía, en marketing, en telecomunicaciones, en transporte, etc.

Sistema de Control de Gestión

El sistema de control de gestión está destinado a ayudar a los distintos niveles de decisión a coordinar las acciones, a fin de alcanzar los objetivos de mantenimiento,

desempeño y evolución, fijados a distintos plazos, especificando que si los datos contables siguen siendo importantes, está lejos de tener el carácter casi exclusivo que se le concede en muchos sistemas de control de gestión.

El Control de Gestión es: "el conjunto de mecanismos que puede utilizar la dirección que permiten aumentar la probabilidad de que el comportamiento de las personas que forman parte de la Organización sea coherente con los objetivos de ésta". [1]

Este concepto propone una nueva dimensión del control de gestión, pues no solo se centran en el carácter contable y a corto plazo de éste, sino que reconocen la existencia de otros factores e indicadores no financieros que influyen en el proceso de creación de valor, ya sea en productos o servicios, y se enfocan sobre la base de la existencia de objetivos propuestos a alcanzar.

Se le incorpora un balance periódico de las Debilidades y Fortalezas, un análisis comparativo e inter-organizaciones, el uso del Cuadro de Mando como mecanismo de control y flujo de información.

Otra filosofía de perfeccionamiento del sistema de gestión está destinada a poner de manifiesto las interrelaciones entre los procesos humanos y el sistema de control, utilizando para ello, factores no formales del control, los cuales han cobrado gran importancia en los últimos años.

[1] Amat Joan Ma. (1992),

6.2.1 CAPITULO 2

6.2.2 ANALISIS Y DISEÑO DE SISTEMAS ORIENTADO A OBJETOS

DIAGRAMAS UML

Es un conjunto de herramientas, que permite modelar (analizar y diseñar) sistemas orientados a objetos.

Los diagramas UML contienen:

Diagrama de casos de uso

No son parte del diseño (cómo), sino parte del análisis (qué). De forma que al ser parte del análisis nos ayudan a describir qué es lo que es sistema debe hacer. Los Casos de Uso son qué hace el sistema desde el punto de vista del usuario. Es decir, describen un uso del sistema y cómo este interactúa con el usuario.

Diagrama de clases

Describen las relaciones y las dependencias entre un grupo de *casos de uso* y los actores participantes en el proceso.

Es importante resaltar que los diagramas de casos de uso no están pensados para representar el diseño y no puede describir los elementos internos de un sistema. Los diagramas de casos de uso sirven para facilitar la comunicación con los futuros usuarios del sistema, y con el cliente, y resultan especialmente útiles para determinar las características necesarias que tendrá el sistema. En otras palabras, los diagramas de casos de uso describen *qué* es lo que debe hacer el sistema, pero no *cómo*.

Diagrama de estados

Los diagramas de estado muestran los diferentes estados de un objeto durante su vida, y los estímulos que provocan los cambios de estado en un objeto.

Los diagramas de estado ven a los objetos como *máquinas de estado* o autómatas finitos que pueden estar en un conjunto de estados finitos y que pueden cambiar su

estado a través de un estímulo perteneciente a un conjunto finito. Por ejemplo, un objeto de tipo *NetServer* puede tener durante su vida uno de los siguientes estados:

- Listo
- Escuchando
- Trabajando
- Detenido

Y los eventos que pueden producir que el objeto cambie de estado son:

- Se crea el objeto
- El objeto recibe un mensaje de escucha
- Un cliente solicita una conexión a través de la red
- Un cliente finaliza una solicitud
- La solicitud se ejecuta y ser termina
- El objeto recibe un mensaje de detención
- etc

Diagrama de secuencias

Los diagramas de secuencia muestran el intercambio de mensajes (es decir la forma en que se invocan) en un momento dado. Los diagramas de secuencia ponen especial énfasis en el orden y el momento en que se envían los mensajes a los objetos.

En los diagramas de secuencia, los objetos están representados por líneas intermitentes verticales, con el nombre del objeto en la parte más alta. El eje de tiempo también es vertical, incrementándose hacia abajo, de forma que los mensajes son enviados de un objeto a otro en forma de flechas con los nombres de la operación y los parámetros.

Diagrama de actividades

Los diagramas de actividad describen la secuencia de las actividades en un sistema. Los diagramas de actividad son una forma especial de los diagramas de estado, que únicamente (o mayormente) contienen actividades.

Diagramas de relaciones de entidad

Diagramas E/R muestran el diseño conceptual de las aplicaciones de bases de datos. Representan varias entidades (conceptos) en el sistema de información y las relaciones y restricciones existentes entre ellas. Una extensión de los diagramas de relaciones de entidad llamado «diagramas de relaciones de entidad extendida» o «diagramas de relaciones de entidad mejoradas» (EER), se utiliza para incorporar las técnicas de diseño orientadas a objetos en los diagramas ER.

Entidad

Una **entidad** es cualquier concepto del mundo real con una existencia independiente. Puede ser un objeto con una existencia física (ejemplo, máquina, robot) o puede ser un objeto con una existencia conceptual (p. ej.: Curso de universidad). Cada entidad tiene un conjunto de atributos que describen las propiedades de la entidad.

En un diagrama ER, las entidades se representan como rectángulos, con el nombre de la clase, y también pueden mostrar atributos y operaciones de la clase en otros dos «compartimentos» dentro del rectángulo.

BASE DE DATOS

Almacén de datos relacionados con diferentes modos de organización. Una base de datos representa algunos aspectos del mundo real, aquellos que le interesan al diseñador. Se diseña y almacena datos con un propósito específico.

Los "datos" se hace referencia a hechos conocidos que pueden registrarse, como ser números telefónicos, direcciones, nombres, etc. Las bases de datos almacenan datos que luego pueden ser visualizados fácilmente y mostrarlos de diversas formas.

El proceso de construir una base de datos es llamado diseño de base de datos.

MÁQUINA VIRTUAL

Es un software que emula a una computadora y puede ejecutar programas como si fuese una computadora real. Este software en un principio fue definido como "un

duplicado eficiente y aislado de una máquina física". La acepción del término actualmente incluye a máquinas virtuales que no tienen ninguna equivalencia directa con ningún hardware real.

Una característica esencial de las máquinas virtuales es que los procesos que ejecutan están limitados por los recursos y abstracciones proporcionados por ellas. Estos procesos no pueden escaparse de esta "computadora virtual".

VMware Workstation es un producto de software de la empresa VMware Inc., que consiste de una máquina virtual para computadoras x86 y x86-64.

Este software permite a los usuarios armar múltiples computadoras virtuales x86 y x86-64 y usar una o más de esas computadoras virtuales simultáneamente con el sistema operativo anfitrión. Cada instancia de máquina virtual puede ejecutar su propio sistema operativo huésped como Windows, Linux, etc.

En términos sencillos, VMware Workstation permite a una máquina física ejecutar dos o más sistemas operativos simultáneamente. Otros productos de VMWare pueden ayudar a gestionar máquinas virtuales VMware a lo largo de múltiples máquinas anfitrionas (hosts).

VMware Workstation además provee la habilidad de simular algunos dispositivos de hardware. Por ejemplo, se puede montar un archivo ISO como un CD-ROM, o archivos .vmdk como discos duros.

También se pueden tomar "instantáneas" de un sistema operativo, creando "puntos de restauración", permitiendo así volver al estado exacto en que se encontraba el sistema en ese momento cuando se desee.

6.3.1CAPITULO 3

6.3.1.1 INTRODUCCION A LA ADMINISTRACION DE BASE DE DATOS ORACLE 10G Y AL APLICATION SERVER 10G.

Oracle

Oracle es un sistema de gestión de base de datos relacional, fabricado por Oracle Corporation. Se considera a Oracle como uno de los sistemas de bases de datos más completos, destacando su:

- ✓ Soporte de transacciones.
- ✓ Estabilidad.
- ✓ Escalabilidad.
- ✓ Es multiplataforma.

Oracle es básicamente un herramienta cliente/servidor para la gestión de base de datos la gran potencia que tiene y su elevado precio hace que solo se vea en empresas muy grandes y multinacionales, por norma general.

Características de Oracle

Desarrollado sobre Oracle Database, Oracle Content Database ha sido diseñada para que las organizaciones puedan controlar y gestionar grandes volúmenes de contenidos no estructurados en un único repositorio con el objetivo de reducir los costes y los riesgos asociados a la pérdida de información.

La Base de Datos

La base de datos de Oracle tiene una capa lógica y otra física. La capa física consiste

de archivos que residen en el disco y los componentes de la capa lógica son estructuras que mapean los datos hacia estos componentes físicos.

LA CAPA FÍSICA

Consiste de archivos físicos que se encuentran en los discos. Estos pueden ser de tres tipos diferentes:

Uno o más datafiles

Los datafiles almacenan toda la información ingresada en una base de datos. Se pueden tener sólo uno o cientos de ellos. Muchos objetos (tablas, índices) pueden compartir varios datafiles. El número máximo de datafiles que pueden ser configurados está limitado por el parámetro de sistema MAXDATAFILES.

Dos o más archivos redo log (de deshacer)

Los archivos del tipo redo log almacenan información que se utiliza para la recuperación de una base de datos en caso de falla. Estos archivos almacenan la historia de cambios efectuados sobre la base de datos y son particularmente útiles cuando se necesita corroborar si los cambios que la base de datos ya ha confirmado se han efectuado realmente en los datafiles.

Uno o más control files

Estos archivos contienen información que se utiliza cuando se levanta una instancia, tal como la información de dónde se encuentran ubicados los datafiles y los archivos redo log. Estos archivos de control deben encontrarse siempre protegidos.

LA CAPA LÓGICA

La capa lógica de una base de datos consta de los siguientes elementos:

Uno o más tablespaces

El esquema de la base de datos (schema), el cual consiste de objetos como tablas, clusters, índices, vistas, procedimientos almacenados, triggers, secuencias y otros.

ORACLE 10g

Oracle Database 10g está diseñada para poder desplegarse con eficacia tanto en los pequeños servidores blade como en los servidores más grandes SMP y clusters de todos los tamaños. Ofrece la posibilidad de gestión automatizada para una operación fácil y de bajo coste. Con su capacidad única para gestionar toda clase de datos, desde la información tradicional del negocio hasta documentos XML e informaciones espaciales y de localización, Oracle Database 10g es la elección idónea tanto para el proceso transaccional como para las aplicaciones de soporte a la decisión y de gestión de contenidos.

Oracle 10g provee de la primera infraestructura completa e integrada para potenciar el grid computing. Oracle 10g recoge los atributos fundamentales de esta tecnología de computación:

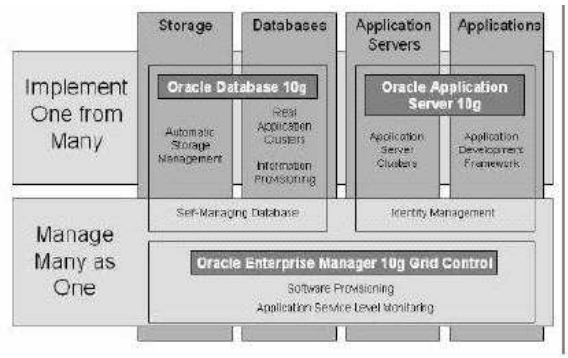
- Implementar uno a partir de muchos (Implement One from Many): Se trata de conseguir que un número indeterminado de máquinas funcionen como una sola. Para

ello, se tiene que aplicar un proceso de virtualización en todas las capas del sistema y una zona común donde se almacenen los recursos (resource pooling).

- Administrar Muchos como si fueran Uno (Manage Many as One): Para conseguirlo debemos contar con un software a medida y de un sistema de administración centralizado.

Los dos puntos anteriores deben aplicarse a cada elemento del grid: sistemas de almacenamiento, bases de datos, servidores de aplicación y aplicaciones.

Gráfico #10
Estructura Oracle 10G



Fuente:www.oracle.com

La apuesta de Oracle por la tecnología Grid Computing se basa en sus tres productos fundamentales:

- ✓ Oracle 10g Database.
- ✓ Oracle 10g Application Server.
- ✓ Oracle Enterprise Manager 10g Grid Control.
- ✓ Oracle Database 10g

Esta construida sobre el éxito de la versión anterior, Oracle 9i, y añade muchas

nuevas características enfocadas al Grid Computing, como por ejemplo una zona de almacenamiento común, Oracle es la primera compañía en ofrecer una base de datos para grid real. Oracle 10g se basa en Real Application Clusters, una opción incluida ya en Oracle 9i, y muchos clientes en producción sirven para asegurar la utilidad y fiabilidad de esta herramienta.

Oracle Enterprise Manager 10g Grid Control

Oracle Enterprise manager 10g Grid Control es la consola central de administración y el entorno que automatiza las tareas administrativas para el conjunto de sistemas implicados en un entorno grid. Esta consola ayuda a reducir los costes de administración; con ella, los DBAs pueden agrupar múltiples nodos hardware como bases de datos, servidores de aplicación, servidores web etc. como si fueran unidades lógicas. Ejecutando trabajos, diseñando políticas, monitorizando el rendimiento y automatizando muchas otras tareas sobre un conjunto de destinos en vez de sobre muchos sistemas individuales, la OEM Grid Control permite escalar un grid fácilmente. Debido a esta característica, la existencia de muchas pequeñas máquinas no incrementa la complejidad de la administración.

Debido al gran número potencial de nodos físicos en un grid, es especialmente importante que la instalación y configuración del software de esas máquinas sea rápida y no requiera intervención humana. Con OEM Grid Control, Oracle 10g automatiza la instalación, configuración y clonación de servidores de aplicación y de bases de datos sobre múltiples nodos. Este entorno puede utilizarse tanto para la adición de nuevos sistemas como para aplicar parches o añadir utilidades a sistemas ya existentes. También mantiene la sincronía entre los nodos.

Estándares en Grid

Con Oracle 10g, las compañías pueden empezar a implementar sus estructuras de Grid Computing, pero el estándar abierto fue concebido primeramente por el GGF (Global Grid Forum). Este organismo ha definido un estándar (Open Grid Services Architecture, OGSA) para asentar los servicios generales y estándares de programación que mejor se adaptan al grid computing.

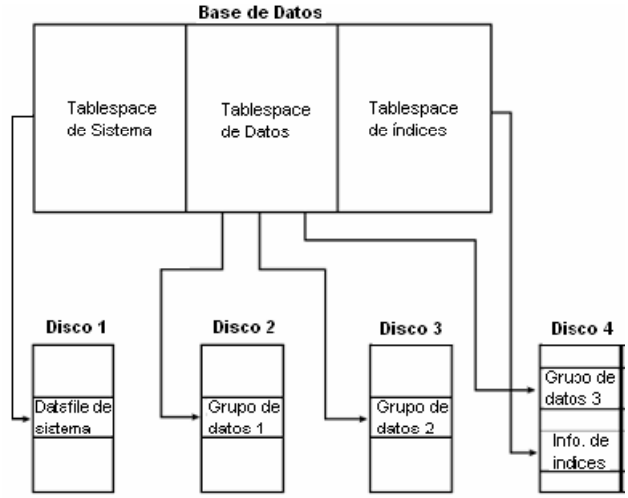
TABLESPACES Y LOS DATAFILES

Como se mencionó, una base de datos se encuentra dividida en una o más piezas lógicas llamadas tablespaces, que son utilizados para separar la información en grupos y así simplificar la administración de los datos. Los tablespaces pueden ocupar uno o más datafiles. Si se decide que utilice varios datafiles, el administrador del sistema puede gestionar que éstos queden localizados en discos diferentes, lo que aumentará el rendimiento del sistema, principalmente por la mejora en la distribución de la carga de entrada / salida.

En la figura #11 se aprecia la diferencia entre estos tres conceptos. Una base de datos de ejemplo contiene tres tablespaces lógicos (parte superior de la figura) que utiliza para almacenar información del sistema, de los datos del usuario y de los índices de las tablas. Asimismo, existen los espacios físicos (datafiles) que guardan esta información en los diferentes discos disponibles y que se señalan en la parte inferior del dibujo.

Gráfico #11

Base de Datos –Tablespace



Fuente: www.oracle.com

SEGMENTOS,

EXTENSIONES Y BLOQUES

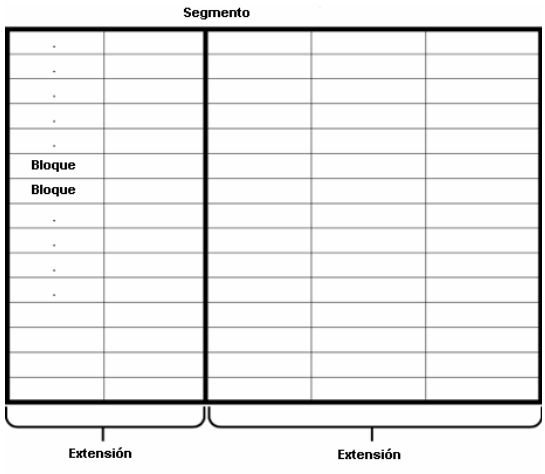
Dentro de los tablespaces y datafiles, el espacio utilizado para almacenar datos es controlado por el uso de ciertas estructuras; éstas son las siguientes:

Bloques: Un bloque es la unidad de almacenamiento más pequeña en una base de datos Oracle. Contiene una pequeña porción de información (header) referente al bloque en sí y el resto a los datos que guarda. Generalmente, un bloque de datos ocupará aprox. 2 KB de espacio físico en el disco (asignación típica).

Extensiones: Es un grupo de bloques de datos. Se establecen en un tamaño fijo y crecen a medida que van almacenando más datos. También se pueden redimensionar para aprovechar mejor el espacio de almacenamiento.

Segmentos: Es un grupo de extensiones utilizados para almacenar un tipo particular de datos. Existen 4 tipos de segmentos: datos, índices, rollback y temporales.

Gráfico # 12
***Relación entre* segmentos, extensiones y bloques**



Fuente: www.oracle.com

EL ESQUEMA DE LA BASE DE DATOS

Un esquema es una colección de objetos lógicos, utilizados para organizar de manera más comprensible la información y conocidos como objetos del esquema. Una breve descripción de los objetos que lo componen es la siguiente:

Tabla: Es la unidad lógica básica de almacenamiento. Contiene filas y columnas (como una matriz) y se identifica por un nombre. Las columnas también tienen un nombre y deben especificar un tipo de datos. Una tabla se guarda dentro de un tablespace (o varios, en el caso de las tablas particionadas).

Cluster: Un cluster es un grupo de tablas almacenadas en conjunto físicamente como una sola tabla que comparte una columna en común. Si a menudo se necesita recuperar datos de dos o más tablas basado en un valor de la columna que tienen en común, entonces es más eficiente organizarlas como un cluster, ya que la información podrá ser recuperada en una menor cantidad de operaciones de lectura realizadas sobre el disco.

Índice: Un índice es una estructura creada para ayudar a recuperar datos de una manera más rápida y eficiente. Un índice se crea sobre una o varias columnas de una misma tabla. De esta manera, cuando se solicita recuperar datos de ella mediante alguna condición de búsqueda (cláusula where de la sentencia), ésta se puede acelerar si se dispone de algún índice sobre las columnas-objetivo.

Vista: Una vista implementa una selección de varias columnas de una o diferentes tablas. Una vista no almacena datos; sólo los presenta en forma dinámica. Se utilizan para simplificar la visión del usuario sobre un conjunto de tablas, haciendo transparente para él la forma de obtención de los datos.

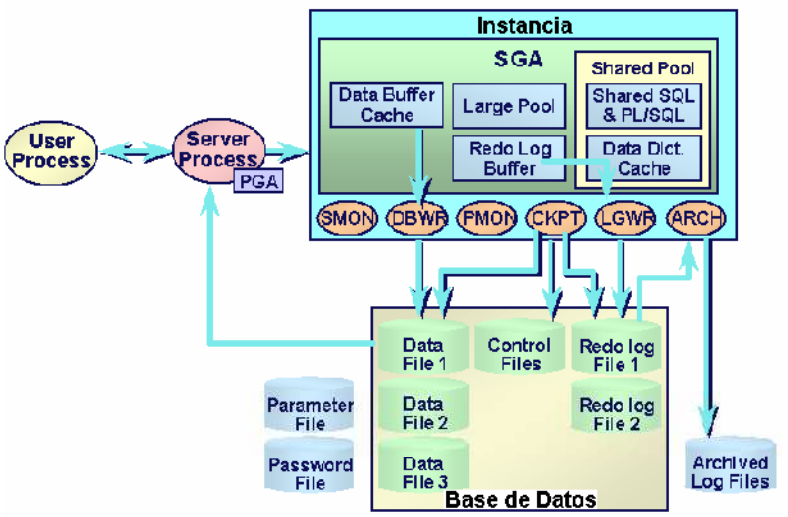
Procedimiento Almacenado: Son programas que permiten independizar el manejo de datos desde una aplicación y efectuarla directamente desde el motor de base de datos, disminuyendo así el tráfico de información a través de la red y mejorando el rendimiento de los procesos implementados mediante estos programas.

Trigger: Un trigger es un procedimiento que se ejecuta en forma inmediata cuando ocurre un evento especial. Estos eventos sólo pueden ser la inserción, actualización o eliminación de datos de una tabla.

Secuencias: El generador de secuencias de Oracle se utiliza para generar números únicos y utilizarlos, por ejemplo, como claves de tablas. La principal ventaja es que libera al programador de obtener números secuenciales que no se repitan con los que pueda generar otro usuario en un instante determinado.

ARQUITECTURA DE ORACLE

Gráfico #13
Arquitectura de Oracle



Fuente: http://www.napolifirewall.com/ORACLE.htm

La Arquitectura general de Oracle consiste de varios procesos corriendo en la máquina donde reside la instancia, más los espacios de memoria dedicados a ejecutar procesos específicos o al almacenaje de información de cada proceso y la base de datos física propiamente tal, con sus archivos de control, de datos y de transacciones.

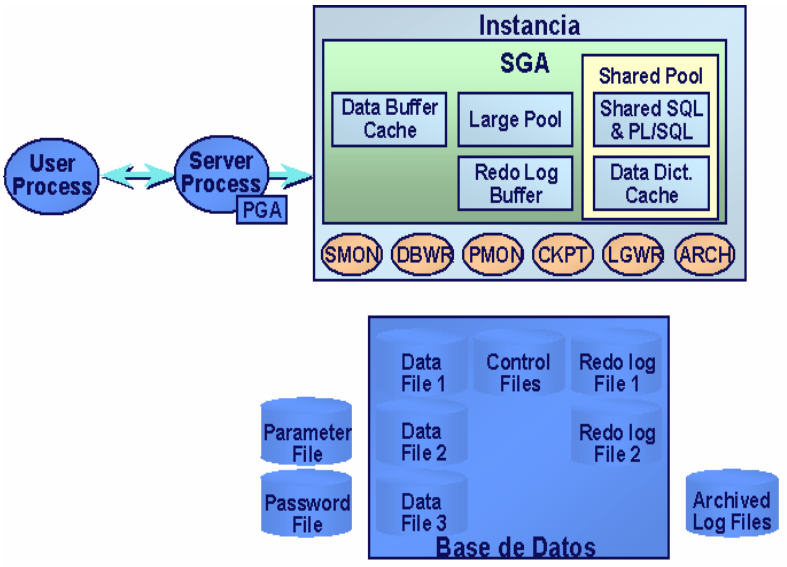
LA INSTANCIA ORACLE

Una instancia de Oracle está conformada por varios procesos y espacios de memoria compartida que son necesarios para acceder a la información contenida en la base de datos.

La instancia está conformada por procesos del usuario, procesos que se ejecutan en el background de Oracle y los espacios de memoria que comparten estos procesos.

Gráfico # 14

Instancia de Oracle



Fuente: http://www.napolifirewall.com/ORACLE.htm

El Área Global del Sistema (SGA)

El SGA es un área de memoria compartida que se utiliza para almacenar información de control y de datos de la instancia. Se crea cuando la instancia es levantada y se borra cuando ésta se deja de usar (cuando se hace shutdown). La información que se almacena en esta área consiste de los siguientes elementos, cada uno de ellos con un tamaño fijo:

EL BUFFER DE CACHÉ (DATABASE BUFFER CACHE)

Almacena los bloques de datos utilizados recientemente (se hayan o no confirmado sus cambios en el disco). Al utilizarse este buffer se reduce las operaciones de entrada y salida y por esto se mejora el rendimiento.

El buffer de redo log: Guarda los cambios efectuados en la base de datos. Estos buffers escriben en el archivo físico de redo log tan rápido como se pueda sin perder eficiencia.

Este último archivo se utiliza para recuperar la base de datos ante eventuales fallas del sistema.

El área shared pool: Esta sola área almacena estructuras de memoria compartida, tales como las áreas de código SQL compartido e información interna del diccionario. Una cantidad insuficiente de espacio asignado a esta área podría redundar en problemas de rendimiento. En resumen, contiene las áreas del caché de biblioteca y del caché del diccionario de datos.

- El caché de biblioteca se utiliza para almacenar código SQL compartido. Aquí se manejan los árboles de parsing y el plan de ejecución de las que se desee. Si varias aplicaciones utilizan la misma sentencia SQL, esta área compartida garantiza el acceso por parte de cualquiera de ellas en cualquier instante.
- El caché del diccionario de datos está conformado por un grupo de tablas y vistas que se identifican la base de datos. La información que se almacena aquí guarda relación con la estructura lógica y física de la base de datos.
- El diccionario de datos contiene información tal como los privilegios de los usuarios, restricciones de integridad definidas para algunas tablas, nombres y tipos de datos de todas las columnas y otra información acerca del espacio asignado y utilizado por los objetos de un esquema.

PROCESOS DE LA INSTANCIA

Los procesos que se implementan en una instancia de Oracle y su función principal son los siguientes:

DBWR (database writer): Es el responsable de la escritura en disco de toda la información almacenada en los buffers de bloques que no se han actualizado.

LGWR (log writer): Es el responsable de escribir información desde el buffer de log hacia el archivo redo log.

CKPT (checkpoint): Es el responsable de advertir al proceso DBWR de efectuar un proceso de actualización en el disco de los datos mantenidos en memoria, incluyendo los datafiles y control files (para registrar el checkpoint). Este proceso es opcional, si no está presente, es el proceso LGWR quien asume la responsabilidad de la tarea.

PMON (process monitor): Su misión es monitorizar los procesos del servidor y tomar acciones correctivas cuando alguno de ellos se interrumpe en forma abrupta, limpiando la caché y liberando los posibles recursos que pudieran estar asignados en ese momento.

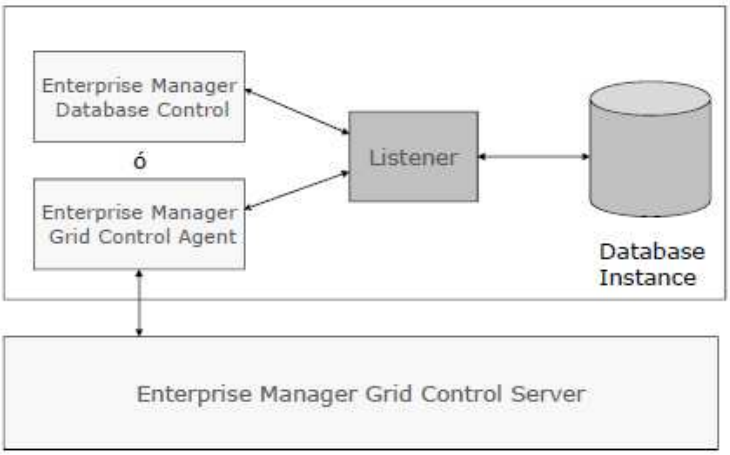
También es responsable por el restablecimiento de aquel proceso que se ha interrumpido bruscamente.

SMON (system monitor): Levanta una instancia cuando se le da la instrucción de partida (al comienzo del trabajo, encontrándose previamente en shutdown). Enseguida limpia los segmentos temporales y recupera las transacciones que pudieran haberse interrumpido debido a una falla del sistema. Además disminuye la fragmentación del sistema agrupando aquellas extensiones libres que existen dentro de la base de datos.

ARCH (archiver): La función de este proceso es la de respaldar la información almacenada en los archivos redo log cuando éstos se llenan. Este proceso está siempre activo cuando se ha establecido el modo ARCHIVELOG. Si el sistema no está operando

en este modo se hace más difícil recuperar el sistema sin problemas luego de una falla general.

Gráfico # 15
Instancia ORACLE



Fuente: http://www.napolifirewall.com/ORACLE.htm

Levantando Servicios

Listener
Utilitario lsnrctl
Puede ser en línea de comando o ambiente interactivo
lsnrctl [start | stop | status]
Enterprise Manager Database Control
Utilitario emctl
emctl [start | stop | status] dbconsole
iSQL*Plus
Utilitario isqlplusctl
Isqlplusctl [start | stop]

El Área Global de Programas (PGA)

Esta área de memoria contiene datos e información de control para los procesos que se ejecutan en el servidor de Oracle (relacionados con la base de datos, por supuesto). El tamaño y contenido de la PGA depende de las opciones del servidor que se hayan instalado.

Las Transacciones

El término transacción describe a una unidad lógica de trabajo que está compuesta de una o más sentencias SQL, que deben terminar con una instrucción commit o rollback. En ese instante, una nueva transacción dará comienzo y estará activa hasta que se ejecute alguno de esos dos comandos otra vez.

Cabe destacar que una transacción no se considera confirmada hasta que ésta se termina de escribir en el archivo de redo log.

Tareas del Administrador de la Base de Datos

- ✓ Dimensionar las necesidades de hardware
- ✓ Instalar el Software en los servidores
- ✓ Intervenir en los diseños de las bases de datos
- ✓ Crear las bases de Datos
- ✓ Realizar las actualizaciones de version
- ✓ Inicializar y detener las instancias
- ✓ Administrar las estructuras de Almacenamiento
- ✓ Administrar las cuentas de usuario y la seguridad
- ✓ Administrar los objetos de los esquemas
- ✓ Establecer el plan respaldos y realizar recuperaciones cuando sean necesarias
- ✓ Monitorear proactivamente el funcionamiento
- ✓ Afinar la base de datos para mejorar el performance

6.4.1 CAPITULO 4

6.4.1.1 PROGRAMACION EN PLSQL ORACLE

CONCEPTO DE PL/SQL

El PL/SQL soporta todas las consultas y manipulación de datos que se usan en SQL, pero incluye nuevas características:

- El manejo de variables.
- Estructuras modulares.
- Estructuras de control de flujo y toma de decisiones.
- Control de excepciones.

El lenguaje PL/SQL está incorporado en:

- Servidor de la base de datos.
- Herramientas de Oracle (Forms, Reports).

En un entorno de base de datos los programadores pueden construir bloques PL/SQL para utilizarlos como procedimientos o funciones, o bien pueden escribir estos bloques como parte de scripts SQL*Plus.

PL/SQL amplia SQL con los elementos característicos de los lenguajes de programación, variables, sentencias de control de flujo, bucles, etc.

Cuando se desea realizar una aplicación completa para el manejo de una base de datos relacional, resulta necesario utilizar alguna herramienta que soporte la capacidad de consulta del SQL y la versatilidad de los lenguajes de programación tradicionales. PL/SQL es el lenguaje de programación que proporciona **Oracle** para extender el SQL estándar con otro tipo de instrucciones.

Los programas o paquetes de PL/SQL se pueden almacenar en la base de datos como otro objeto, y todos los usuarios que estén autorizados tienen acceso a estos paquetes. Los programas se ejecutan en el servidor para ahorrar recursos a los clientes.

Oracle es una potente herramienta cliente/servidor para la gestión de bases de datos.

ORACLE FORMS SERVICIOS

Oracle Forms, un componente de Oracle Fusion Middleware, se establece la tecnología a largo Oracle para diseñar y construir aplicaciones empresariales de forma rápida y eficiente. Oracle mantiene su compromiso con el desarrollo de esta tecnología, y la liberación continua como un componente de la plataforma Oracle.

Este continuo compromiso con la tecnología de formularios le permite aprovechar su inversión existente al permitir el mejoramiento y la integración de aplicaciones existentes de Oracle Forms para aprovechar las tecnologías web y arquitecturas orientadas a servicios (SOA).

BLOQUES DE INSTRUCCIONES PL/SQL

Es la estructura general de los bloques de instrucciones de PL/SQL que se usarán mas adelante en la creación de procedimientos, funciones y triggers.

La unidad de programación utilizada por PL/SQL es el bloque. Todos los programas de PL/SQL están conformados por bloques. Tipicamente, cada bloque lleva a cabo una acción lógica en el programa. Un bloque tendrá siempre la siguiente estructura:

```
DECLARE

//Sección declarativa: variables, tipos, y subprogramas
//de uso local

BEGIN

//Sección ejecutable: las instrucciones procedimentales, y de SQL
//aparecen aquí. Es la unica sección obligatoria en el bloque.

EXCEPTION

//Sección de manejo de excepciones. Las rutinas de manejo de errores aparecen aquí

END;
```

CURSORES EXPLICITOS EN PL/SQL

Declaración de cursores explicitos. Los cursores explicitos se emplean para realizar consultas **SELECT** que pueden devolver cero filas, o más de una fila.

Para trabajar con un cursor explicito necesitamos realizar las siguientes tareas:

- Declarar el cursor.
- Abrir el cursor con la instrucción **OPEN**.
- Leer los datos del cursor con la instrucción **FETCH**.
- Cerrar el cursor y liberar los recursos con la instrucción **CLOSE**

PROCEDIMIENTO ALMACENADO

Es un conjunto de instrucciones en PL/SQL, que pueden ser llamados usando el nombre que se le haya asignado.

Es un programa (o procedimiento) el cual es almacenado físicamente en una base de datos. Su implementación varía de un manejador de bases de datos a otro.

La ventaja de un procedimiento almacenado es que al ser ejecutado, en respuesta a una petición de usuario, es ejecutado directamente en el motor de bases de datos, el cual usualmente corre en un servidor separado. Como tal, posee acceso directo a los datos que necesita manipular y sólo necesita enviar sus resultados de regreso al usuario, deshaciéndose de la sobrecarga resultante de comunicar grandes cantidades de datos salientes y entrantes.

Usos típicos para procedimientos almacenados incluyen la validación de datos siendo integrados a la estructura de base de datos (los procedimientos almacenados utilizados para este propósito a menudo son llamados disparadores; *triggers* en inglés), o encapsular un proceso grande y complejo. El último ejemplo generalmente ejecutará más rápido como un procedimiento almacenado que de haber sido

implementado como, por ejemplo, un programa corriendo en el sistema cliente y comunicándose con la base de datos mediante el envío de consultas SQL y recibiendo sus resultados.

Los procedimientos pueden ser ventajosos: Cuando una base de datos es manipulada desde muchos programas externos. Al incluir la lógica de la aplicación en la base de datos utilizando procedimientos almacenados, la necesidad de embeber la misma lógica en todos los programas que acceden a los datos es reducida. Esto puede simplificar la creación y, particularmente, el mantenimiento de los programas involucrados.

Podemos ver un claro ejemplo de estos procedimientos cuando requerimos realizar una misma operación en un servidor dentro de algunas o todas las bases de datos y a la vez dentro de todas o algunas de las tablas de las bases de datos del mismo. Para ello podemos utilizar a los Procedimientos almacenados auto creables que es una forma de generar ciclos redundantes a través de los procedimientos almacenados.

Estructuras de control en PL/SQL

Permiten modificar el flujo de ejecución de las instrucciones de un programa.

Estrcuturas de control de flujo

En PL/SQL solo disponemos de la estructura condicional IF. Su sintaxis se muestra a continuación:

Un aspecto a tener en cuenta es que la instrucción condicional anidada es ***ELSIF*** y no *"ELS**E**IF"*.

Sentencia GOTO

PL/SQL dispone de la sentencia GOTO. La sentencia GOTO desvia el flujo de ejecució a una determinada etiqueta.

En PL/SQL las etiquetas se indican del siguiente modo: << *etiqueta* >>

El siguiente ejemplo ilustra el uso de GOTO.

Bucles

En PL/SQL tenemos a nuestra disposición los siguientes iteradores o bucles:

- LOOP
- WHILE
- FOR

El bucle **LOOP,** se repite tantas veces como sea necesario hasta que se fuerza su salida con la instrucción **EXIT.** Su sintaxis es la siguiente

El bucle **WHILE,** se repite mientras que se cumpla *expresion*.
El bucle **FOR**, se repite tanta veces como le indiquemos en los identificadores *inicio* y *final.*

En el caso de especificar **REVERSE** el bucle se recorre en sentido inverso.

6.5.1 CAPITULO 5

6.5.1.1 LENGUAJE DE PROG. – ORACLE FORMS DEVELOPER 10G

GENERALIDADES DE LA BASE DE DATOS DE ORACLE 10G.

Oracle es un sistema de administración de base de datos (o RDBMS Relational Data Base Management System por las siglas en inglés), fabricado por Oracle corporation, básicamente una herramienta cliente/servidor para la gestión de Bases de Datos. Es un producto vendido a nivel mundial, aunque la gran potencia que tiene y su elevado precio hacen que sólo se vea en empresas muy grandes y multinacionales, por norma general.

Oracle, es una base de datos Orientadas a Objetos (BDOO) con beneficios OODBMS (Object Oriented DataBase Management System), ya que permite almacenar y manipular información que puede ser digitalizada (representada) por Objetos,

proporciona una estructura flexible con acceso ágil, rápido con gran capacidad de modificación.

Oracle es sin duda una de las mejores bases de datos que tenemos en el mercado, es un sistema gestor de base de datos robusto, tiene muchas características que nos garantizan la seguridad e integridad de los datos; que las transacciones se ejecuten de forma correcta, sin causar inconsistencias; ayuda a administrar y almacenar grandes volúmenes de datos; estabilidad, escalabilidad y es multiplataforma.

Con todas las características antes mencionadas de Oracle nos garantizan la seguridad de todos los datos migrados teniéndolos en un repositorio de alta confidencialidad, los cuales tuvieron que pasar por un proceso de migración utilizando la herramienta “Power Builder 10.0” que es muy poderosa, con múltiples opciones de la herramienta presentadas al usuario y paneles de crear objetos y editar sintaxis SQL y PL/SQL, compatible (en drivers) con Oracle y otros motores.

CARACTERISTICAS IMPORTANTES

Oracle pone al alcance del DBA varios niveles de seguridad:

- Seguridad de cuentas para la validación de usuarios.
- Seguridad en el acceso a los objetos de la base de datos.
- Seguridad a nivel de sistema para la gestión de privilegios globales.

Seguridad de Cuentas

Para acceder a los datos en una BD Oracle, se debe tener acceso a una cuenta en esa BD. Cada cuenta debe tener una palabra clave o *password* asociada. Una cuenta en una BD puede estár ligada con una cuenta de sistema operativo.

Los *passwords* son fijados cuando se crea un usuario y pueden ser alterados por el DBA o por el usuario mismo. La BD almacena una versión encriptada del *password* en una tabla del diccionario llamada dba_users. Si la cuenta en la BD está asociada a

una cuenta del sistema operativo puede evitarse la comprobación del *password*, dándose por válida la comprobación de la identidad del usuario realizada por el SO.

Seguridad de Objetos

El acceso a los objetos de la BD se realiza via privilegios. Estos permiten que determinados comandos sean utilizados contra determinados objetos de la BD. Esto se especifica con el comando GRANT, *conceder*. Los privilegios se pueden agrupar formando lo que se conoce por roles. La utilización de los roles simplifica la administración de los privilegios cuando tenemos muchos usuarios. Los roles pueden ser protegidos con *passwords*, y pueden activarse y desactivarse dinámicamente, con lo que constituyen una capa más de seguridad en el sistema.

Roles del Sistema

Los roles se pueden utilizar para gestionar los comandos de sistema disponibles para los usuarios. Estos incluyen comandos como CREATE TABLE o SELECT ANY TABLE. Todos los usuarios que quieran acceder a la BD deben tener el rol CONNECT; aquellos que necesiten crear segmentos necesitaran el rol RESOURCE. Un usuario con el rol DBA tiene derecho para ver y manejar todos los datos de la BD. En Oracle CONNECT, RESOURCE y DBA son roles de sistema. Las acciones contra cada tipo de objeto son autorizadas por privilegios separados. Así, un usuario puede tener concedido el privilegio CREATE TABLE, pero no el ALTER TABLE.

Arquitectura Oracle

Es indispensable conocer los factores y parámetros que influyen en el funcionamiento de nuestro SGBD para poder solucionar los problemas que se pueden plantear en cuanto nos salgamos de las aplicaciones estándares y básicas de BD, o en cuanto tengamos algún problema.

Tiene tres componentes básicos.

- ✓ **Las estructuras de memoria**: Almacenar los datos y el codigo ejecutable.

- ✓ **Procesos:** Que corren en el sistema de base de datos y las tareas de cada usuario conectando a la base de datos.
- ✓ **Archivos:** Sirven para el almacenamiento físico, en disco, de la información de la base de datos.

Gráfico # 16

Estructura de Oracle

Fuente: http://mksistemas.wordpress.com/2007/12/13/arquitectura-de-oracle/

ESTRUCTURAS DE MEMORIA

Hay dos clases, una de ellas compartida por todos los usuarios conectados y otra dedicada al trabajo de cada uno de ellos. El área global del sistema o SGA, es el área compartida por todos los usuarios, esta dividida en tres partes.

*Fondo común compartido (Shared pool), se mantiene el dicionario de datos y las areas compartidas de las órdenes SQL que se solicita para su procesamiento.

*Area de memoria rápida (Database buffer cache), se mantiene los datos traídos por las órdenes SQL de los usuarios conectados a la base de datos.

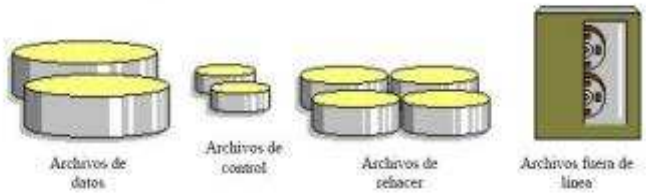
*Area de registros de rehacer (REdo lof buffer), se registran los cambios hechos a la base de datos.

Cada sesión de usuario se crea también en memoria llamada área global o PGA, esta área no se comparte con otras sesiones de usuario.

LOS ARCHIVOS

Se clasifican en cuatro grupos:

Gráfico #17
Tipos de Archivos



Fuente: http://mksistemas.wordpress.com/2007/12/13/arquitectura-de-oracle/

*Los Archivos de Datos (DAtafiles): Son los únicos que contienen los datos de los usuarios de la base de datos, sirven para el almacenamiento físico de las tablas, índices o agrupamientos y procedimientos.

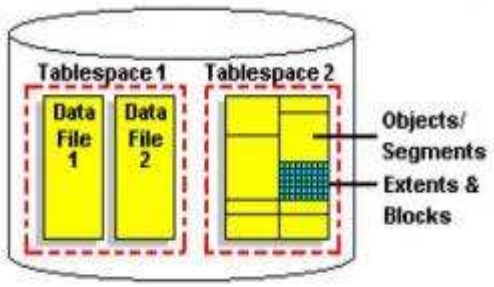
*Espacio de Tablas (Tablespaces): Unidades lógicas para el almacenamiento de los datos, permiten manejar o controlar el espacio en los discos.

No todos los espacios de tablas pueden estar en un mismo disco, para un buen desempeño y manejo del esapacio de almacenamiento es preferible crear es distintos discos. Se puede tener un solo espacio de tablas pero es mejor en varios espacios de tablas, por cada aplicación para los usuarios y para los índices.

El espacio de tablas SYSTEM se crea automáticamente al crear la BD, aqui se guardan los archivos de control, el diccionario de datos y la información de los procedimientos almacenados.

Gráfico # 18
Espacio System

Fuente:



http://mksistemas.wordpress.com/2007/12/13/arquitectura-de-oracle/

EL DBA crea un espacio de tablas de esta forma: CREATE

TABLESPACE.

Los archivos de datos datafiles almacena datos del usuario, estos archivos de datos son fijos en tamaño sino hay el suficiente espacio se debe adicionar más.

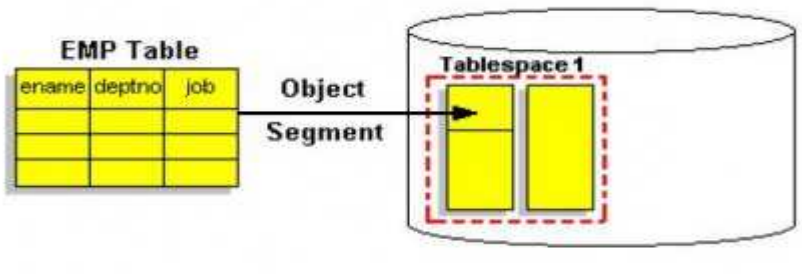
Si no hay espacio se tiene dos alternativas.

1. Adicionar un nuevo archivo de datos, asi ALTER TABLESPACE ALTER TABLESPACE.

2. Crear un nuevo espacio de tablas.

En la creación de una BD, el DBA debe estimar los requerimientos de almacenamiento, el nombre, tamaño y localización de los archivos de recuperar datos, junto con el número máximo de archivos de datos permitido. EL DBA puede crear varios espacios de tablas en discos separados para el crecimiento de la BD teniendo una mejor administración de BD.

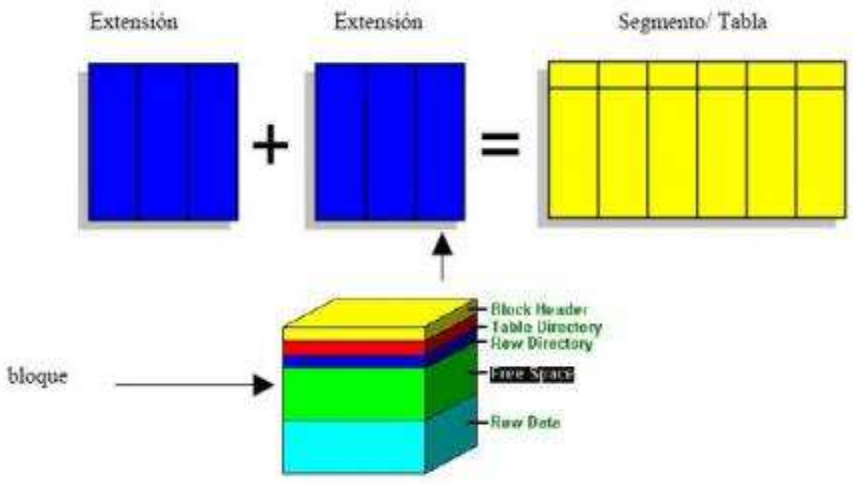
Gráfico #19
DBA



Fuente: http://mksistemas.wordpress.com/2007/12/13/arquitectura-de-oracle/

Una extención esta hecha por bloques que de acuerdo con el Sistema Operativo posee un nýmero de bytes, el tamaño del blque es dependiente del sistema y nunca puede ser menor al que este maneja.

Gráfico #20
Extension y Segmento de Tabla



Fuente: http://mksistemas.wordpress.com/2007/12/13/arquitectura-de-oracle/

En una BD pueden existir otros objetos que no contienen datos como las vistas, sinónimos y secuencias, todo objeto debe pertenecer a un esquema que puede ser creado de tres formas.

1. Si un usuario da una orden de creación de un objeto por defecto el sistema lo crea en su propio esquema.

2. Al nombre de un objeto siempre se le antepone el nombre del esquema (juan.empleado)con la orden Create table empleado as select *from juan.empleado;

3. Otro usuario lo crea para uno, la orden

Create table maria.proyecto(codigo number primay key…etc)
tablespace planeación
storage(initial 1000 next 1000
minextentents 6…)

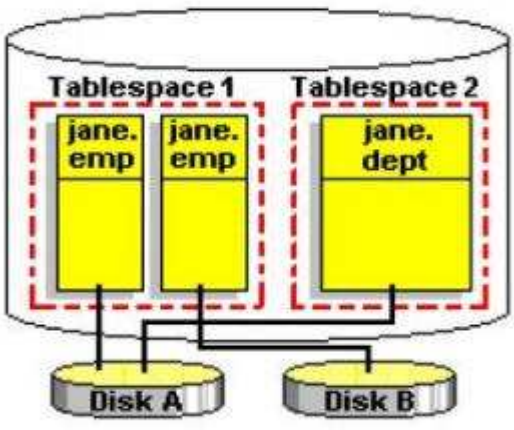
REGLAS PARA EL ALMACENAMIENTO DE OBJETOS EN LA BASE DE DATOS.

Un objeto puede almacenarse en uno o más archivos de datos (datafiles) pero de un solo espacio de tablas (tablaspace).

Dos objetos diferentes de un esquema pueden estar en dsitintos tablespaces. Los objetos pueden almacenarse en multiples discos como jane.emp es almacenado en el archivo de datos sobre el disco A y parte en archivo de datos sobre el disco B

Gráfico # 20

Esquema de Objetos



Fuente: http://mksistemas.wordpress.com/2007/12/13/arquitectura-de-oracle/

ARCHIVOS DE CONTROL (Control Files)

Descripción física y dirección de los archivos de la BD y de los archivos de rehacer, en etos archivos se especifica que datafiles conforman la BD para poder tener el acceso a los datos o para recuperar la base de datos ante una falla.

Los archivos de control se crean automaticamente con la orden CREATE DATABASE y no son editables, se actualizan automáticamente.

- **Archivos de rehacer (redo log files),** posee los cambios hechos a la base de datos para la recuperación ante fallas o manejo de transacciones, el próposito es de servir de respaldo de los datos en la memoria RAM, debe estar conformado por dos grupos mínimoy cada grupo debe estar almacenado en discos separados. El DBMS va sobrescribiendo sobre la información más vieja cuando se agota el espacio en estos grupos de archivos.
- **Archivos fuera de linea(archived files),** son archivos opcionales donde se guarda información vieja de los archivos de rehacer son convenientes para los respaldos de la base de datos.

LOS PROCESOS

Son programas que se ejecutan para permitir el acceso de datos, estos se cargan en memoria y son transparentes para los usuarios, se clasifican en tres grupos: Procesos de base, de usuario y procesos servidores.

- Los Procesos de Base o Soporte.

Los procesos de base (background), se encargan de traer datos desde y hacia la SGA, mejorando el desempeño de tareas impartidas por todos los usuarios, tiene su propia área de memoria.
Los procesos de base o soporte son lso siguientes:

DBWR (DAtabase writer), maneja los buffers de memoria caché para que los procesos del usuario encuentren uno de ellos disponibles, es un proceso obligatorio, escribe los bloques de datos modificados por los usuarios en los archivos que compone la BD.

LGWR (Log writer), escribe datos desde la SGA a los archivos de rehacer(redo log files)que sirve en caso de fallas en la instancia, este proceso es obligatorio y es el encargado de escribir y leer estos archivos, antes de empezar a sobrescribir en uno de ellos se marca el punto de verificación y LGWR envía las orden de escritura en los datafiles al proceso DBWR. LCKn, Lock(lock processes), Bloqueos entre instancias en ambientes con servidores paralelos(hasta 10 servidores), es opcional.

CKPT (Check point), la comprobación es un proceso opcional, cuando los usuarios conectados a la BD realizan solicitudes de examenes de datos.

SNPn (Snapshot process), refresca los snapshot ó replicas de tablas que se usan en ambientes distribuidos.

SMON (System monitor), recupera el sistema ante una falla de una instancia.

RECO (Recovery), recupera ante fallas, en una transacción en ambientes distribuidos.

ARCH (Archive), copia los registros de rehacer de la RAM en archivo de datos, permite la recuperación ante fallas de los medios magneticos.

PMON (Process Monitor), ante una falla de un proceso de usuario se recupera, libera los recursos del proceso ue falló

- ❖ PROCESOS DE USUARIO

Gráfico # 21
SGA

Fuente: http://mksistemas.wordpress.com/2007/12/13/arquitectura-de-oracle/

Cuando un usuario se conecta a la BD, se crea un proceso de usuario encargado de ejecutar el código de aplicación del usuario y manejar el perfil del usuario con sus variables de ambiente, los procesos de usuario no se pueden comunicar directamente con la BD lo hacen a travez de servidores

- PROCESOS SERVIDORES

Ejecutan órdenes SQL de los usuarios y llevan los datos al"database buffer cache", para que los procesos de usuario puedan tener acceso a los datos, hay distintas arquitecturas en ORACLE según los tipos de servidores.

- INSTANCIA DE ORACLE

Conjunto de estructuras de memoria y procesos que acceden los archivos de BD, una misma BD puede ser accedida por múltiples instancias, cada una de ellas residiendo en una máquina diferente. Pasos cuando el sistema de BD inicia.

1. Iniciar la instancia. ORACLE lee el archivo de parámetros y configura la instancia donde se crea la SGA y se activan los procesos de base, pero aún no se puede realizar nada.

2. Montar la base de datos, Preparar el sistema para su uso trayendo a la RAM el diccionario de datos.

3. Abrir la base de datos. Se abren los archivos y los usuarios ya pueden acceder a los datos.

De acuerdo como se defina la instancia ORACLE por sus parámetros se determina que tan poderoso es el motor, los parámetros se definen en el archivo INIT.ORA.

db_block_buffers=número de bloques de BD en la SGA, un buffer por cada bloque.
db_block_size=tamaño del bloque de la BD.
shared_pool_size=tamaño del área compartida en bytes

Aquí se especifica el número de usuarios concurrentes, transacciones concurrentes y los nombres de los archivos para la BD.

TIPOS DE DATOS ORACLE

Los tipos de datos Oracle se agrupan en los siguientes conjuntos como se muestra en el gráfico #21**.**

Gráfico # 22
Tipos de Datos Oracle

Alfanuméricos	**Numéricos**	**Fecha**	**Binarios**	**Otros**
CHAR	NUMBER	DATE	RAW	ROWID
VARCHAR(2)	FLOAT	TIMESTAMP	LONG RAW	UROWID
VARCHAR		TIMESTAMP WITH TIME ZONE	BLOB	
NCHAR		INTERVAL	CLOB	
NVARCHAR(2)			NLOB	
LONG			BFILE	

Elaborado: Babici Bogdan, Achig María, Tierra Doris
Fuente: Tópico de Graduación UPS.

6.6 MARCO CONCEPTUAL

Los términos relacionados con los módulos vistos para el desarrollo de nuestro Sistema que son necesarios tener en cuenta para facilitar su identificación, selección y definición posterior son los siguientes:

- **Proceso:** Conjunto de recursos y actividades interrelacionados que transforman elementos de entrada en elementos de salida. Los recursos pueden incluir personal, finanzas, instalaciones, equipos, técnicas y métodos.

- **Proceso clave:** Son aquellos procesos que inciden de manera significativa en los objetivos estratégicos y son críticos para el éxito del negocio.

- **Subprocesos:** Son partes bien definidas en un proceso. Su identificación puede resultar útil para aislar los problemas que pueden presentarse y posibilitar diferentes tratamientos dentro de un mismo proceso.

- **Sistema:** Estructura organizativa, procedimientos, procesos y recursos necesarios para implantar una gestión determinada, como por ejemplo la gestión de la calidad, la gestión del medio ambiente o la gestión de la prevención de riesgos laborales. Normalmente están basados en una norma de reconocimiento internacional que tiene como finalidad servir de herramienta de gestión en el aseguramiento de los procesos.

- **Procedimiento**: Forma especifica de llevar a cabo una actividad. En muchos casos los procedimientos se expresan en documentos que contienen el objeto y el campo de aplicación de una actividad; que debe hacerse y quien debe hacerlo; cuando, donde y como se debe llevar a cabo; que materiales, equipos y documentos deben utilizarse; y como debe controlarse y registrarse.

Actividad: Es la suma de tareas, normalmente se agrupan en un procedimiento para facilitar su gestión. La secuencia ordenada de actividades da como resultado un subproceso o un proceso. Normalmente se desarrolla en un departamento o función.

Proyecto: Suele ser una serie de actividades encaminadas a la consecución de un objetivo, con un principio y final claramente definidos. La diferencia fundamental con los procesos y procedimientos estriba en la no repetitividad de los proyectos.

Indicador: Es un dato o conjunto de datos que ayudan a medir objetivamente la evolución de un proceso o de una actividad.

Estrategas: "Son los individuos responsables del éxito o fracaso de una organización". [2]

Estrategia: Es describir el procedimiento y el método con el que se va a lograr la meta. Y Estrategias es un conjunto de decisiones y criterios por lo cual una organización se orienta la obtención de determinados objetivos

Fortalezas interna: "Se refiere a las actividades internas de una organización que se llevan a cabo especialmente bien. Las funciones de gerencia, mercadeo, finanzas, investigación entre otros".[3]

Debilidades interna: Se refiere a las actividades de gerencia, mercadeo, finanzas, investigación y desarrollo que limitan o inhiben el éxito general de una organización

---

[2]Fred David (1994),

[3] Fred David (1994),

Oportunidades externas: "Se refiere a las tendencias económicas, sociales, políticas, tecnológicas y competitivas".[4]

Amenazas externas: "Se refiere a las tendencias económicas, sociales, políticas, tecnológicas y competitivas. Así como hechos que son potencialmente dañinos para posición competitiva presente o futura de una organización. Las organizaciones exitosas crean estrategias que sirven para contrarrestar el impacto de las amenazas externas".5

Objetivos: Son los resultados a largo plazo que una organización aspira a lograr a través de su misión básica". Los objetivos son de vital importancia en el éxito de las organizaciones, pues suministran dirección, ayuda en evaluación, crean sinergia, revelan prioridades, permiten coordinación y son esenciales para las actividades de control, motivación, organización y planificación efectivas. Los objetivos deben reunir las siguientes características: ser medibles, razonables, claros, coherentes estimulantes. En un conglomerado diversificado, los objetivos deben fijarse tanto para la organización en general como para cada división.

Metas: (de un año o menos) Son como puntos de referencia o aspiraciones que las organizaciones deben lograr, con el objeto de alcanzar en el futuro objetivos a un plazo más largo. Ellas deben ser medibles, cuantitativas, realistas, estimulantes, coherentes y prioritarias. Las metas representan la base para la asignación de recursos.

La meta como la forma por medio de la cual las metas fijas van a lograrse, o las pautas establecidas para lograr las metas ya definidas.

1. Son guías para la toma de decisiones;

2. Se establecen para situaciones repetitivas o recurrentes en la vida de una estrategia.

---

[4] Fred David (1994),

[5] Fred David (1994)

Debilidades: Se refiere a las actividades internas de la gerencia, mercadeo, finanzas, producción, investigación y desarrollo que limitan o inhiben el éxito general de una organización".

Organización: Es la Coordinación racional de las actividades de un cierto número de personas, que intentan corregir una finalidad y objetivo común y explícito, mediante la división de las funciones y del trabajo, y a través de una jerarquización de la autoridad y de la responsabilidad".

Indicadores: Son las medidas que cuantifican o indican el nivel del logro, pueden ser, indicadores guías o indicadores de actuación o de resultado".

Amenazas: Son las tendencias Económicas, Sociales, Políticas, Tecnológicas y Competitivas; así como hechos que son potencialmente dañinos para la posición competitiva presente o futura de una organización".

Oportunidades: Son tendencias Económicas, Sociales, Tecnológicas y Competitivas, así como hechos que podrían de forma significativa beneficiar a una organización en el futuro".

Expertos: Son las personas que conocen a profundidad determinado tema.

Nivel estratégico: Conformado por los directivos de alto nivel y cuya función es establecer los planes y estrategias a seguir, así como también la toma de decisiones, deben tener un conocimiento general de las actividades de la empresa pues se le asignan responsabilidades y decisiones importantes".

Actores: Son las personas que juegan un papel importante en el sistema por medición de las variables que caracterizan sus proyectos".

Tabla: Es la unidad básica de almacenamiento en un sistema de bases de datos relacionales, en ellas son almacenados los datos de los usuarios y los datos del sistema Oracle.

Servidor.- es una computadora que, formando parte de una red, provee servicios a otras computadoras denominadas clientes.

Base de datos.- es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente.

Oracle 10G.- es una base de datos de entrada de footprint pequeño, creada sobre la base de código Oracle Database 10g Release 2 que puede desarrollarse, implementarse y distribuirse sin cargo; es fácil de descargar y fácil de administrar.

Dependencia.- es una conexión entre uno o más atributos.

Esquema.- Describe la estructura de una Base de datos.

Tablespace.- es una unidad lógica de almacenamiento dentro de una base de datos oracle.

Es un puente entre el sistema de ficheros del sistema operativo y la base de datos.

Cada tablespace se compone de, al menos, un datafile y un datafile solo puede pertenecer a un tablespace.

Cada tabla o indice de oracle pertenece a un tablespace, es decir cuando se crea una tabla o indice se crea en un tablespace determinado.

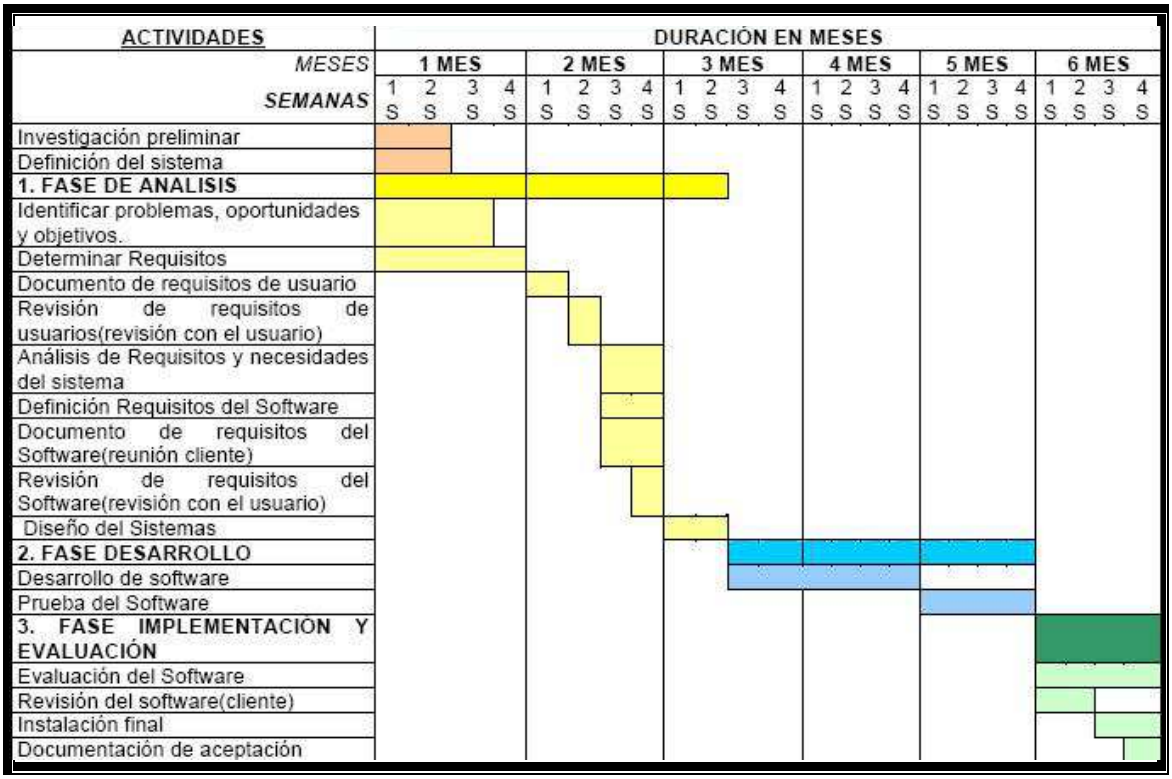
Vista.- En el modelo de datos relacional la forma de guardar la información no es la mejor para ver los datos.(consulta de datos predefinida de una tabla)

Paquete.- Conjunto de procedimientos almacenados, tipos o funciones.

Procedimiento Almacenado.- es un programa (o procedimiento) el cual es almacenado físicamente en una base de datos.

- **Usuario.-** Colección de objetos y privilegios identificado con un nombre y password
- **Rol .-** Conjunto de privilegios, asignables a un usuario o un rol.
- **Indice:** Así como el índice de un libro ayuda a accesar su contenido de una manera mas ágil, un índice de una tabla le ayuda a la base de datos a recuperar información con mayor velocidad.
- **Privilegio.-** Permiso para realizar una acción, asignable a un usuario o un rol
- **Triggers**.- Un trigger es un bloque PL/SQL asociado a una tabla, que se ejecuta cuando una determinada instrucción en SQL se va a ejecutar sobre dicha tabla.
- **Perfil:** Conjunto de restricciones relativas al uso de recursos, y asignable a usuarios. Un usuario sólo puede tener un perfil.
- **Recurso:** Uso susceptible de ser restringido, asignable a un perfil.
- **Datafiles:** Arhivos físicos que contienen toda la información de la base de datos; en ellos están estructuras tales como tablas e índices.
- **Expertos:** son las personas que conocen a profundidad de un tema determinado .
- **Nivel estratégico:** Esta conformado por los directivos de alto nivel y cuya función es establecer los planes y estrategias a seguir, así como también la toma de decisiones, deben tener un conocimiento general de las actividades de la empresa pues se le asignan responsabilidades y decisiones importantes".
- **Actores:** Personas que juegan un papel importante en el sistema por medición de las variables que caracterizan sus proyectos.

7. Cronograma de Actividades



Fuente: Tópico de Graduación UPS

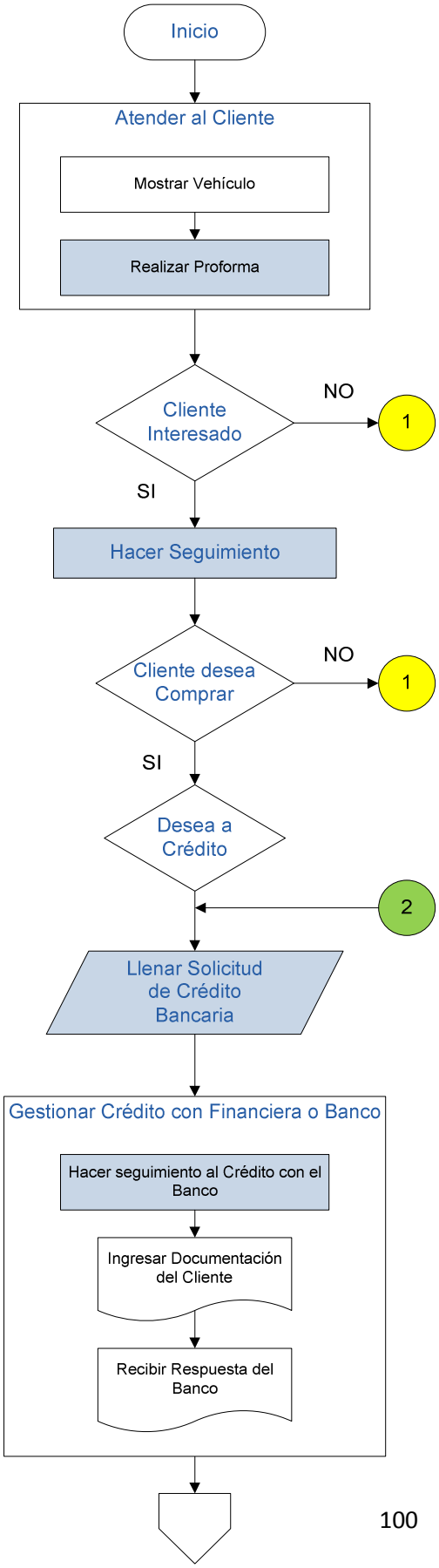
Elaborado: Babici Bogdan, Achig María, Tierra Doris

TERRASOFT©

SISTEMA DE GESTIÓN COMERCIAL PARA CONCESIONARIOS AUTOMOTRICES

8. MANUAL DE DISEÑO

Diagrama de Flujo de Terrasoft



Descripción

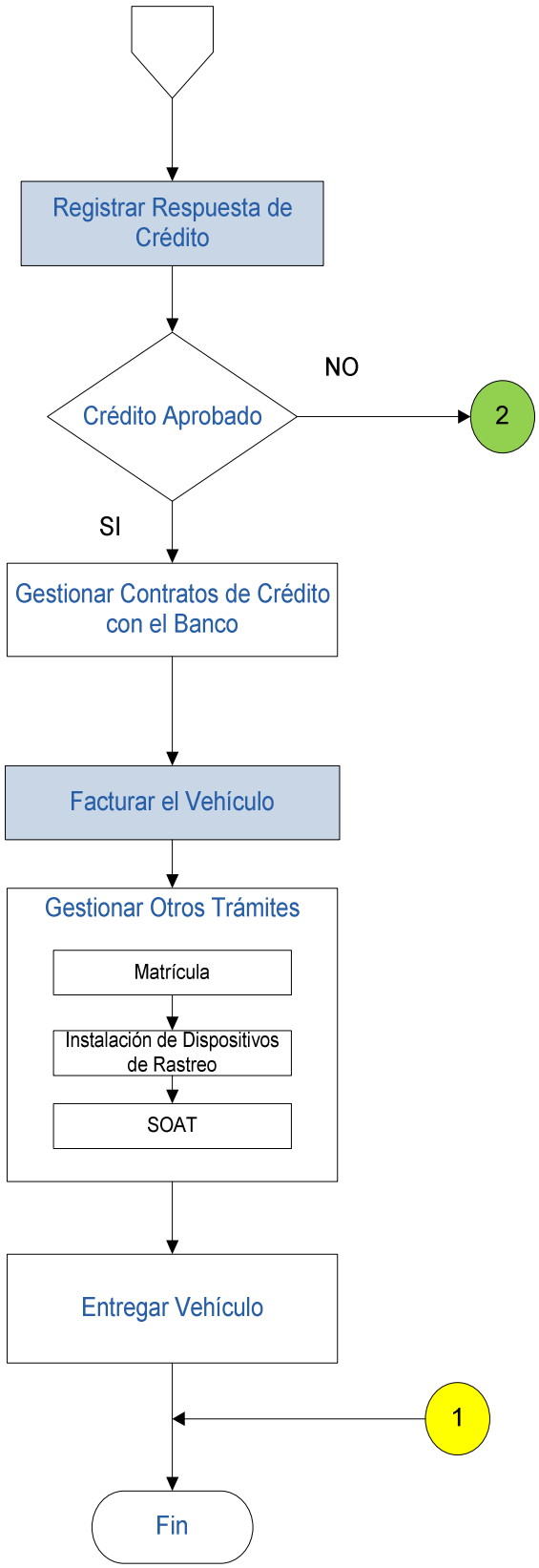
- Registrar los Datos del Cliente en una Proforma y el Vehículo de su Interés así como la Forma de Pago que el Cliente desea

- Realizar llamadas al cliente para verificar si ya se decidió por la Compra
- Pedir Documentación para los Créditos
- Tratar de convencer al Cliente para que se decida a hacer la Compra

- Llenar una Solicitud de Crédito por cada Banco al que se desea aplicar un crédito para el Cliente

- Llamar al banco para saber si el crédito ya fue aprobado y bajo qué condiciones.
- Hacer seguimiento a reconsideraciones de créditos negados
- Ir al Banco personalmente a Gestionar una respuesta
- Mandar documentación de respaldo económico del cliente vía mail, fax o personalmente

- Recibir la respuesta y registrarla, si fue Negado el Crédito se puede intentar de ser viable una Reconsideración.



- Solicitar al Banco la Emisión de los Contratos de Crédito a Firmar por el Cliente y Concesionario

  (Pagarés, Reservas de Dominio, etc.)

- Receptar Entrada del Cliente y con la Aprobación del Banco solicitar la Factura del Vehículo

- Realizar la entrega formal del Vehículo

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

NOTA: Nuestro Sistema en cuanto a este Proceso, tendrá acción en aquellas Tareas pintadas de color Celeste

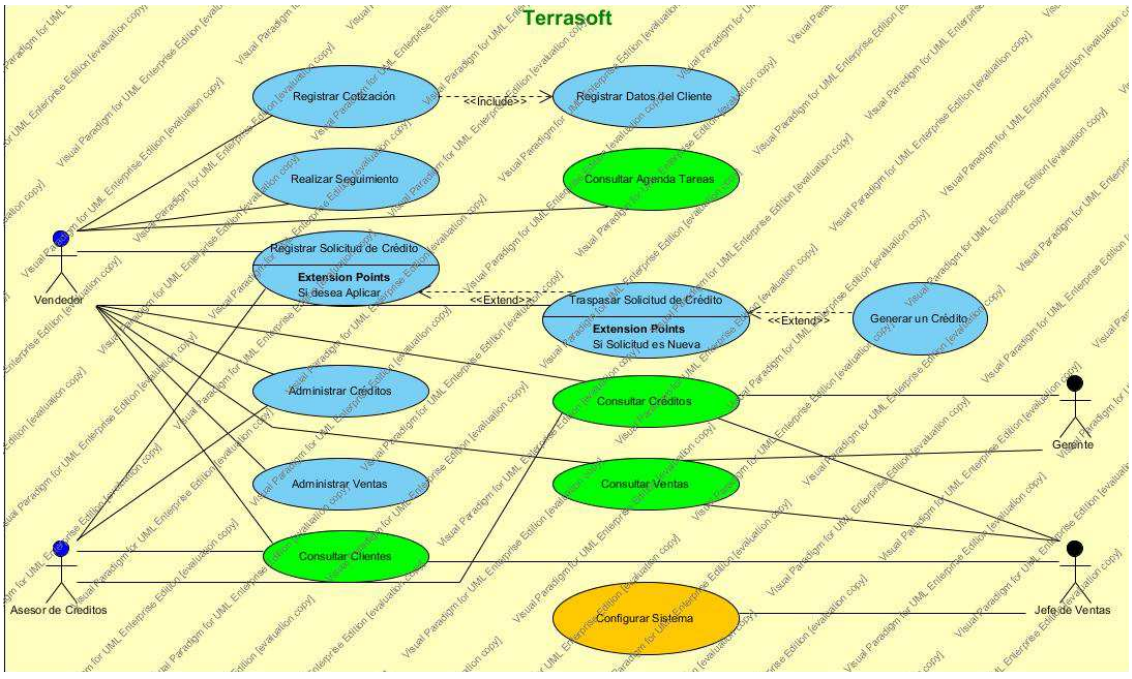
DIAGRAMAS UML

DIAGRAMA DE CASOS DE USO

En este diagrama podemos identificar a las personas involucradas en el sistema cada una tendrá los permisos necesarios según el cargo que posea dentro del Concesionario.

- Vendedor.-
  - Registrar cotización, solicitudes de crédito y nuevos clientes.
  - Realizar seguimientos.
  - Consultas de tareas, créditos, ventas y clientes.
  - Administrar créditos y ventas.
- Asesor de Crédito.-
  - Registra Solicitud.
  - Administra Créditos.
  - Consulta Cliente y Créditos.
- Gerente
  - Consulta Créditos y Ventas.
- Jefe de Ventas
  - Consulta créditos, Ventas y Clientes.
  - Configura Sistema.

Elaborado por: Babici Bogdan,Tierra Doris,Achig María.

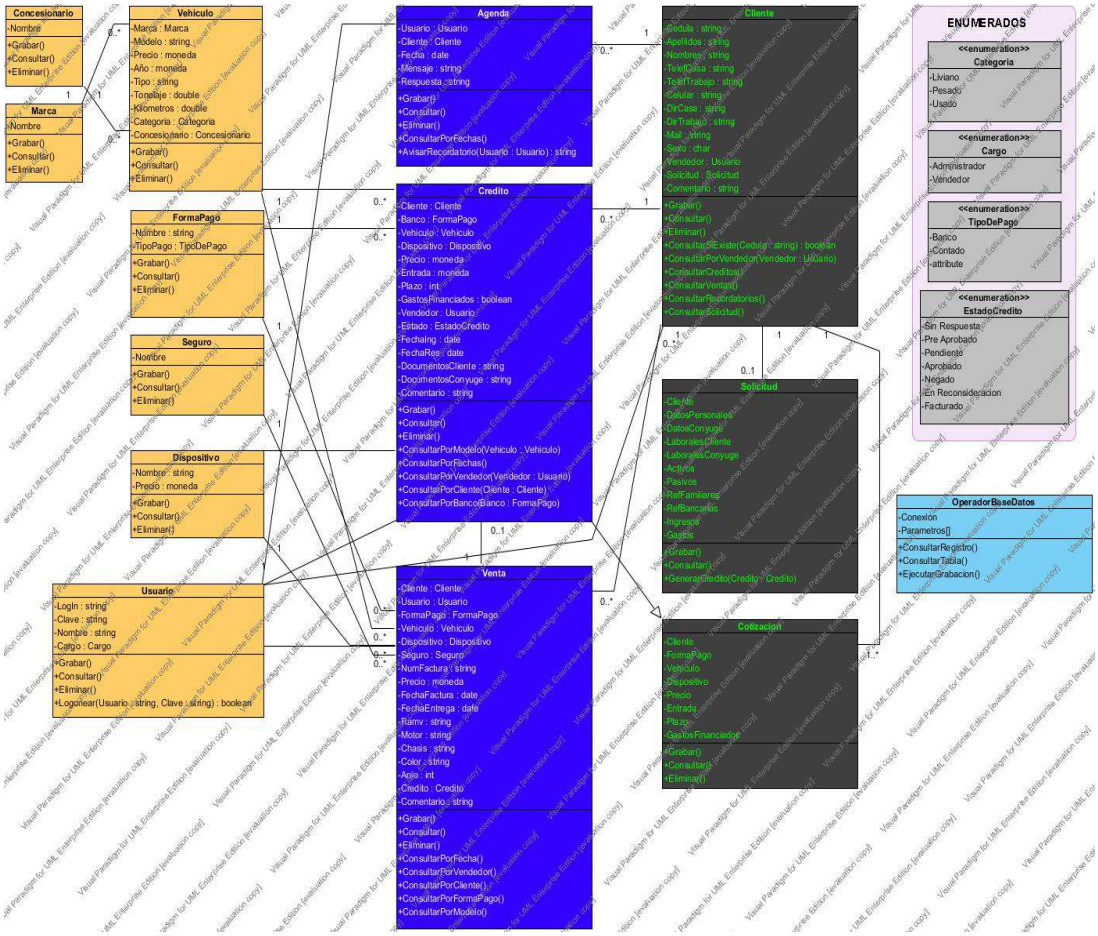


Fuente: Tópico de Graduación UPS.

Nombre de caso de uso:	Registro de Clientes
Actores Participantes:	Asesor de Créditos, Jefe de Ventas, Gerente, Vendedor
Condición inicial:	1) Un paciente se acerca a la ventanilla a solicitar un turno para la atención médica.
Flujo de Eventos:	2) si el paciente no tiene asignado un número de historia clínica, la recepcionista procede a la creación de la misma solicitando los datos al paciente. 3) Luego de esto, la recepcionista procede a registrar los antecedentes familiares y personales del paciente.
Condición de Salida:	4) Se genera el código y la ficha médica del paciente.
Requerimientos especiales:	El paciente no debe estar registrado en el sistema.

DIAGRAMA DE CLASES

En este diagrama tenemos detallado los campos que contiene cada una de las tablas involucradas en nuestro sistema, con sus respectivas relaciones.



Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

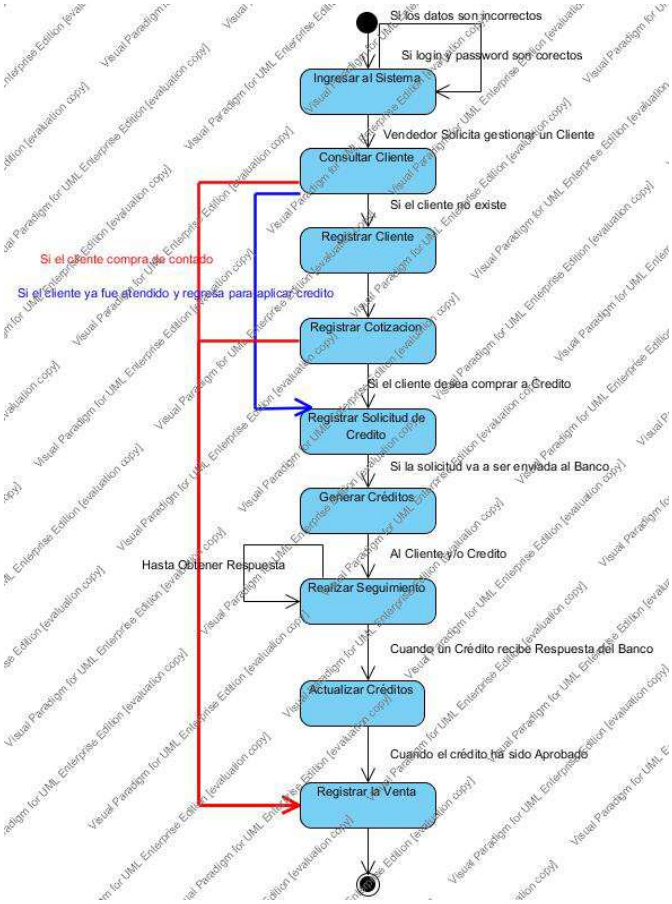
DIAGRAMAS MAQUINA DE ESTADO

***Vendedor**

En este diagrama se presenta todos los pasos o actividades que realizará el vendedor dentro del sistema.

El vendedor ingresará su login y password para acceder al módulo dentro del mismo podrá consultar y registrar clientes así como también registrar cotizaciones de los clientes interesados. Adicionalmente el vendedor podrá registrar las solicitudes de crédito del cliente que cumpla con los requerimientos del crédito y de la institución bancaria con la que aplicará el crédito.

Una vez realizada la solicitud de crédito el vendedor podrá realizar a su cliente en gestión el seguimiento respectivo hasta obtener una respuesta favorable y culminar finalmente en el registro de la venta del vehículo

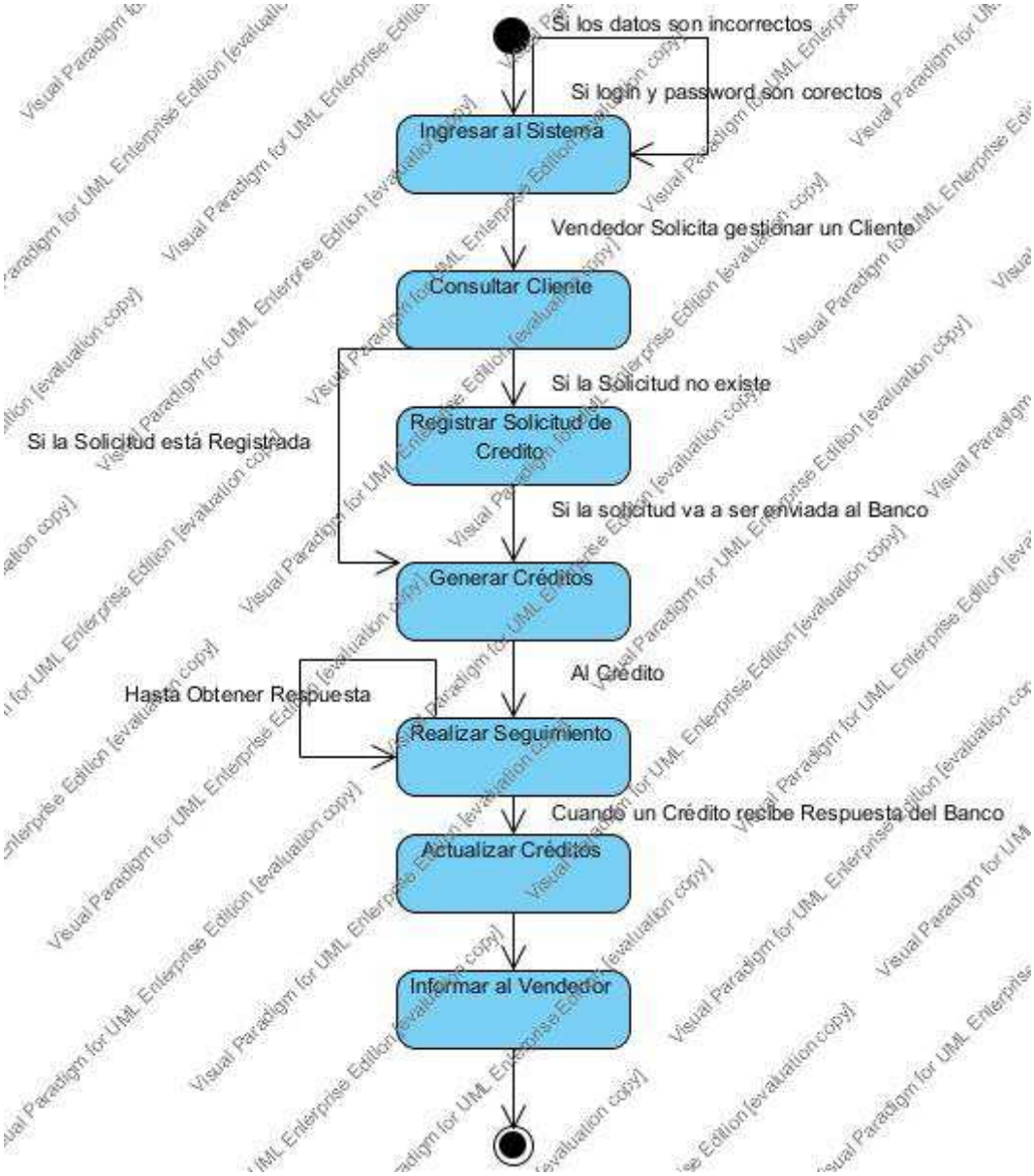


Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

***Asesor de Crédito**

El asesor de Crédito es la persona encargada de revisar las solicitudes de créditos precalificadas por el vendedor, tendrá su loggin y password para acceder a su módulo dentro del sistema así como consultar, realizar seguimientos, actualizar créditos y realizar o emitir informes al vendedor que esté a cargo de algún cliente específico que el asesor tenga interés en conocer más detalles notificándole alguna sugerencia o comentario respectivo.

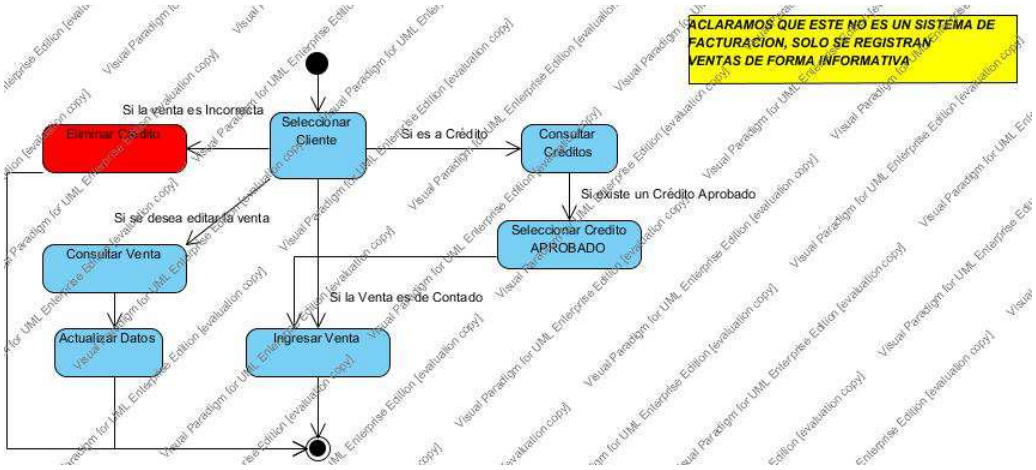


Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

*Administrar Ventas

Una vez registrada la solicitud de crédito y debidamente aprobada ésta constará en la base de datos del sistema y se podrá realizar selecciones por cliente, consultar créditos, seleccionar créditos aprobados, ingresar ventas, consultar ventas, actualizar datos y eliminar créditos que no cumplan con algún requisito.

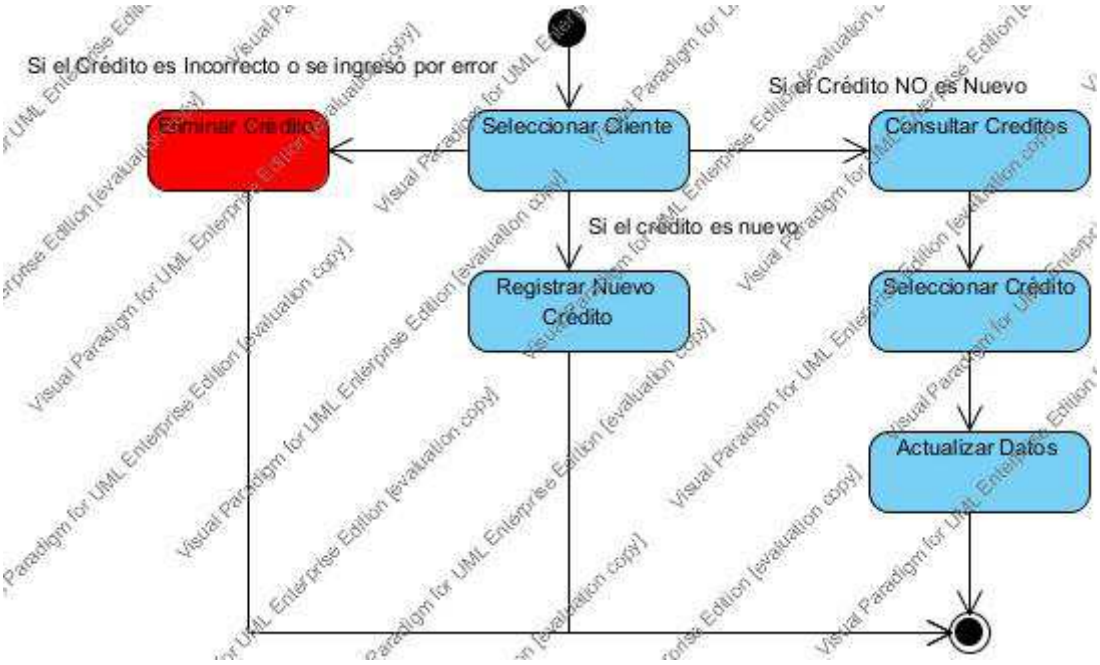


Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

*Administrar Créditos

En esta sección seleccionaremos el cliente, registraremos el nuevo crédito y a los clientes existentes podremos consultarles sus créditos, seleccionar sus créditos y actualizar datos.



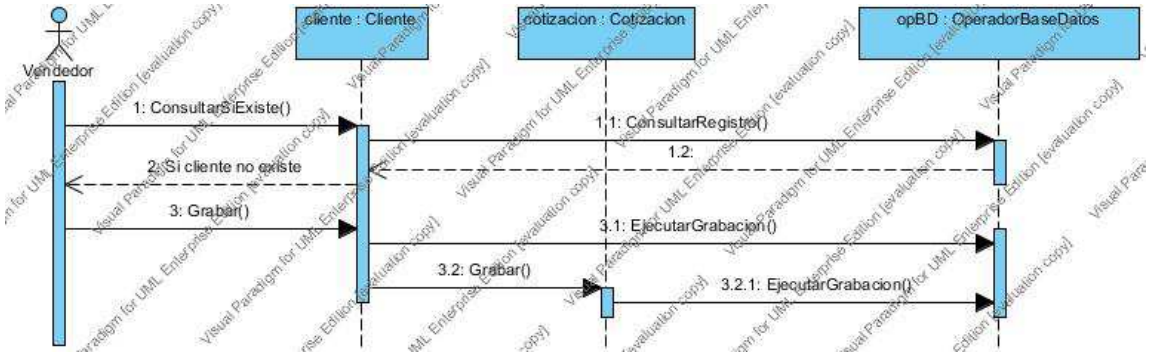
Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

DIAGRAMA DE SECUENCIAS

***Registrar Cliente Cotización.**

Esta secuencia la realiza el Vendedor recepta los datos de los clientes que se acercan a pedir una cotización del vehículo que desean adquirir; lo ingresan para tener almacenado en la base y posteriormente realizar el seguimiento para que el cliente adquiera el vehículo.

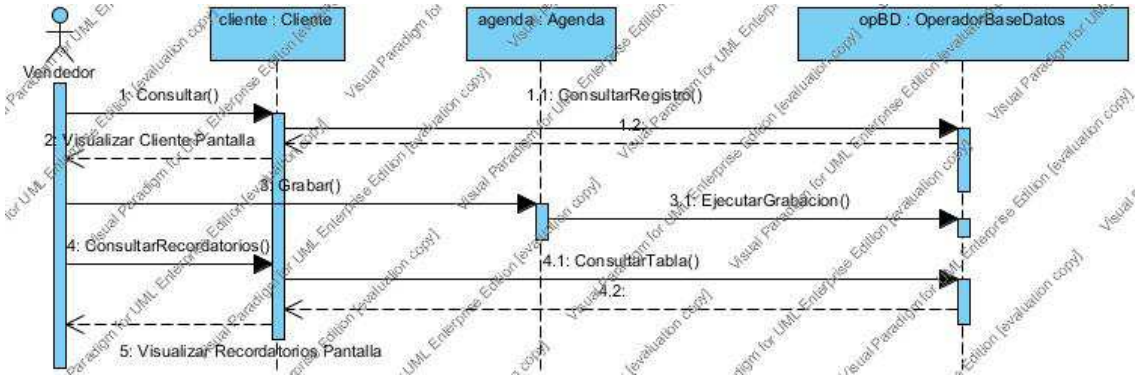


Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

***Realizar seguimiento.**

En esta secuencia el Vendedor consulta por uno de sus Clientes en la base, accede a la detalle del mismo donde está incluido el vehículo que desea adquirir.

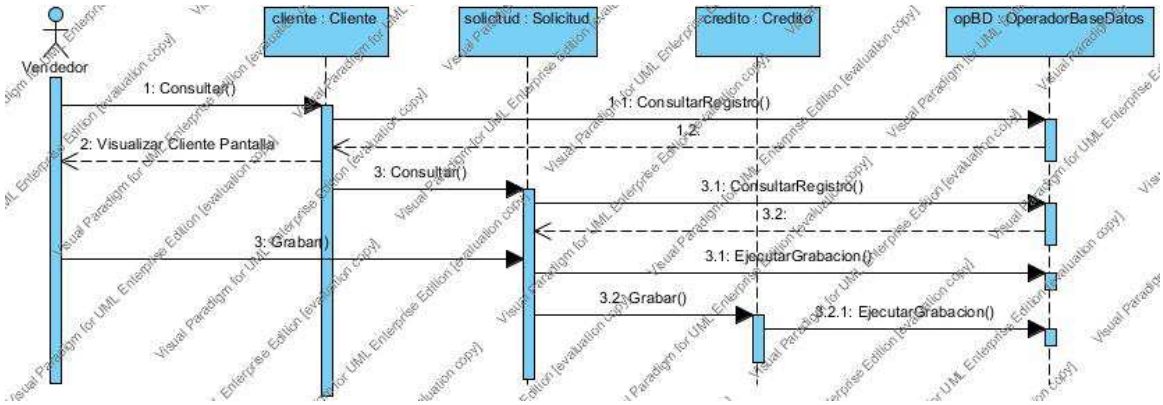


Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

***Registrar Solicitud de Crédito**

El Vendedor recepta la información en la cual el cliente acepta la compra del vehículo elegido, este imparte información acerca de su situación crediticia más detallada al vendedor para llenar los datos que necesita una institución financiera para otorgar un crédito.

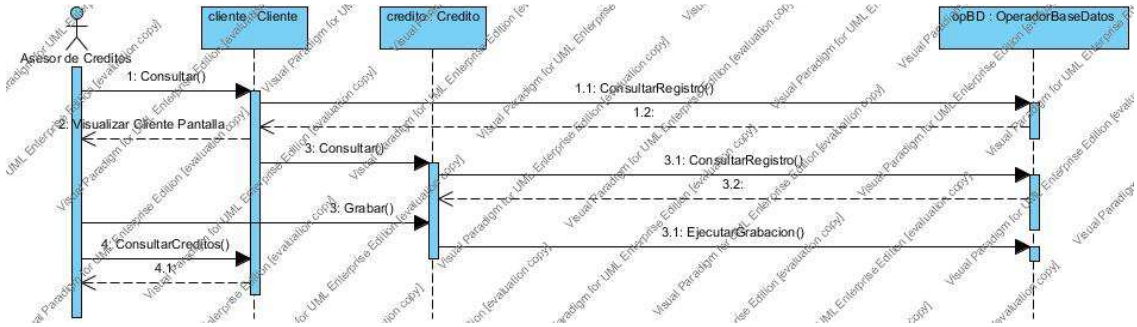


Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

***Administrar Créditos**

Aquí el Asesor de Créditos recepta la información que le provee el Vendedor internamente realiza sus pasos para ver que tan factible es otorgarle un préstamo al cliente que lo está solicitado. Esta respuesta se la almacena en la base del sistema para tener así un historial.



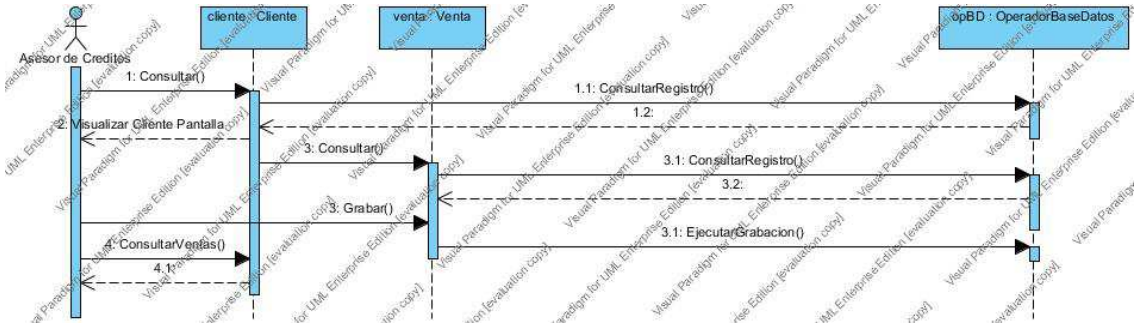
Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

***Administrar Ventas**

En esta secuencia el Asesor de Créditos consulta si el cliente adquiere el vehículo, registra la venta la almacena para poder tener el historial de ventas y en tiempos prudentes realizar seguimientos para que el cliente adquiera otro vehículo.

Esto ayudara a obtener reportes de las ventas que se realizan un periodo determinado dentro de la concesionaria.



Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

DIAGRAMA ENTIDAD RELACIÓN

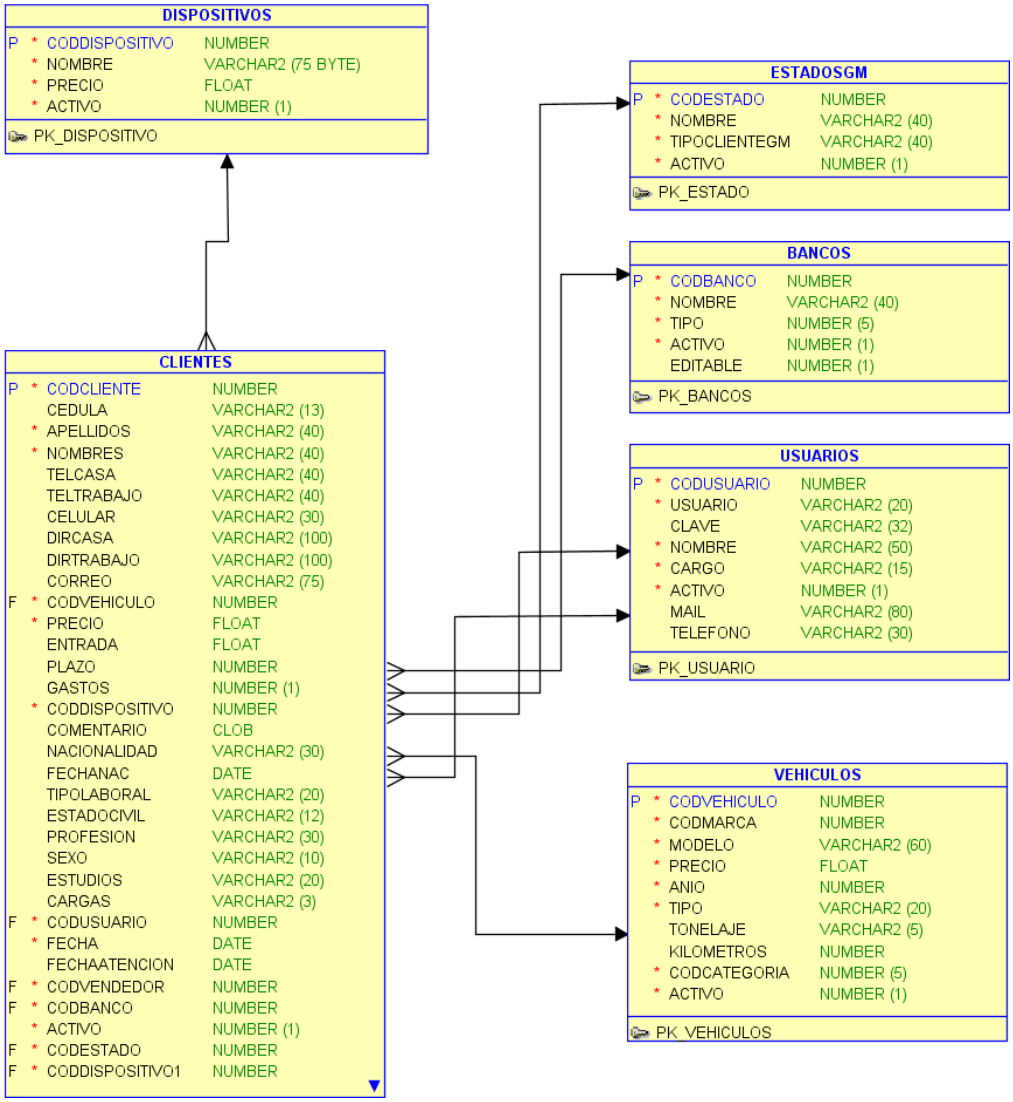
***Concesionario.**

El sistema Terrasoft contará con 23 tablas las mismas que estarán relacionadas entre sí. A continuación se detalla las tablas y su relación mediante el gráfico.

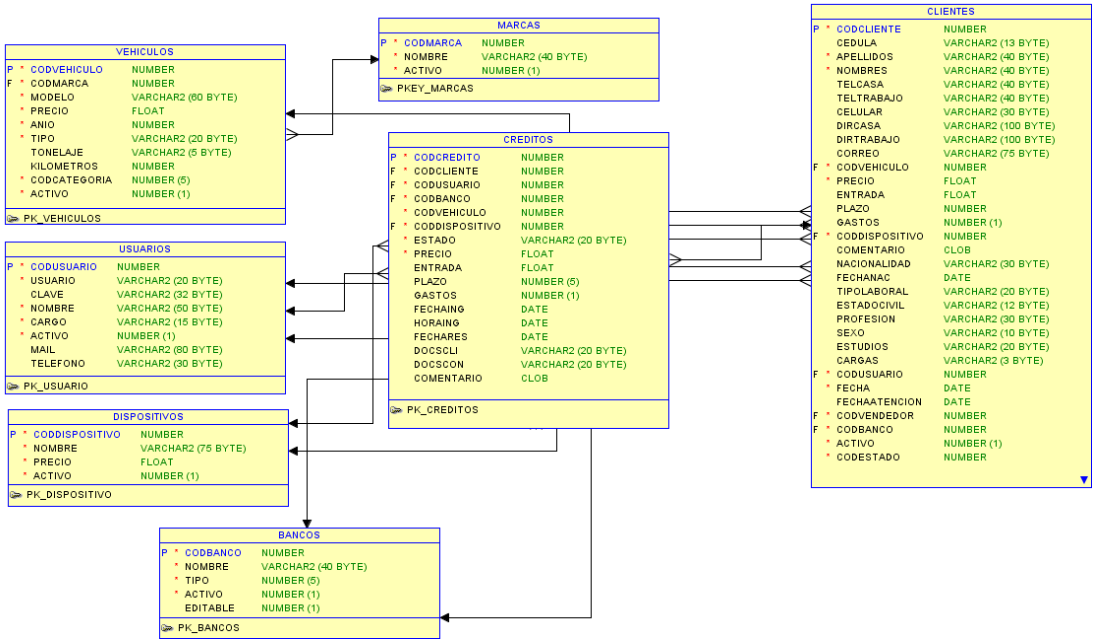
Clientes, Dispositivos, Bancos, Créditos, Marcas, Dealer, Seguros, Ventas, RefTarjetas, TrabajoClientes, Vehículo, Parientes, Categorías, Vivienda, Usuarios, RefComerciales, Gastos, EstadosGM, Agenda, TrabajosConyuge, Activos, Pasivos, Cónyuges.

Diagrama de Entidad Relaciòn Por Mòdulo

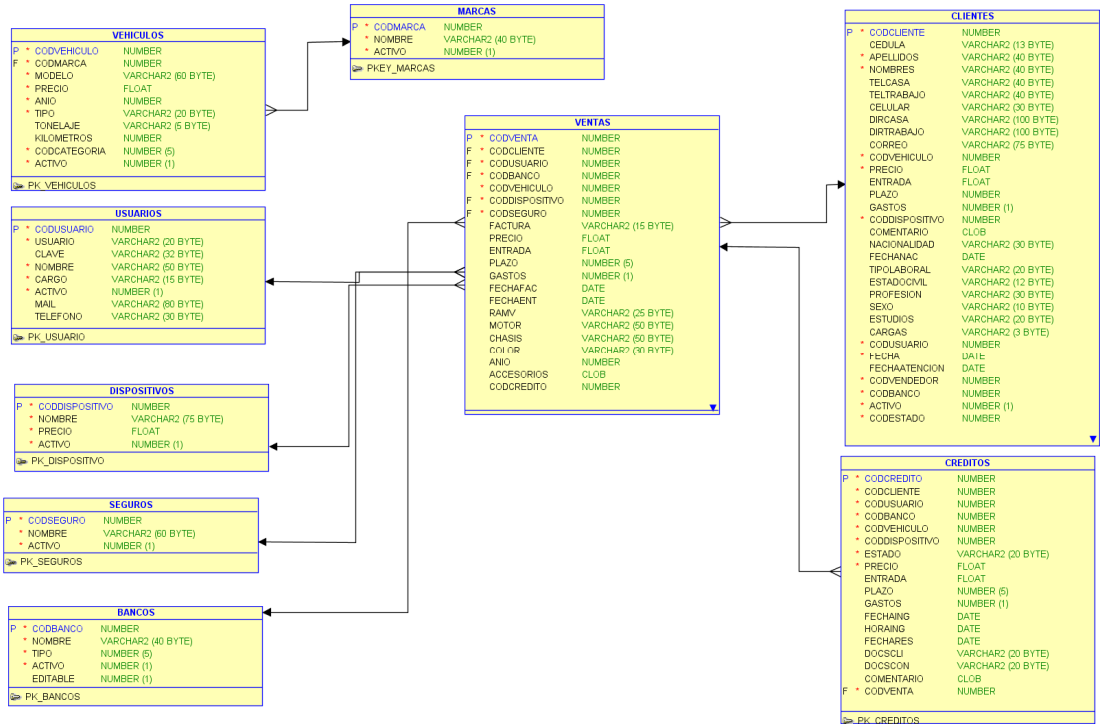
Módulo de Clientes



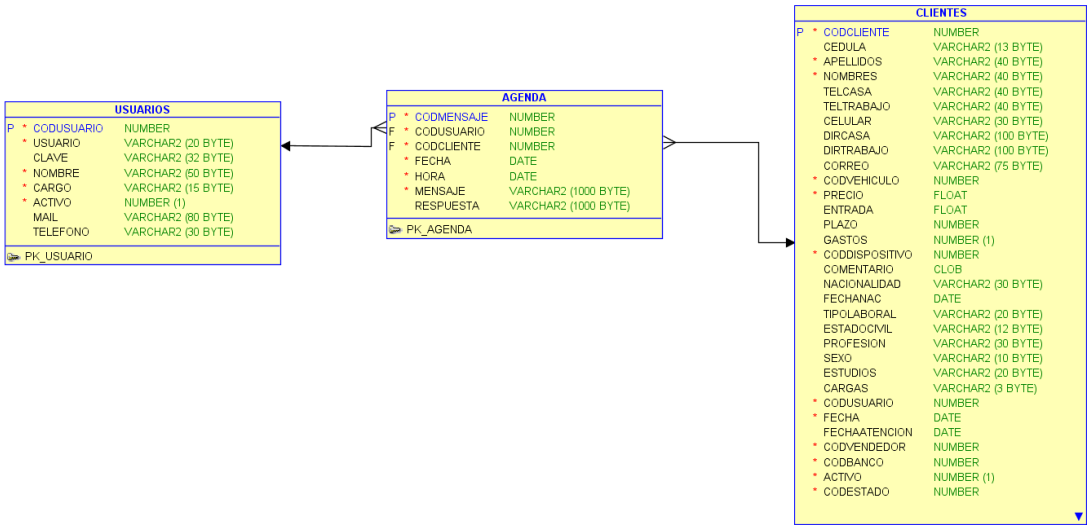
Módulo de Créditos



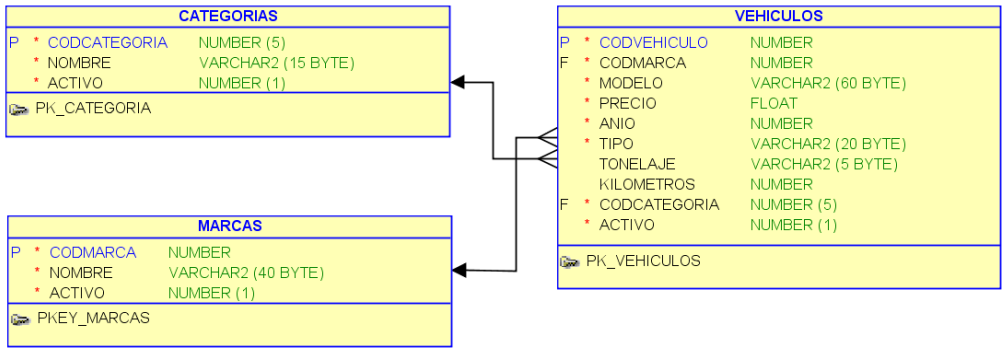
Módulo de Ventas



Módulo de Agenda



Módulo de vehículo



Módulo de Solicitud

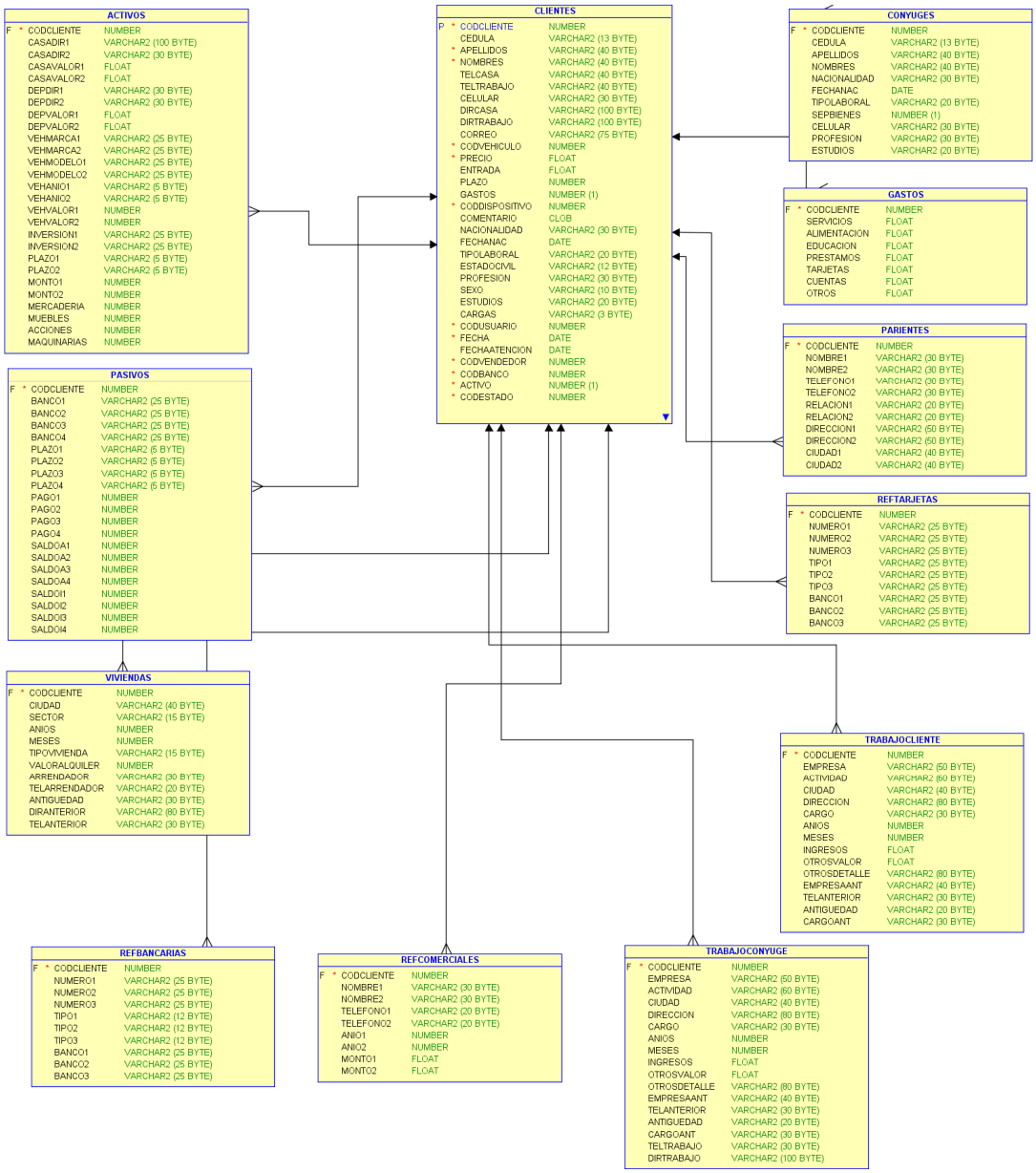
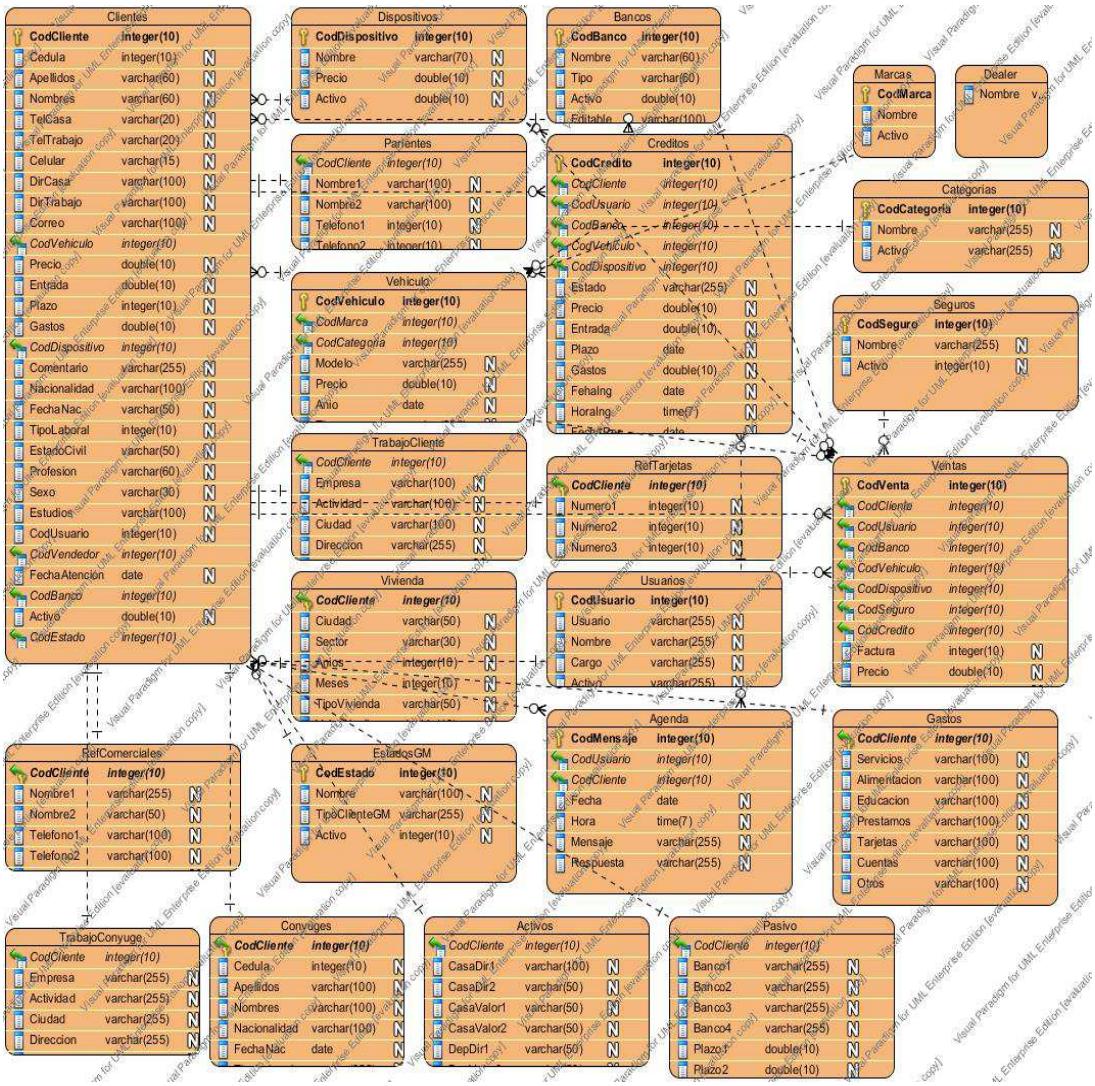


Diagrama General de Entidad Relación



Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

A continuación se especificará el modelo lógico de la base de datos del sistema Terrasoft mencionando cada entidad con sus respectivos atributos y claves principales y foráneas.

DISEÑO LÓGICO

ESQUEMA

Nuestro Sistema cuenta con un Usuario dueño del Esquema llamado CREDISOFT

Este Usuario principal debe tener los siguientes roles y privilegios:

```
-- Privilegios
grant connect to CREDISOFT with admin option;
grant dba to CREDISOFT with admin option;
-- Roles
grant alter user to CREDISOFT with admin option;
grant unlimited tablespace to CREDISOFT with admin option;
```

El mismo esta conformado por 4 TABLESPACES:

- CREDISOFT_DATA.- Para Datos
- CREDISOFT_INDEX.- Para Indices
- CREDISOFT_TEMP.- TableSpace Temporal
- CREDISOFT_UNDO.- TableSpace de Deshacer

SINONIMOS:

Debido a que otros usuarios se conectan a este Esquema, y no queremos quemar el Nombre del Esquema en las Consultas SQL cada Tabla, Vista, Paquete, Procedimiento Almacenado y Función, constituyen también un Sinónimo Publico con el mismo nombre del Objeto.

ROLES

Nuestro Sistema se basa en Usuarios Administradores (Jefes y Gerentes) y Vendedores, por ende dentro de la Base de Datos encontramos 2 Roles justamente para estos Cargos que son **credi_administrador** y **credi_vendedor**.

Los mismos cuentan con los siguientes permisos:

CREDI_ADMINISTRADOR:

```
-- ROLES
create role CREDI_ADMINISTRADOR;
-- PRIVILEGIOS
grant select, insert, update on ACTIVOS to CREDI_ADMINISTRADOR;
grant select, insert, update on AGENDA to CREDI_ADMINISTRADOR;
grant select, insert, update on BANCOS to CREDI_ADMINISTRADOR;
grant select, insert, update on CATEGORIAS to CREDI_ADMINISTRADOR;
```

```
grant select, insert, update on CLIENTES to CREDI_ADMINISTRADOR;
grant select, insert, update on CONYUGES to CREDI_ADMINISTRADOR;
grant select, insert, update on CREDITOS to CREDI_ADMINISTRADOR;
grant select, insert, update on DEALER to CREDI_ADMINISTRADOR;
grant select, insert, update on DISPOSITIVOS to CREDI_ADMINISTRADOR;
grant select, insert, update on ESTADOSGM to CREDI_ADMINISTRADOR;
grant execute on FN_CLIENTE_EXISTE to CREDI_ADMINISTRADOR;
grant select, insert, update on GASTOS to CREDI_ADMINISTRADOR;
grant select, insert, update on MARCAS to CREDI_ADMINISTRADOR;
grant select, insert, update on PARIENTES to CREDI_ADMINISTRADOR;
grant select, insert, update on PASIVOS to CREDI_ADMINISTRADOR;
grant execute on PK_AGENDA to CREDI_ADMINISTRADOR;
grant execute on PK_BANCOS to CREDI_ADMINISTRADOR;
grant execute on PK_CLIENTES to CREDI_ADMINISTRADOR;
grant execute on PK_CREDITOS to CREDI_ADMINISTRADOR;
grant execute on PK_DEALER to CREDI_ADMINISTRADOR;
grant execute on PK_DISPOSITIVOS to CREDI_ADMINISTRADOR;
grant execute on PK_ESTADOS to CREDI_ADMINISTRADOR;
grant execute on PK_LOGIN to CREDI_ADMINISTRADOR;
grant execute on PK_MARCAS to CREDI_ADMINISTRADOR;
grant execute on PK_REPORTES to CREDI_ADMINISTRADOR;
grant execute on PK_SEGUROS to CREDI_ADMINISTRADOR;
grant execute on PK_SOLICITUD to CREDI_ADMINISTRADOR;
grant execute on PK_USUARIOS to CREDI_ADMINISTRADOR;
grant execute on PK_UTILITARIOS to CREDI_ADMINISTRADOR;
grant execute on PK_VEHICULOS to CREDI_ADMINISTRADOR;
grant execute on PK_VENTAS to CREDI_ADMINISTRADOR;
grant select, insert, update on REFBANCARIAS to CREDI_ADMINISTRADOR;
grant select, insert, update on REFCOMERCIALES to CREDI_ADMINISTRADOR;
grant select, insert, update on REFTARJETAS to CREDI_ADMINISTRADOR;
grant select, insert, update on SEGUROS to CREDI_ADMINISTRADOR;
grant execute on SP_CAMBIARCLAVE to CREDI_ADMINISTRADOR;
grant execute on SP_DEALER to CREDI_ADMINISTRADOR;
grant select, insert, update on TMP_REP_CLIENTES to CREDI_ADMINISTRADOR;
grant select, insert, update on TRABAJOCLIENTE to CREDI_ADMINISTRADOR;
grant select, insert, update on TRABAJOCONYUGE to CREDI_ADMINISTRADOR;
grant select, insert, update on USUARIOS to CREDI_ADMINISTRADOR;
grant select, insert, update on VEHICULOS to CREDI_ADMINISTRADOR;
grant select, insert, update on VENTAS to CREDI_ADMINISTRADOR;
```

```
grant select on VIS_CLIENTES_ELIMINADOS to CREDI_ADMINISTRADOR;
grant select on VISCONSULAGENDA to CREDI_ADMINISTRADOR;
grant select on VISCONSULCLIENTES to CREDI_ADMINISTRADOR;
grant select on VISCONSULCREDITOS to CREDI_ADMINISTRADOR;
grant select on VISCONSULCREDITOS_CLIENTES to CREDI_ADMINISTRADOR;
grant select on VISCONSULVENTAS to CREDI_ADMINISTRADOR;
grant select on VISCONSULVENTAS_CLIENTES to CREDI_ADMINISTRADOR;
grant select on VISVEHICULOS to CREDI_ADMINISTRADOR;
grant select, insert, update on VIVIENDAS to CREDI_ADMINISTRADOR;
-- Grant/Revoke role privileges
grant connect to CREDI_ADMINISTRADOR with admin option;
-- Grant/Revoke system privileges
grant alter user to CREDI_ADMINISTRADOR with admin option;
grant create session to CREDI_ADMINISTRADOR with admin option;
```

CREDI_VENDEDOR

```
-- ROLES
create role CREDI_VENDEDOR;
-- PRIVILEGIOS
grant select, insert, update on ACTIVOS to CREDI_VENDEDOR;
grant select, insert, update on AGENDA to CREDI_VENDEDOR;
grant select on BANCOS to CREDI_VENDEDOR;
grant select on CATEGORIAS to CREDI_VENDEDOR;
grant select, insert, update on CLIENTES to CREDI_VENDEDOR;
grant select, insert, update on CONYUGES to CREDI_VENDEDOR;
grant select, insert, update on CREDITOS to CREDI_VENDEDOR;
grant select on DEALER to CREDI_VENDEDOR;
grant select on DISPOSITIVOS to CREDI_VENDEDOR;
grant select on ESTADOSGM to CREDI_VENDEDOR;
grant execute on FN_CLIENTE_EXISTE to CREDI_VENDEDOR;
grant select, insert, update on GASTOS to CREDI_VENDEDOR;
grant select on MARCAS to CREDI_VENDEDOR;
grant select, insert, update on PARIENTES to CREDI_VENDEDOR;
grant select, insert, update on PASIVOS to CREDI_VENDEDOR;
grant execute on PK_AGENDA to CREDI_VENDEDOR;
grant execute on PK_CLIENTES to CREDI_VENDEDOR;
grant execute on PK_CREDITOS to CREDI_VENDEDOR;
grant execute on PK_LOGIN to CREDI_VENDEDOR;
```

```
grant execute on PK_REPORTES to CREDI_VENDEDOR;
grant execute on PK_SOLICITUD to CREDI_VENDEDOR;
grant execute on PK_USUARIOS to CREDI_VENDEDOR;
grant execute on PK_UTILITARIOS to CREDI_VENDEDOR;
grant execute on PK_VENTAS to CREDI_VENDEDOR;
grant select, insert, update on REFBANCARIAS to CREDI_VENDEDOR;
grant select, insert, update on REFCOMERCIALES to CREDI_VENDEDOR;
grant select, insert, update on REFTARJETAS to CREDI_VENDEDOR;
grant select, insert, update on SEGUROS to CREDI_VENDEDOR;
grant execute on SP_CAMBIARCLAVE to CREDI_VENDEDOR;
grant select, insert, update on TMP_REP_CLIENTES to CREDI_VENDEDOR;
grant select, insert, update on TRABAJOCLIENTE to CREDI_VENDEDOR;
grant select, insert, update on TRABAJOCONYUGE to CREDI_VENDEDOR;
grant select on USUARIOS to CREDI_VENDEDOR;
grant select on VEHICULOS to CREDI_VENDEDOR;
grant select on VENTAS to CREDI_VENDEDOR;
grant select on VIS_CLIENTES_ELIMINADOS to CREDI_VENDEDOR;
grant select on VISCONSULAGENDA to CREDI_VENDEDOR;
grant select on VISCONSULCLIENTES to CREDI_VENDEDOR;
grant select on VISCONSULCREDITOS to CREDI_VENDEDOR;
grant select on VISCONSULCREDITOS_CLIENTES to CREDI_VENDEDOR;
grant select on VISCONSULVENTAS to CREDI_VENDEDOR;
grant select on VISCONSULVENTAS_CLIENTES to CREDI_VENDEDOR;
grant select on VISVEHICULOS to CREDI_VENDEDOR;
grant select, insert, update on VIVIENDAS to CREDI_VENDEDOR;
-- Grant/Revoke role privileges
grant connect to CREDI_VENDEDOR with admin option;
-- Grant/Revoke system privileges
grant alter user to CREDI_VENDEDOR with admin option;
grant create session to CREDI_VENDEDOR with admin option;
```

TABLA 1: Activos

Descripción y Uso.

La tabla **Activos** representa los activos que posee el cliente y que se llenaran en la solicitud de crédito.

Tamaño esperado y crecimiento

La tabla **Activos** tendrá como tamaño inicial 64Kb y crecimiento ilimitado.

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODCLIENTE	NUMBER	N		FK	Clientes.codcliente Relación de uno a uno
CASADIR1	VARCHAR2(100)	Y			Dirección de la casa 1
CASADIR2	VARCHAR2(30)	Y			Dirección de la casa 2
CASAVALOR1	FLOAT	Y			Valor de la casa 1
CASAVALOR2	FLOAT	Y			Valor de la casa 2
DEPDIR1	VARCHAR2(30)	Y			Dirección del Departamento 1
DEPDIR2	VARCHAR2(30)	Y			Dirección del Departamento 2
DEPVALOR1	FLOAT	Y			Valor del Departamento 1
DEPVALOR2	FLOAT	Y			Valor del Departamento 2
VEHMARCA1	VARCHAR2(25)	Y			Marca del Vehículo1
VEHMARCA2	VARCHAR2(25)	Y			Marca del Vehiculo2
VEHMODELO1	VARCHAR2(25)	Y			Modelo del Vehículo1
VEHMODELO2	VARCHAR2(25)	Y			Modelo del Vehículo2
VEHANIO1	VARCHAR2(5)	Y			Año del Vehiculo1
VEHANIO2	VARCHAR2(5)	Y			Año del Vehiculo2
VEHVALOR1	NUMBER	Y			Valor del Vehículo1
VEHVALOR2	NUMBER	Y			Valor del Vehículo2

INVERSION1	VARCHAR2(25)	Y			Inversión1
INVERSION2	VARCHAR2(25)	Y			Inversión2
PLAZO1	VARCHAR2(5)	Y			Plazo1
PLAZO2	VARCHAR2(5)	Y			Plazo2
MONTO1	NUMBER	Y			Valor del Monto1
MONTO2	NUMBER	Y			Valor del Monto2
MERCADERIA	NUMBER	Y			Valor en Mercaderías
MUEBLES	NUMBER	Y			Valor en muebles
ACCIONES	NUMBER	Y			Valor en Acciones
MAQUINARIAS	NUMBER	Y			Valor en maquinarias

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 2: Agenda

Descripción y Uso.

La tabla **Agenda** almacenará todos los recordatorios de un usuario con respecto al cliente.

Tamaño esperado y crecimiento

La tabla **Agenda** tendrá como tamaño inicial 64Kb y crecimiento ilimitado.

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODMENSAJE	NUMBER	N	X		Código del mensaje

CODUSUARIO	NUMBER	N		FK	Usuarios, codusuario A que usuario pertenece el Mensaje
CODCLIENTE	NUMBER	N		FK	Clientes.codcliente A que cliente pertenece el Mensaje
FECHA	DATE	Y			Fecha del Recordatorio
HORA	DATE	Y			Hora
MENSAJE	CLOB	Y			Descripción del mensaje
RESPUESTA	CLOB	Y			Respuesta del mensaje

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 3: Bancos

Descripción y Uso.

La tabla **Bancos** almacenará los datos referentes a una entidad bancaria.

Tamaño esperado y crecimiento

La tabla **Bancos** tendrá como tamaño inicial 64Kb y crecimiento ilimitado.

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODBANCO	NUMBER	N	X		Código del Banco
NOMBRE	VARCHAR2(40)	N			Nombre del Banco
TIPO	NUMBER(5)	N			Si es 1=Banco,2=Crédito

					directo,3=Contado
ACTIVO	NUMBER(1)	N		Activo IN (1,0)	Si es 0 significa que el registro está eliminado lógicamente, caso contrario está activo o se lo puede utilizar.
EDITABLE	NUMBER(1)	Y		Editable IN (1,0)	Si es 0 significa que ningún usuario podrá realizar modificaciones o eliminaciones de ese registro

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 4: Categorias

Descripción y Uso

Es una tabla de configuración que almacena las categorías de los vehículos que comercializa el concesionario como por ejemplo: camioneta, jeep, sedan, etc.

Tamaño esperado y crecimiento

La tabla **Categorias** tendrá como tamaño inicial 64Kb y crecimiento ilimitado.

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODCATEGORIA	NUMBER(5)	N	X		Código de la categoría
NOMBRE	VARCHAR2(15)	N			Nombre de la categoría
ACTIVO	NUMBER(1)	N		Activo IN (1,0)	Si es 0 significa que el registro está eliminado lógicamente, caso

					contrario está activo o se lo puede utilizar.

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 5: Clientes

Descripción y Uso.

La tabla **Clientes** almacenará los datos de los clientes registrados a registrarse en el sistema.

Tamaño esperado y crecimiento

La tabla **Clientes** tendrá como tamaño inicial 64Kb y crecimiento ilimitado.

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODCLIENTE	NUMBER	N	X		Código del cliente
CEDULA	VARCHAR2(13)	Y		Unique	Numero de cédula del cliente
APELLIDOS	VARCHAR2(40)	N			Apellidos del cliente
NOMBRES	VARCHAR2(40)	N			Nombres del cliente
TELCASA	VARCHAR2(40)	Y			Teléfono de casa
TELTRABAJO	VARCHAR2(40)	Y			Teléfono del Trabajo
CELULAR	VARCHAR2(30)	Y			Número de teléfono Móvil
DIRCASA	VARCHAR2(100)	Y			Dirección de la casa
DIRTRABAJO	VARCHAR2(100)	Y			Dirección del trabajo
CORREO	VARCHAR2(75)	Y			Dirección del correo

					electrónico
CODVEHICULO	NUMBER	N		FK	Vehiculos.codvehiculo Vehiculo por default en que el cliente está interesado
PRECIO	FLOAT	N		Precio > 0	Precio del vehículo
ENTRADA	FLOAT	Y			Valor de la entrada
PLAZO	NUMBER	Y			Cantidad del plazo. Se refiere al Plazo del Crédito
GASTOS	NUMBER(1)	Y			Si es 1=Financiados,0=No financiados. Indica si los Gastos Bancarios van financiados
CODDISPOSITIVO	NUMBER	Y		FK	Dispositivos.coddispositivo Indica si el Crédito incluye un Dispositivo de Seguidad Financiado
COMENTARIO	CLOB	Y			Comentario acerca del cliente
NACIONALIDAD	VARCHAR2(30)	Y			Nacionalidad del cliente
FECHANAC	DATE	Y			Fecha de nacimiento del cliente
TIPOLABORAL	VARCHAR2(20)	Y			Tipo de relación Laboral (Dependiente, Independiente, Jubilado, etc)

ESTADOCIVIL	VARCHAR2(12)	Y			Estado civil del cliente
PROFESION	VARCHAR2(30)	Y			Profesión del cliente
SEXO	VARCHAR2(10)	Y			Sexo del cliente
ESTUDIOS	VARCHAR2(20)	Y			Estudios realizados por el cliente (Secundarios, Superior, Postgrado, etc)
CARGAS	VARCHAR2(3)	Y			Nº de Cargas que posee el cliente
CODUSUARIO	NUMBER	N		FK	Usuarios.codusuario Usuario que lo ingresó al Sistema
FECHA	DATE	N			Fecha de Ingreso al Sistema
FECHAATENCION	DATE	N			Fecha de atención al cliente
CODVENDEDOR	NUMBER	N		FK	Usuarios.codusuario Vendedor que atendió al Cliente
CODBANCO	NUMBER	Y		FK	Bancos.codbanco Banco o Forma de Pago por default en que el cliente desea comprar
ACTIVO	NUMBER(1)	N		Activo IN (1,0)	Si es 0 significa que el registro está eliminado lógicamente, caso contrario está activo o se lo puede

					utilizar.

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 6: Conyuges

Descripción y Uso.

La tabla **Conyuges** guardará los datos del cónyuge de un cliente.

Tamaño esperado y crecimiento

La tabla **Conyuges** tendrá como tamaño inicial 64Kb y crecimiento ilimitado

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODCLIENTE	NUMBER	N		FK	Clientes.codcliente
CEDULA	VARCHAR2(13)	Y			Número de cédula del cónyuge
APELLIDOS	VARCHAR2(40)	Y			Apellidos del cónyuge del cliente
NOMBRES	VARCHAR2(40)	Y			Nombres del cónyuge
NACIONALIDAD	VARCHAR2(30)	Y			Nacionalidad del cónyuge
FECHANAC	DATE	Y			Fecha de nacimiento del cónyuge
TIPOLABORAL	VARCHAR2(20)	Y			Tipo de actividad laboral (Dependiente,

					Independiente, etc)
SEPBIENES	NUMBER(1)	Y			1 = Si , 0 = No
CELULAR	VARCHAR2(30)	Y			Numero de teléfono móvil
PROFESION	VARCHAR2(30)	Y			Profesión del cónyuge
ESTUDIOS	VARCHAR2(20)	Y			Estudios realizados del cónyuge (Secundarios, Postgrado, etc)

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 7: Creditos

Descripción y Uso.

La tabla **Creditos** guardará los créditos de un cliente

Tamaño esperado y crecimiento

La tabla **Creditos** tendrá como tamaño inicial 64Kb y crecimiento ilimitado

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODCREDITO	NUMBER	N	X		Código del crédito
CODCLIENTE	NUMBER	N		FK	Clientes.codcliente
CODUSUARIO	NUMBER	N		FK	Usuarios.codusuario
CODBANCO	NUMBER	N		FK	Bancos.codbanco
CODVEHICULO	NUMBER	N		FK	Vehiculos.codvehiculo
CODDISPOSITIVO	NUMBER	N		FK	Dispositivos.coddispositivo
ESTADO	VARCHAR2(20)	N			Aprobado, Negado, Sin

					respuesta
PRECIO	FLOAT	N		Precio > 0	Precio del vehículo
ENTRADA	FLOAT	Y			Valor de la entrada del vehículo
PLAZO	NUMBER(5)	Y			Plazo a diferenciar el valor del vehiculo
GASTOS	NUMBER(1)	Y			Gastos Booleano 1= Gastos Financiados, 0=No Financiados
FECHAING	DATE	Y			Fecha ingreso del crédito
HORAING	DATE	Y			Hora de ingreso del crédito
FECHARES	DATE	Y			Fecha respuesta del crédito
DOCSCLI	VARCHAR2(20)	Y			Documentos del cliente
DOCSCON	VARCHAR2(20)	Y			Documentos del cónyuge del cliente
COMENTARIO	CLOB	Y			Comentarios acerca al crédito

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 8: Dealer

Descripción y Uso.

La tabla **Dealer** guardará el nombre del concesionario que está utilizando el sistema.

Tamaño esperado y crecimiento

La tabla **Dealer** tendrá como tamaño inicial 64Kb y crecimiento ilimitado.

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
NOMBRE	VARCHAR2(40)	Y			Nombre del Concesionario

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 9: Dispositivos

Descripción y Uso.

La tabla **Dispositivos** guardará los datos de los dispositivos que se pueden financiar en un crédito.

Tamaño esperado y crecimiento

La tabla **Dispositivos** tendrá como tamaño inicial 64Kb y crecimiento ilimitado

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODDISPOSITIVO	NUMBER	N	X		Código del dispositivo
NOMBRE	VARCHAR2(75)	N			Nombre del dispositivo
PRECIO	FLOAT	N			Precio del dispositivo
ACTIVO	NUMBER(1)	N		Activo IN (1,0)	Si es 0 significa que el registro está eliminado lógicamente, caso contrario está activo o se lo puede utilizar.

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 10: Gastos

Descripción y Uso.

La tabla **Gastos** guarda la información de los gastos de un cliente que va a llenarse en la solicitud de crédito

Tamaño esperado y crecimiento

La tabla **Gastos** tendrá como tamaño inicial 64Kb y crecimiento ilimitado

COLUMNA	**TIPO DE DATO**	**NULO**	**PK**	**CHECK**	**COMENTARIO**
CODCLIENTE	NUMBER	N		FK	Cliente.codcliente
SERVICIOS	FLOAT	Y			Valor de gastos en servicios
ALIMENTACION	FLOAT	Y			Valor de gastos en alimentación
EDUCACION	FLOAT	Y			Valor de gastos en educación
PRESTAMOS	FLOAT	Y			Valor de pagos en préstamos
TARJETAS	FLOAT	Y			Valor de pagos de tarjetas.
CUENTAS	FLOAT	Y			Valor en cuentas
OTROS	FLOAT	Y			Otros gastos

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 11: Marcas

Descripción y Uso.

La tabla **Marcas** guardará la información de las marcas de automotores que comercializa el concesionario.

Tamaño esperado y crecimiento

La tabla **Marcas** tendrá como tamaño inicial 64Kb y crecimiento ilimitado

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODMARCA	NUMBER	N	X		Código de la marca del vehículo
NOMBRE	VARCHAR2(40)	N			Nombre de la marca
ACTIVO	NUMBER(1)	N		Activo IN (1,0)	Si es 0 significa que el registro está eliminado lógicamente, caso contrario está activo o se lo puede utilizar.

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 12: Parientes

Descripción y Uso.

La tabla **Parientes** guardará la información de los parientes que va a llenarse en la solicitud de crédito

Tamaño esperado y crecimiento

La tabla **Parientes** tendrá como tamaño inicial 64Kb y crecimiento ilimitado

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODCLIENTE	NUMBER	N		FK	Clientes.codcliente
NOMBRE1	VARCHAR2(30)	Y			Nombre del pariente1
NOMBRE2	VARCHAR2(30)	Y			Nombre del pariente2
TELEFONO1	VARCHAR2(30)	Y			Número de teléfono del pariente1
TELEFONO2	VARCHAR2(30)	Y			Número de teléfono del pariente2
RELACION1	VARCHAR2(20)	Y			Tipo de relación de parentezco1
RELACION2	VARCHAR2(20)	Y			Tipo de relación de parentezco2
DIRECCION1	VARCHAR2(50)	Y			Dirección del pariente1
DIRECCION2	VARCHAR2(50)	Y			Dirección del pariente2
CIUDAD1	VARCHAR2(40)	Y			Ciudad del pariente1
CIUDAD2	VARCHAR2(40)	Y			Ciudad del pariente2

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 13: Pasivos

Descripción y Uso.

La tabla **Pasivos** guardará la información de todos los pasivos, deudas y obligaciones bancarias del cliente a llenarse en la solicitud de crédito

Tamaño esperado y crecimiento

La tabla **Pasivos** tendrá como tamaño inicial 64Kb y crecimiento ilimitado.

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODCLIENTE	NUMBER	N		FK	Clientes.codcliente
BANCO1	VARCHAR2(25)	Y			Nombre del Banco 1 que tiene obligación
BANCO2	VARCHAR2(25)	Y			Nombre del Banco 2 que tiene obligación
BANCO3	VARCHAR2(25)	Y			Nombre del Banco 3 que tiene obligación
BANCO4	VARCHAR2(25)	Y			Nombre del Banco 4 que tiene obligación
PLAZO1	VARCHAR2(5)	Y			Plazo otorgado con el banco1
PLAZO2	VARCHAR2(5)	Y			Plazo otorgado con el banco2
PLAZO3	VARCHAR2(5)	Y			Plazo otorgado con el banco3
PLAZO4	VARCHAR2(5)	Y			Plazo otorgado con el banco4
PAGO1	NUMBER	Y			Valor de la Cuota con el Banco1
PAGO2	NUMBER	Y			Valor de la Cuota con el Banco2

PAGO3	NUMBER	Y			Valor de la Cuota con el Banco 3
PAGO4	NUMBER	Y			Valor de la Cuota con Banco 4
SALDOA1	NUMBER	Y			Saldo Actual de la Deuda en Bco1
SALDOA2	NUMBER	Y			Saldo Actual de la Deuda en Bco2
SALDOA3	NUMBER	Y			Saldo Actual de la Deuda en Bco3
SALDOA4	NUMBER	Y			Saldo Actual de la Deuda en Bco4
SALDOI1	NUMBER	Y			Monto Inicial de la Deuda en Bco1
SALDOI2	NUMBER	Y			Monto Inicial de la Deuda en Bco2
SALDOI3	NUMBER	Y			Monto Inicial de la Deuda en Bco3
SALDOI4	NUMBER	Y			Monto Inicial de la Deuda en Bco4

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 14: RefBancarias

Descripción y Uso.

La tabla **RefBancarias** guardará la información de todas las referencias bancarias del cliente a llenarse dentro de la solicitud de crédito (cuentas corrientes y ahorros)

Tamaño esperado y crecimiento

La tabla **RefBancarias** tendrá como tamaño inicial 64Kb y crecimiento ilimitado.

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODCLIENTE	NUMBER	N		FK	Clientes.codcliente
NUMERO1	VARCHAR2(25)	Y			Número de la cuenta bancaria1
NUMERO2	VARCHAR2(25)	Y			Número de la cuenta bancaria2
NUMERO3	VARCHAR2(25)	Y			Número de la cuenta bancaria3
TIPO1	VARCHAR2(12)	Y			Tipo de cuenta bancaria1
TIPO2	VARCHAR2(12)	Y			Tipo de cuenta bancaria2
TIPO3	VARCHAR2(12)	Y			Tipo de cuenta bancaria3
BANCO1	VARCHAR2(25)	Y			Nombre del banco de la cuenta1
BANCO2	VARCHAR2(25)	Y			Nombre del banco de la cuenta2
BANCO3	VARCHAR2(25)	Y			Nombre del banco de la cuenta3

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 15: RefComerciales

Descripción y Uso.

La tabla **RefComerciales** guardará la información de todas las referencias comerciales que tiene el cliente (Compras realizadas en Casas Comerciales).

Tamaño esperado y crecimiento

La tabla **RefComerciales** tendrá como tamaño inicial 64Kb y crecimiento ilimitado.

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODCLIENTE	NUMBER	N		FK	Clientes.codcliente
NOMBRE1	VARCHAR2(30)	Y			Nombre del local comercial1
NOMBRE2	VARCHAR2(30)	Y			Nombre del local comercial2
TELEFONO1	VARCHAR2(20)	Y			Teléfono del comercial1
TELEFONO2	VARCHAR2(20)	Y			Teléfono del comercial2
ANIO1	NUMBER	Y			Año que realizó la compra1
ANIO2	NUMBER	Y			Año que realizó la compra2
MONTO1	FLOAT	Y			Monto de la compra1
MONTO2	FLOAT	Y			Monto de la compra2

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 16: RefTarjetas

Descripción y Uso.

La tabla **RefTarjetas** guardará la información de todas las tarjetas de crédito que posee el cliente a llenarse dentro de la solicitud de crédito.

Tamaño esperado y crecimiento

La tabla **RefTarjetas** tendrá como tamaño inicial 64Kb y crecimiento ilimitado.

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODCLIENTE	NUMBER	N		FK	Clientes.codcliente
NUMERO1	VARCHAR2(25)	Y			Numero de la tarjeta de crédito1
NUMERO2	VARCHAR2(25)	Y			Numero de la tarjeta de crédito2
NUMERO3	VARCHAR2(25)	Y			Número de la tarjeta de crédito3
TIPO1	VARCHAR2(25)	Y			Tipo de tarjeta de crédito1 (Ej: Amercian Express, Visa, Mastercard)
TIPO2	VARCHAR2(25)	Y			Tipo de tarjeta de crédito2
TIPO3	VARCHAR2(25)	Y			Tipo de tarjeta de crédito3
BANCO1	VARCHAR2(25)	Y			Banco de relación de la tarjeta1
BANCO2	VARCHAR2(25)	Y			Banco de relación de la tarjeta2

BANCO3	VARCHAR2(25)	Y			Banco de relación de la tarjeta3

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 17: Seguros

Descripción y Uso.

La tabla **Seguros** guardará la información de las compañías aseguradoras con las que puede asegurar un vehículo en el momento de registrar una venta.

Tamaño esperado y crecimiento

La tabla **Seguros** tendrá como tamaño inicial 64Kb y crecimiento ilimitado.

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODSEGURO	NUMBER	N	X		Código del Seguro
NOMBRE	VARCHAR2(60)	N			Nombre del Seguro
ACTIVO	NUMBER(1)	N		Activo IN (1,0)	Si es 0 significa que el registro está eliminado lógicamente, caso contrario está activo o se lo puede utilizar.

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 18: TrabajoCliente

Descripción y Uso.

La tabla **TrabajoCliente** guardará todo el historial laboral del cliente

Tamaño esperado y crecimiento

La tabla **TrabajoCliente** tendrá como tamaño inicial 64Kb y crecimiento ilimitado.

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODCLIENTE	NUMBER	N		FK	Clientes.codcliente
EMPRESA	VARCHAR2(50)	Y			Nombre de la empresa donde trabajo
ACTIVIDAD	VARCHAR2(60)	Y			Actividad que desempeña la empresa
CIUDAD	VARCHAR2(40)	Y			Ciudad donde está ubicado el trabajo
DIRECCION	VARCHAR2(80)	Y			Dirección del trabajo del Cliente
CARGO	VARCHAR2(30)	Y			Cargo que desempeña el cliente
ANIOS	NUMBER	Y			Años en el cargo
MESES	NUMBER	Y			Meses en el cargo
INGRESOS	FLOAT	Y			Ingresos del cliente

OTROSVALOR	FLOAT	Y			Otros valores del cliente
OTROSDETALLE	VARCHAR2(80)	Y			De donde provienen los Otros Ingresos
EMPRESAANT	VARCHAR2(40)	Y			Nombre de la empresa anterior
TELANTERIOR	VARCHAR2(30)	Y			Teléfono de la empresa anterior
ANTIGUEDAD	VARCHAR2(20)	Y			Antigüedad en dicha empresa
CARGOANT	VARCHAR2(30)	Y			Cargo anterior que desempeñó

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 19: TrabajoConyuge

Descripción y Uso.

La tabla **TrabajoClonyuge** guardará todo el historial laboral del cliente

Tamaño esperado y crecimiento

La tabla **TrabajoConyuge** tendrá como tamaño inicial 64Kb y crecimiento ilimitado

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODCLIENTE	NUMBER	N		FK	Clientes.codcliente
EMPRESA	VARCHAR2(50)	Y			Empresa en que labora el cónyuge

ACTIVIDAD	VARCHAR2(60)	Y			Actividad que realiza la Empresa
CIUDAD	VARCHAR2(40)	Y			Ciudad actual trabajo del cónyuge
DIRECCION	VARCHAR2(80)	Y			Dirección del trabajo del cónyuge
CARGO	VARCHAR2(30)	Y			Cargo del cónyuge
ANIOS	NUMBER	Y			Años que desempeño el cargo
MESES	NUMBER	Y			Y Meses que desempeñó el cargo
INGRESOS	FLOAT	Y			Ingresos del cónyuge
OTROSVALOR	FLOAT	Y			Otros valores de ingresos
OTROSDETALLE	VARCHAR2(80)	Y			De donde provienen los Otros Ingresos
EMPRESAANT	VARCHAR2(40)	Y			Empresa anterior
TELANTERIOR	VARCHAR2(30)	Y			Teléfono anterior
ANTIGUEDAD	VARCHAR2(20)	Y			Antigüedad en el trabajo
CARGOANT	VARCHAR2(30)	Y			Cargo que ejerció
TELTRABAJO	VARCHAR2(30)	Y			Teléfono del trabajo anterior
DIRTRABAJO	VARCHAR2(100)	Y			Dirección del

					trabajo ant.

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 20: Usuarios

Descripción y Uso.

La tabla **Usuarios** guardará toda la información del usuario que utilice el sistema.

Tamaño esperado y crecimiento

La tabla **Usuarios** tendrá como tamaño inicial 64Kb y crecimiento ilimitado.

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODU*SUARIO	NUMBER	N	X		Código del usuario
USUARIO	VARCHAR2(20)	N		Unique	Usuario del sistema
CLAVE	VARCHAR2(15)	N			Clave del usuario
NOMBRE	VARCHAR2(50)	N			Nombre del usuario
CARGO	VARCHAR2(15)	N			Vendedor, Administrador
ACTIVO	NUMBER(1)	N		Activo IN (1,0)	Si es 0 significa que el registro está eliminado lógicamente, caso contrario está activo o se lo puede utilizar.

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 21: Vehiculos

Descripción y Uso.

La tabla **Vehiculos** guardará toda la información de los vehículos existentes en el sistema

Tamaño esperado y crecimiento

La tabla **Vehiculos** tendrá como tamaño inicial 64Kb y crecimiento ilimitado.

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODVEHICULO	NUMBER	N	X		Código del Vehículo
CODMARCA	NUMBER	N		FK	Marcas.CodMarca
MODELO	VARCHAR2(60)	N			Modelo del Vehículo
PRECIO	FLOAT	N		Precio > 0	Precio del Vehículo
ANIO	NUMBER	N			Año del Vehículo
TIPO	VARCHAR2(20)	N			Liviano, Usado, Pesado
TONELAJE	VARCHAR2(5)	Y			Tonelaje del camión
KILOMETROS	NUMBER	Y			Kilometraje del vehículo
CODCATEGORIA	NUMBER(5)	N		FK	Categorias.codcategoria
ACTIVO	NUMBER(1)	N		Activo IN (1,0)	Si es 0 significa que el registro está eliminado lógicamente, caso contrario está activo o se lo puede utilizar.

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 22: Ventas

Descripción y Uso.

La tabla **Ventas** almacenará la información de todas las ventas que realiza un determinado vendedor

Tamaño esperado y crecimiento

La tabla **Ventas** tendrá como tamaño inicial 64Kb y crecimiento ilimitado.

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODVENTA	NUMBER	N	X		Código de la venta
CODCLIENTE	NUMBER	N		FK	Clientes.codcliente
CODUSUARIO	NUMBER	N		FK	Usuarios.codusuario
CODBANCO	NUMBER	N		FK	Bancos.codbanco
CODVEHICULO	NUMBER	N		FK	Vehiculos.codvehiculo
CODDISPOSITIVO	NUMBER	N		FK	Dispositovos.coddispositivo
CODSEGURO	NUMBER	N		FK	Seguros.codseguro
FACTURA	VARCHAR2(15)	Y			Nº de factura
PRECIO	FLOAT	Y		Precio > 0	Precio del vehiculo
ENTRADA	FLOAT	Y			Valor de entrada del vehiculo
PLAZO	NUMBER(5)	Y			Plazo a vender el vehiculo

GASTOS	NUMBER(1)	Y			Gastos
FECHAFAC	DATE	Y			Fecha actual
FECHAENT	DATE	Y			Fecha de entrega
RAMV	VARCHAR2(25)	Y			Registro Aduanero de Matriculación Vehicular
MOTOR	VARCHAR2(50)	Y			Número de motor
CHASIS	VARCHAR2(50)	Y			Número de chasis
COLOR	VARCHAR2(30)	Y			Color del vehiculo
ANIO	NUMBER	Y			Año del vehiculo
ACCESORIOS	CLOB	Y			Accesorios adicionales
CODCREDITO	NUMBER	Y		FK	Creditos.codcredito

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

TABLA 23: Viviendas

Descripción y Uso.

La tabla **Viviendas** almacenará la información referente a la o las viviendas que posea el cliente.

Tamaño esperado y crecimiento

La tabla **Viviendas** tendrá como tamaño inicial 64Kb y crecimiento ilimitado.

COLUMNA	TIPO DE DATO	NULO	PK	CHECK	COMENTARIO
CODCLIENTE	NUMBER	N		FK	Clientes.codcliente
CIUDAD	VARCHAR2(40)	Y			Ciudad donde está la

					vivienda
SECTOR	VARCHAR2(15)	Y			Sector de ubicación de la Viv.
ANIOS	NUMBER	Y			Años que tiene la vivienda
MESES	NUMBER	Y			Meses que tiene la vivienda
TIPOVIVIENDA	VARCHAR2(15)	Y			Tipo de vivienda
VALORALQUILER	NUMBER	Y			Valor del alquiler
ARRENDADOR	VARCHAR2(30)	Y			Nombre del arrendador
TELARRENDADOR	VARCHAR2(20)	Y			Teléfono del arrendador
ANTIGUEDAD	VARCHAR2(30)	Y			Antigüedad
DIRANTERIOR	VARCHAR2(80)	Y			Dirección ant. Del arrendador
TELANTERIOR	VARCHAR2(30)	Y			Telef. anterior del arrendador

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS

Flujo de Pantallas

El Sistema se divide en 4 Módulos:

Módulo	Descripción

Seguridad	• Mediante el cual se pueden crear usuarios, los cuales se dividen en Administradores y Vendedores. • Los Administradores tienen acceso a todas las Pantallas mientras que los Vendedores solo a Registrar Clientes y Transacciones como Créditos, Ventas y Recordatorios • Incluye la Pantalla de Ingreso al Sistema, mediante Autenticación de Usuario y Contraseña • Así como la Pantalla para cambiar su Propia Clave. • También incluye una Papelera de Reciclaje donde se almacenan todos los Objetos Borrados de las Tablas (El borrado es lógico): La Papelera no incluye transacciones como créditos, recordatorios y ventas; solo Clientes, Dispositivos de Seguridad, Vehículos, Bancos y Cias. De Seguros Una vez marcadas como eliminadas, el Sistema simplemente no los mostrará para Futuras Transacciones hasta que se los restaure
Clientes	En este Módulo registramos los Clientes que ingresan al Concesionario así como el Vehículo y Forma de Pago en que están interesados. También se pueden registrar: • Créditos • Recordatorios (Citas, Llamadas, Seguimientos en General) • Ventas • Solicitud de Crédito
Configuración	Aquí se pueden parametrizar todos los objetos que forman parte del Sistema como

	Dispositivos de Seguridad • Vehículos (Modelo, Precio, etc) • Marcas de Vehículos que tiene el Concesionario • Bancos que pueden dar Créditos • Compañías de Seguros con las que se puede asegurar un Vehículo
Consultas	Módulo de Consultas y Reportes del Sistema Los Vendedores solo pueden ver su Información, mientras que los Administradores pueden ver la Información de todos los vendedores

Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

Flujo de Pantallas Módulo Seguridad

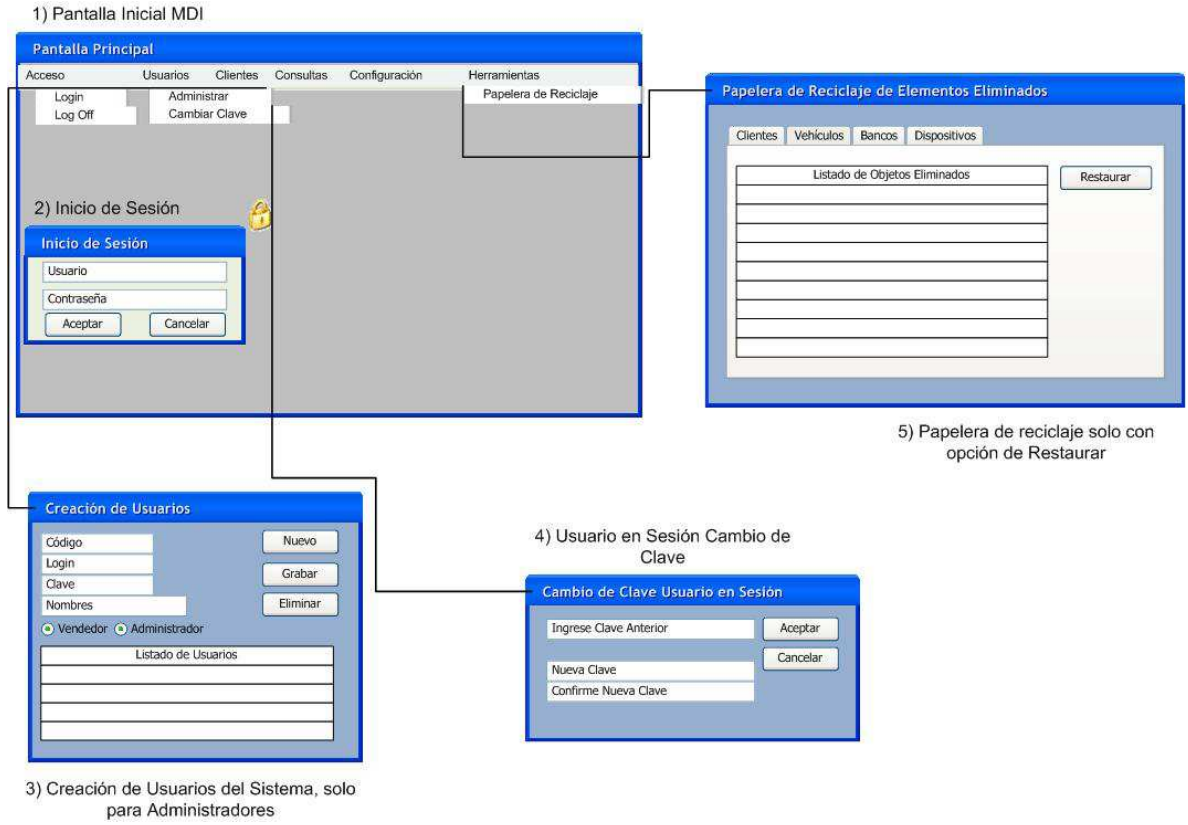
1) Se debe iniciar Sesión (2)

2) En caso de ser administrador se puede:

a. Acceder a la Pantalla de Creación, Eliminación y Modificación de Usuarios (3)

3) También se puede acceder a la Pantalla de Reciclaje (5)

4) Una vez iniciada sesión el Usuario puede acceder a Cambiar su clave ingresando su clave anterior y la nueva clave con su respectiva confirmación (4)



Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

Flujo de Acceso:

- Se ingresa a cada Pantalla de Forma individual desde el Menú Principal

Flujo de Pantallas Módulo de Clientes

1) Desde esta Pantalla podemos ingresar, modificar y eliminar clientes

2) Se ingresan los datos Básicos del Cliente

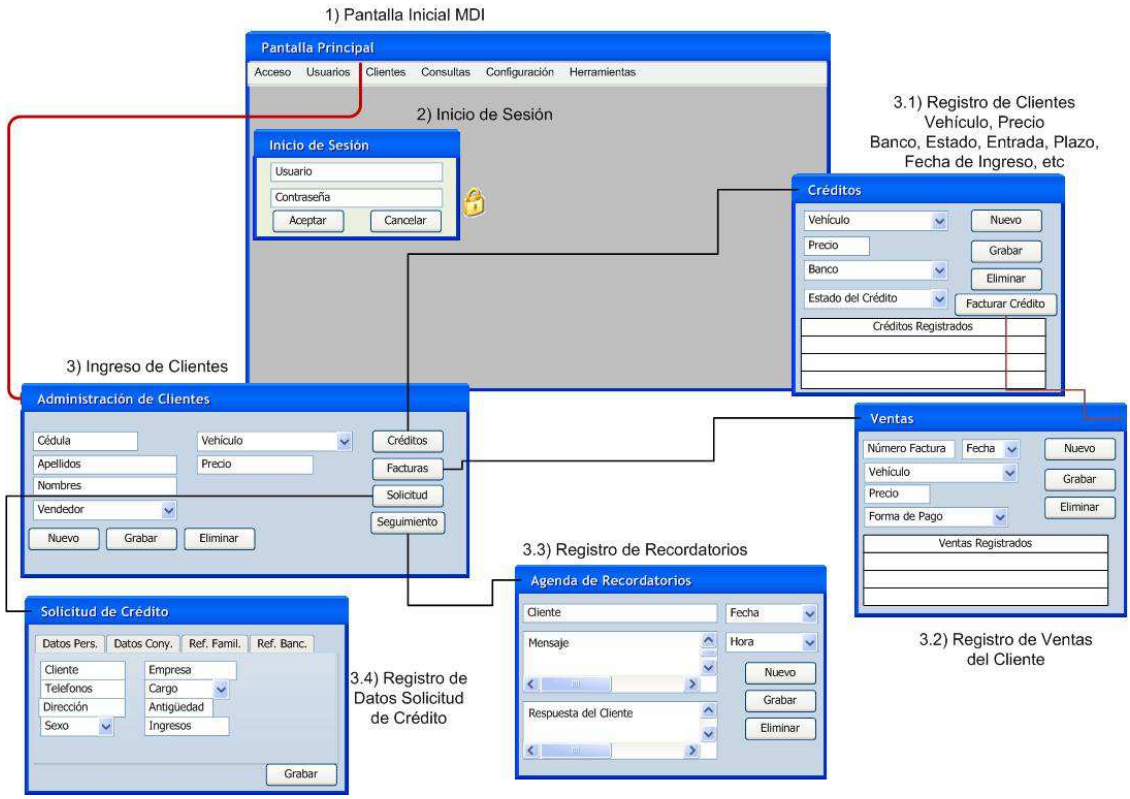
3) Se ingresa la Cotización, o sea los datos del vehículo y la forma del Pago que desea el Cliente

4) Se pueden realizar consulta de Clientes y seleccionar un Cliente para Modificación de Datos y Eliminación

5) Presenta 4 accesos directos para:

   ✓ Ingresar Créditos al Cliente (Banco, Entrada, Plazo, Estado del Crédito, Observaciones, etc)

- ✓ Registrar una Venta al Cliente (Forma de Pago, Vehículo, etc). La venta también puede ser hecha en base a un Crédito Aprobado
- ✓ Registrar Recordatorios para asistir a Citas, realizar llamadas, o realizar cualquier tipo de seguimiento
- ✓ Registrar todos los datos de un cliente que intervienen en una Solicitud de Crédito como:
  - ❖ Datos Laborales
  - ❖ Datos del Cónyuge
  - ❖ Referencias Familiares, Bancarias y Comerciales
  - ❖ Ingresos y Gastos
  - ❖ Patrimonio y Activos, etc



Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

Flujo de Acceso:

- El Acceso a las Pantallas es primero ingresando a la Pantalla de Registro de Clientes. (3)
- Luego se procede a seleccionar a un Cliente desde un Listado

- Una vez seleccionado el cliente se puede acceder mediante los botones en Pantalla a las diferentes opciones antes mencionadas (3.1) (3.2) (3.3) (3.4)

Flujo de Pantallas Módulo de Consultas

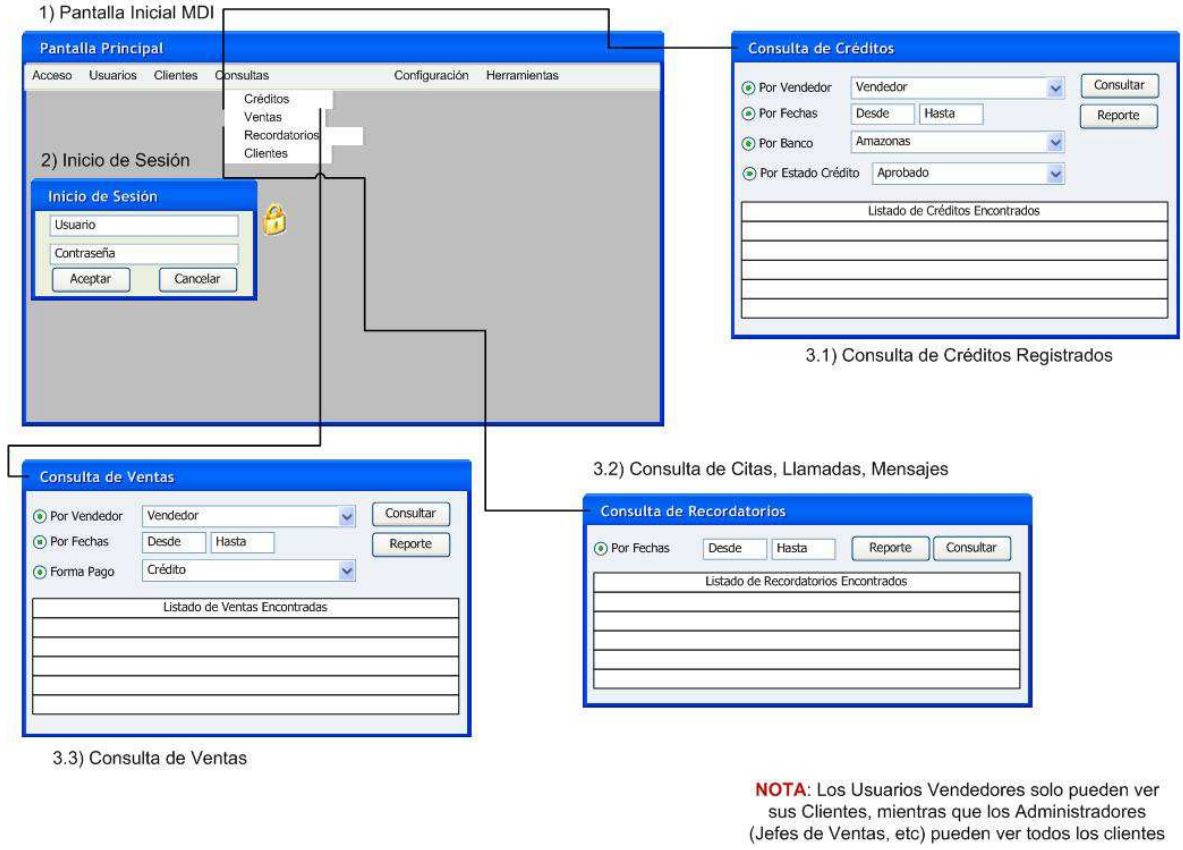
En este módulo contamos con las diferentes pantallas para realizar Consultas, enfocadas básicamente a los Jefes de Ventas y Gerentes. De aquí también se podrán emitir Reportes. El acceso a las Pantallas es de Forma Individual.

Las Consultas pueden ser de:

- Créditos
    - Filtradas por:
        - Vendedores
        - Bancos
        - Estados de Crédito
        - Vehículos
        - Rangos de Fecha

- Ventas
    - Filtradas por:
        - Vendedores
        - Formas de Compra
        - Vehículos
        - Rangos de Fecha

- Recordatorios
    - Filtrados por
        - Rangos de Fecha



Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

Flujo de Acceso:

El Acceso a cada Pantalla es de forma individual desde el Menú Principal

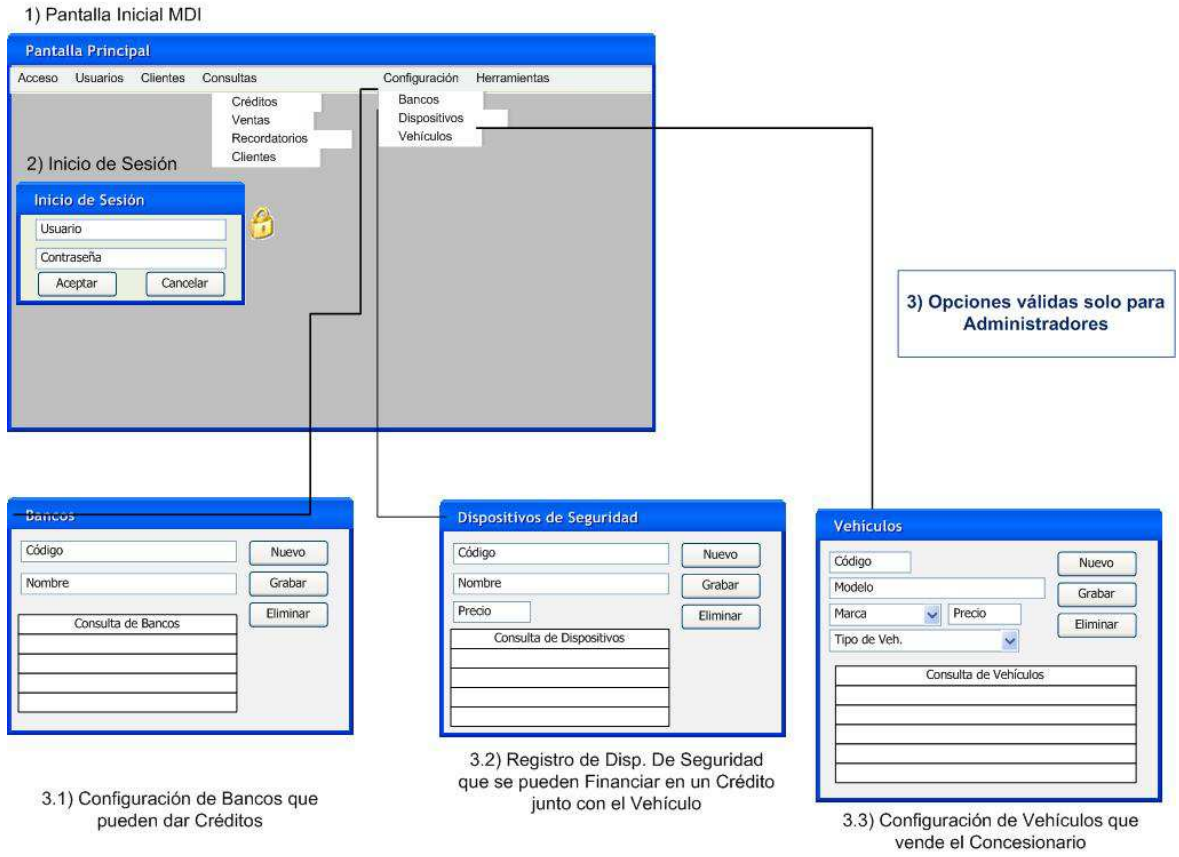
Flujo de Pantallas Módulo de Configuración:

En este Módulo los usuarios administradores de sistema, pueden configurar los parámetros necesarios vinculados al concesionario donde el sistema está implementado.

Estos parámetros incluyen:

- Vehículos usados para Ventas y Créditos (Modelo, Precio, etc)
- Marcas de Vehículos que tiene el Concesionario
- Bancos que pueden dar Créditos
- Compañías de Seguros con las que se puede asegurar un Vehículo, al momento de registrar la Venta

- Dispositivos de Seguridad que se pueden financiar para los Vehículos



Elaborado por: Babici Bogdan, Tierra Doris, Achig María.

Fuente: Tópico de Graduación UPS.

Flujo de Acceso:

El Acceso a cada Pantalla es de forma individual desde el Menú Principal, y a este Menú solo tienen acceso los usuarios Administradores.

PAQUETES DE LA BASE DE DATOS

- ***PK_AGENDA.***

Objetivo: Administrar los Recordatorios por Usuario.

Contiene los Procedimientos Almacenados para la Creación, Modificación, Eliminación y Consulta de la Tabla Agenda, en la cual se guardan todos los Recordatorios para Citas, Llamadas, Mensajes, etc.

```
prompt Creating package body PK_AGENDA
prompt ==============================

CREATE OR REPLACE Package Body USER15.PK_AGENDA IS

  NEW_ID          NUMBER;
  MSG             VARCHAR2(100);
  EXP_ERROR       EXCEPTION;
  FECHA_HORA      VARCHAR2(20);
  CONTADOR        NUMBER;

--Todo lo que va a continuación de estos simbolos ¨--¨ es comentario.
--PROCEDURE SP_INSERTAR
--Recibe todos los parámetros de la tabla para ingresar o registrar una nueva agenda.

  PROCEDURE SP_INSERTAR
  (
        P_CODUSUARIO      IN NUMBER     DEFAULT NULL,
        P_CODCLIENTE      IN NUMBER     DEFAULT NULL,
        P_FECHA           IN DATE       DEFAULT NULL,
        P_HORA            IN VARCHAR2   DEFAULT NULL,
        P_MENSAJE         IN VARCHAR2   DEFAULT NULL,
        P_RESPUESTA       IN VARCHAR2   DEFAULT NULL,
        P_ID              OUT           NUMBER
  )
  IS
  BEGIN
     FECHA_HORA := TO_CHAR(P_FECHA,'DD/MM/YYYY') || ' ' || P_HORA || ':00';
     MSG:= 'NO SE PUDO INSERTAR MESNAJE';
     SELECT NVL(MAX(CODMENSAJE) + 1, 1) INTO NEW_ID FROM AGENDA;
     INSERT INTO Agenda
       (codmensaje, codusuario, codcliente, fecha, hora, mensaje, respuesta)
     VALUES
      (NEW_ID, P_CODUSUARIO, P_CODCLIENTE, P_FECHA,
      TO_DATE(FECHA_HORA,'DD/MM/YYYY HH24:MI:SS'), P_MENSAJE,
      P_RESPUESTA);
     SELECT NEW_ID INTO P_ID FROM DUAL;
     COMMIT;
     EXCEPTION
     WHEN EXP_ERROR THEN
        RAISE_APPLICATION_ERROR(-20000, MSG);
        ROLLBACK;
```

```
      WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO');
        ROLLBACK;
  END SP_INSERTAR;
```

--PROCEDURE SP_GUARDAR

--Permite guardar los mensajes de los clientes con su respectiva fecha y respuesta/*

```
  PROCEDURE SP_GUARDAR
  (
        P_CODMENSAJE      IN NUMBER     DEFAULT NULL,
        P_FECHA           IN DATE       DEFAULT NULL,
        P_HORA            IN VARCHAR2   DEFAULT NULL,
        P_MENSAJE         IN VARCHAR2   DEFAULT NULL,
        P_RESPUESTA       IN VARCHAR2   DEFAULT NULL
  )
  IS
  BEGIN
     FECHA_HORA := TO_CHAR(P_FECHA,'DD/MM/YYYY') || ' ' || P_HORA || ':00';
     MSG:= 'NO SE PUDO GUARDAR MESNAJE';
     UPDATE Agenda
     SET    Fecha      = P_FECHA,
          Hora       = TO_DATE(FECHA_HORA,'DD/MM/YYYY HH24:MI:SS'),
          Mensaje    = P_MENSAJE,
          Respuesta  = P_RESPUESTA
     WHERE  CODMENSAJE  = P_CODMENSAJE;
     COMMIT;
     EXCEPTION
     WHEN EXP_ERROR THEN
        RAISE_APPLICATION_ERROR(-20000, MSG);
        ROLLBACK;
     WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO');
        ROLLBACK;
  END SP_GUARDAR;
```

--PROCEDURE SP_CONSULTAR

*/*Permite consultar la agenda de un cliente determinado para conocer su condición crediticia.*/*

```
 PROCEDURE SP_CONSULTAR
  (
        P_CODMENSAJE      IN NUMBER     DEFAULT NULL,
        MY_CURSOR         IN OUT        T_CURSOR
  )
  IS
  BEGIN
     OPEN MY_CURSOR FOR
     SELECT * FROM AGENDA WHERE CODMENSAJE = P_CODMENSAJE;
  END SP_CONSULTAR;
```

--PROCEDURE SP_ELIMINAR

--Elimina un mensaje de la agenda de un cliente específico

```
PROCEDURE SP_ELIMINAR
  (
        P_CODMENSAJE      IN NUMBER     DEFAULT NULL
  )
  IS
  BEGIN
     MSG:= 'NO SE PUDO ELIMINAR MENSAJE';
     DELETE FROM AGENDA WHERE CODMENSAJE = P_CODMENSAJE;
     COMMIT;
     EXCEPTION
     WHEN EXP_ERROR THEN
        RAISE_APPLICATION_ERROR(-20000, MSG);
        ROLLBACK;
     WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
        PROCEDIMIENTO');
        ROLLBACK;
  END SP_ELIMINAR;
```

--PROCEDURE SP_CONSULTARPORFECHAS

--Consultar por fechas y usuario

```
  PROCEDURE SP_CONSULTARPORFECHAS
  (
        P_OPCION          IN CHAR       DEFAULT NULL,
        --S=SI INCLUYE FECHAS, N=NO INCLUYE FECHAS
        P_FECINI          IN DATE       DEFAULT NULL,
        P_FECFIN          IN DATE       DEFAULT NULL,
        P_CODUSUARIO      IN NUMBER     DEFAULT NULL,
        MY_CURSOR         IN OUT        T_CURSOR
  )
  IS
  BEGIN
     IF P_OPCION = 'S' THEN
       OPEN MY_CURSOR FOR
       SELECT codmensaje,fecha,cliente,mensaje,respuesta
       FROM   VisConsulAgenda
       WHERE  codusuario = P_CODUSUARIO
       AND    Fecha BETWEEN P_FECINI AND P_FECFIN;
     ELSE
       OPEN MY_CURSOR FOR
       SELECT codmensaje,fecha,cliente,mensaje,respuesta
       FROM   VisConsulAgenda
       WHERE  codusuario = P_CODUSUARIO;
     END IF;
  END SP_CONSULTARPORFECHAS;
```

PROCEDURE SP_CONSULTARPORFECHAS_FORMS

/**Permite consultar la agenda de un cliente por un rango especificado de fechas. */***

```
  PROCEDURE SP_CONSULTARPORFECHAS_FORMS
  (
        P_OPCION          IN CHAR       DEFAULT NULL,
        --S=SI INCLUYE FECHAS, N=NO INCLUYE FECHAS
        P_FECINI          IN DATE       DEFAULT NULL,
        P_FECFIN          IN DATE       DEFAULT NULL,
```

```
        P_CODUSUARIO      IN NUMBER     DEFAULT NULL,
        DATA_AGENDA       IN OUT        T_AGENDA
  )
  IS
  CURSOR C_CONFECHAS IS
  SELECT * FROM VisConsulAgenda
  WHERE  codusuario = P_CODUSUARIO
  AND    Fecha BETWEEN P_FECINI AND P_FECFIN;
  CURSOR C_SINFECHAS IS
  SELECT * FROM VisConsulAgenda
  WHERE  codusuario = P_CODUSUARIO;
  BEGIN
       CONTADOR := 0;
       IF P_OPCION = 'S' THEN
         FOR FILA IN C_CONFECHAS LOOP
            CONTADOR := CONTADOR + 1;
            DATA_AGENDA(CONTADOR) := FILA;
         END LOOP;
         IF C_CONFECHAS%ISOPEN THEN
           CLOSE C_CONFECHAS;
         END IF;
       ELSE
         FOR FILA IN C_SINFECHAS LOOP
            CONTADOR := CONTADOR + 1;
            DATA_AGENDA(CONTADOR) := FILA;
         END LOOP;
         IF C_SINFECHAS%ISOPEN THEN
           CLOSE C_SINFECHAS;
         END IF;
       END IF;
  END SP_CONSULTARPORFECHAS_FORMS;
END PK_AGENDA;
/
```

- ***PK_BANCOS.***

Objetivo: Permitirá la Creación, Modificación, Eliminación y Consulta de la Tabla Bancos mediante procedimientos almacenados.

```
prompt Creating package body PK_BANCOS
prompt ==============================

CREATE OR REPLACE Package Body USER15.PK_BANCOS I
    NEW_ID          NUMBER;
    MSG             VARCHAR2(100);
    CONTADOR        NUMBER;
    ESEDITABLE      NUMBER;
    EXP_ERROR       EXCEPTION;

--PROCEDURE SP_INSERTAR
--Recibe todos los parámetros de la tabla Bancos para  ingresar un nuevo Banco

PROCEDURE SP_INSERTAR
    (
     P_NOMBRE         IN VARCHAR2 DEFAULT NULL,
     MY_CURSOR        OUT NUMBER
    )
    IS
```

```
BEGIN
    MSG:= 'NO SE PUDO INSERTAR EL BANCO';
    SELECT COUNT(*) INTO CONTADOR FROM BANCOS
    WHERE NOMBRE LIKE P_NOMBRE AND ACTIVO = 1;
    IF CONTADOR > 0 THEN
      MSG := 'EL NOMBRE DE ESTE BANCO YA EXISTE';
      RAISE EXP_ERROR;
    END IF;
    SELECT NVL(MAX(CODBANCO) + 1, 1) INTO NEW_ID FROM BANCOS;
    INSERT INTO BANCOS
    (codbanco,nombre,tipo,editable,activo)
    VALUES
    (NEW_ID,P_NOMBRE,1,1,1);
    SELECT NEW_ID INTO MY_CURSOR FROM DUAL;
    COMMIT;
    EXCEPTION
    WHEN EXP_ERROR THEN
        RAISE_APPLICATION_ERROR(-20000, MSG);
        ROLLBACK;
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20000, 'ERROR: ' || SQLERRM);
        ROLLBACK;
END SP_INSERTAR;
```

--PROCEDURE SP_GUARDAR

--Guarda el id, nombre del banco, asì como su condicion de Activo=0 o Inactivo=1*/

```
PROCEDURE SP_GUARDAR
(
 P_ID             IN NUMBER DEFAULT NULL,
 P_NOMBRE          IN VARCHAR2 DEFAULT NULL,
 P_ACTIVO          IN NUMBER DEFAULT NULL
)
IS
BEGIN
    MSG:= 'NO SE PUDO GRABAR EL BANCO';
    SELECT COUNT(*) INTO CONTADOR FROM BANCOS
    WHERE CODBANCO = P_ID;
    IF CONTADOR = 0 THEN
      MSG:= 'NO SE PUDO ENCONTRAR EL BANCO A MODIFICAR';
      RAISE EXP_ERROR;
    END IF;
    SELECT EDITABLE INTO ESEDITABLE FROM BANCOS
    WHERE CODBANCO = P_ID;
    IF ESEDITABLE = 0 THEN
      MSG:= 'ESTE BANCO ES PROPIO DEL SISTEMA Y NO SE PUEDE
      MODIFICAR
      NI ELIMINAR';
      RAISE EXP_ERROR;
    END IF;
    SELECT COUNT(*) INTO CONTADOR FROM BANCOS
    WHERE NOMBRE LIKE P_NOMBRE AND ACTIVO = 1 AND CODBANCO <>
    P_ID;
    IF CONTADOR > 0 THEN
      MSG:= 'EL NOMBRE YA ESTA ASIGNADO A OTRO BANCO';
      RAISE EXP_ERROR;
```

```
    END IF;
    UPDATE BANCOS SET
    NOMBRE = P_NOMBRE,
    ACTIVO = P_ACTIVO
    WHERE CODBANCO = P_ID;
    COMMIT;
    EXCEPTION
    WHEN EXP_ERROR THEN
        RAISE_APPLICATION_ERROR(-20000, MSG);
        ROLLBACK;
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20000, 'ERROR: ' || SQLERRM);
        ROLLBACK;
END SP_GUARDAR;


-- PROCEDURE SP_ELIMINAR
-- Permite la eliminación de un banco

PROCEDURE SP_ELIMINAR
 (
 P_ID            IN NUMBER DEFAULT NULL,
 P_ACTIVO        IN NUMBER DEFAULT NULL
)
IS
BEGIN
    MSG:= 'NO SE PUDO ELIMINAR EL BANCO';
    SELECT COUNT(*) INTO CONTADOR FROM BANCOS
    WHERE CODBANCO = P_ID;
    IF CONTADOR = 0 THEN
      MSG:= 'NO SE PUDO ENCONTRAR EL BANCO';
      RAISE EXP_ERROR;
    END IF;
    SELECT EDITABLE INTO ESEDITABLE FROM BANCOS
    WHERE CODBANCO = P_ID;
    IF ESEDITABLE = 0 THEN
      MSG:= 'ESTE BANCO ES PROPIO DEL SISTEMA Y NO SE PUEDE
      MODIFICAR
      NI ELIMINAR';
      RAISE EXP_ERROR;
    END IF;

    --Valido para eliminar y restaurar dependiendo del parametro
    UPDATE BANCOS SET
    ACTIVO = P_ACTIVO
    WHERE CODBANCO = P_ID;
    COMMIT;
    EXCEPTION
    WHEN EXP_ERROR THEN
        RAISE_APPLICATION_ERROR(-20000, MSG);
        ROLLBACK;
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20000, 'ERROR: ' || SQLERRM);
        ROLLBACK;
END SP_ELIMINAR;

-- PROCEDURE SP_CONSULTAR
-- Permite consultar la table dispositivos.
```

```
PROCEDURE SP_CONSULTAR
(
 P_ID             IN NUMBER DEFAULT NULL,
 MY_CURSOR        IN OUT T_CURSOR
)
IS
BEGIN
    OPEN MY_CURSOR FOR
    SELECT * FROM DISPOSITIVOS WHERE CODDISPOSITIVO = P_ID;
END SP_CONSULTAR;


-- PROCEDURE SP_CONSULTARPORNOMBRE
-- Permite consultar por el nombre de un banco específico

PROCEDURE SP_CONSULTARPORNOMBRE
(
 P_NOMBRE         IN VARCHAR2 DEFAULT NULL,
 MY_CURSOR        IN OUT T_CURSOR
)
IS
BEGIN
    OPEN MY_CURSOR FOR
    SELECT * FROM BANCOS WHERE NOMBRE LIKE '%' || P_NOMBRE || '%'AND
     ACTIVO = 1  AND TIPO = 1
    ORDER BY NOMBRE;
END SP_CONSULTARPORNOMBRE;

-- PROCEDURE SP_CONSULTARTODOS
--Permite consultar todos los bancos que se encuentran registrados

PROCEDURE SP_CONSULTARTODOS
(
 MY_CURSOR        IN OUT T_CURSOR,
 P_ACTIVO         IN NUMBER DEFAULT NULL
)
IS
BEGIN
    OPEN MY_CURSOR FOR
    SELECT * FROM BANCOS WHERE ACTIVO = P_ACTIVO
    ORDER BY NOMBRE;
END SP_CONSULTARTODOS;


--PROCEDURE SP_CONSULTARBANCOS_FORMS
-- Consulta por el nombre de un banco especìfico.*/

PROCEDURE SP_CONSULTARBANCOS_FORMS
(
 BANCOS_DATA IN OUT T_BANCO,
 P_NOMBRE    IN VARCHAR2   DEFAULT NULL,
 P_ACTIVO    IN NUMBER     DEFAULT NULL
)
IS
    CURSOR CUR_BANCOS IS
    SELECT * FROM BANCOS WHERE ACTIVO = P_ACTIVO AND NOMBRE LIKE
'%'
    || P_NOMBRE || '%'
    AND TIPO = 1
```

```
        ORDER BY NOMBRE;
    BEGIN
        OPEN CUR_BANCOS;
        CONTADOR := 1;
        LOOP
        FETCH CUR_BANCOS INTO
            BANCOS_DATA(CONTADOR).CODBANCO,
            BANCOS_DATA(CONTADOR).NOMBRE,
            BANCOS_DATA(CONTADOR).TIPO,
            BANCOS_DATA(CONTADOR).ACTIVO,
            BANCOS_DATA(CONTADOR).EDITABLE;
            EXIT WHEN CUR_BANCOS%NOTFOUND;
            CONTADOR := CONTADOR + 1;
        END LOOP;
        IF CUR_BANCOS%ISOPEN THEN
          CLOSE CUR_BANCOS;
        END IF;
    EXCEPTION
        WHEN OTHERS THEN
        IF CUR_BANCOS%ISOPEN THEN
          CLOSE CUR_BANCOS;
        END IF;
    END SP_CONSULTARBANCOS_FORMS;
END PK_BANCOS;
/
```

- ***PK_CLIENTES.***

Objetivo: Administrar Datos de la Tabla Clientes.

Contiene los Procedimientos Almacenados para la Creación, Modificación, Eliminación y Consulta de la Tabla Clientes.

```
prompt Creating package body PK_CLIENTES
prompt ================================

create or replace package body user15.PK_CLIENTES IS
  NEW_ID          NUMBER;
  MSG             VARCHAR2(100);
  CONTADOR        NUMBER;
  EXP_ERROR       EXCEPTION;

-- PROCEDURE SP_INSERTAR
-- Ingreso de todos los datos obligatorios para llenar la tabla Cliente

  PROCEDURE SP_INSERTAR
  (
      P_CEDULA        IN VARCHAR2 DEFAULT NULL,
```

```
    P_APELLIDO       IN VARCHAR2 DEFAULT NULL,
    P_NOMBRE         IN VARCHAR2 DEFAULT NULL,
    P_TELCASA        IN VARCHAR2 DEFAULT NULL,
    P_TELOFIC        IN VARCHAR2 DEFAULT NULL,
    P_CELULAR        IN VARCHAR2 DEFAULT NULL,
    P_FECATEN        IN DATE     DEFAULT NULL,
    P_DIRCASA        IN VARCHAR2 DEFAULT NULL,
    P_DIROFIC        IN VARCHAR2 DEFAULT NULL,
    P_MAIL           IN VARCHAR2 DEFAULT NULL,
    P_CODUSUARIO     IN NUMBER   DEFAULT NULL,
    P_CODVENDEDOR    IN NUMBER   DEFAULT NULL,
    P_CODVEHICULO    IN NUMBER   DEFAULT NULL,
    P_PRECIO         IN NUMBER   DEFAULT NULL,
    P_ENTRADA        IN NUMBER   DEFAULT NULL,
    P_CODBANCO       IN NUMBER   DEFAULT NULL,
    P_PLAZO          IN NUMBER   DEFAULT NULL,
    P_GASTOS         IN NUMBER   DEFAULT NULL,
    P_DISPOSITIVO    IN NUMBER   DEFAULT NULL,
    P_COMENTARIO     IN CLOB     DEFAULT NULL,
    P_ESTADO         IN NUMBER   DEFAULT NULL,
    P_ID             OUT NUMBER
)
IS
BEGIN
   MSG:= 'NO SE PUDO INSERTAR CLIENTE';
   IF P_CEDULA <> '9999999999' THEN
    SELECT COUNT(*) INTO CONTADOR FROM CLIENTES
    WHERE CEDULA = P_CEDULA;
    IF CONTADOR > 0 THEN
      MSG := 'YA EXISTE UN CLIENTE CON ESTE NÚMERO DE CÉDULA EN LA
      BASE DE DATOS';
      RAISE EXP_ERROR;
    END IF;
   END IF;
   SELECT NVL(MAX(CODCLIENTE) + 1, 1) INTO NEW_ID FROM CLIENTES;
   INSERT INTO CLIENTES
   (
    codcliente,cedula,apellidos,nombres,telcasa,teltrabajo,celular,fechaatencion,
    dircasa,dirtrabajo,correo,codusuario,codvendedor,codvehiculo,precio,entrada,
    codbanco,plazo,gastos,coddispositivo,comentario,codestado,activo,FECHA
   )
   VALUES
   (
NEW_ID,P_CEDULA,P_APELLIDO,P_NOMBRE,P_TELCASA,P_TELOFIC,P_CELULAR
,P_FECATEN,
P_DIRCASA,P_DIROFIC,P_MAIL,P_CODUSUARIO,P_CODVENDEDOR,P_CODVEHIC
ULO,P_PRECIO,P_ENTRADA,
P_CODBANCO,P_PLAZO,P_GASTOS,P_DISPOSITIVO,P_COMENTARIO,P_ESTADO,1,
SYSDATE()
   );
   SELECT NEW_ID INTO P_ID FROM DUAL;
   COMMIT;
   EXCEPTION
   WHEN EXP_ERROR THEN
      RAISE_APPLICATION_ERROR(-20000, MSG);
      ROLLBACK;
   WHEN OTHERS THEN
      RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
      PROCEDIMIENTO');
```

```
        ROLLBACK;
END SP_INSERTAR;
```

-- PROCEDURE SP_GUARDAR

/ Guarda todos los datos del cliente logicamente comprobados. Recibe el id del registro y demás datos y actualiza siempre y cuando el código del cliente sea igual al id*/*

```
PROCEDURE SP_GUARDAR
(
  P_ID            IN NUMBER   DEFAULT NULL,
  P_CEDULA        IN VARCHAR2 DEFAULT NULL,
  P_APELLIDO      IN VARCHAR2 DEFAULT NULL,
  P_NOMBRE        IN VARCHAR2 DEFAULT NULL,
  P_TELCASA       IN VARCHAR2 DEFAULT NULL,
  P_TELOFIC       IN VARCHAR2 DEFAULT NULL,
  P_CELULAR       IN VARCHAR2 DEFAULT NULL,
  P_FECATEN       IN DATE     DEFAULT NULL,
  P_DIRCASA       IN VARCHAR2 DEFAULT NULL,
  P_DIROFIC       IN VARCHAR2 DEFAULT NULL,
  P_MAIL          IN VARCHAR2 DEFAULT NULL,
  P_CODVENDEDOR   IN NUMBER   DEFAULT NULL,
  P_CODVEHICULO   IN NUMBER   DEFAULT NULL,
  P_PRECIO        IN NUMBER   DEFAULT NULL,
  P_ENTRADA       IN NUMBER   DEFAULT NULL,
  P_CODBANCO      IN NUMBER   DEFAULT NULL,
  P_PLAZO         IN NUMBER   DEFAULT NULL,
  P_GASTOS        IN NUMBER   DEFAULT NULL,
  P_DISPOSITIVO   IN NUMBER   DEFAULT NULL,
  P_COMENTARIO    IN CLOB     DEFAULT NULL,
  P_ESTADO        IN NUMBER   DEFAULT NULL,
  P_ACTIVO        IN NUMBER   DEFAULT NULL
)
IS
BEGIN
   MSG:= 'NO SE PUDO GRABAR CLIENTE';
   IF P_CEDULA <> '9999999999' THEN
    SELECT COUNT(*) INTO CONTADOR FROM CLIENTES
    WHERE CEDULA = P_CEDULA AND CODCLIENTE <> P_ID;
    IF CONTADOR > 0 THEN
      MSG := 'YA EXISTE OTRO CLIENTE CON ESTE NÚMERO DE CÉDULA EN LA
      BASE DE DATOS';
      RAISE EXP_ERROR;
    END IF;
   END IF;
   SELECT COUNT(CODCLIENTE) INTO CONTADOR FROM CLIENTES
   WHERE CODCLIENTE = P_ID;
   IF CONTADOR = 0 THEN
     MSG := 'NO SE ENCUENTRA EL CLIENTE A MODIFICAR';
     RAISE EXP_ERROR;
   END IF;
   UPDATE CLIENTES SET
   cedula        = P_CEDULA,
   apellidos     = P_APELLIDO,
   nombres       = P_NOMBRE,
   telcasa       = P_TELCASA,
   teltrabajo    = P_TELOFIC,
   celular       = P_CELULAR,
   dircasa       = P_DIRCASA,
   dirtrabajo    = P_DIROFIC,
```

```
    correo         = P_MAIL,
    codvehiculo    = P_CODVEHICULO,
    precio         = P_PRECIO,
    entrada        = P_ENTRADA,
    plazo          = P_PLAZO,
    gastos         = P_GASTOS,
    coddispositivo = P_DISPOSITIVO,
    comentario     = P_COMENTARIO,
    fechaatencion  = P_FECATEN,
    codvendedor    = P_CODVENDEDOR,
    codbanco       = P_CODBANCO,
    activo         = P_ACTIVO,
    codestado      = P_ESTADO
    WHERE codcliente  = P_ID;
    COMMIT;
    EXCEPTION
    WHEN EXP_ERROR THEN
        RAISE_APPLICATION_ERROR(-20000, MSG);
        ROLLBACK;
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO');
        ROLLBACK;
  END SP_GUARDAR;
```

-- PROCEDURE SP_ELIMINAR

*/*Los datos no se eliminan de la base de datos mas bien se activan e inactivan. Cada tabla tiene su campo activo =0 e inactivo =1. La información que se elimina se va a una papelera de reciclaje y luego si desea restaurarlo también lo podrá hacer*/*

```
  PROCEDURE SP_ELIMINAR
  (
    P_ID            IN NUMBER   DEFAULT NULL,
    P_ACTIVO        IN NUMBER   DEFAULT NULL
  )
  IS
  --Sirve tanto para eliminación como para restauración del cliente

  BEGIN
    MSG:= 'NO SE PUDO ELIMINAR CLIENTE';
    SELECT COUNT(CODCLIENTE) INTO CONTADOR FROM CLIENTES
    WHERE CODCLIENTE = P_ID;
    IF CONTADOR = 0 THEN
      MSG := 'NO SE ENCUENTRA EL CLIENTE A ELIMINAR';
      RAISE EXP_ERROR;
    END IF;
    UPDATE CLIENTES SET ACTIVO = P_ACTIVO
    WHERE CODCLIENTE = P_ID;
    COMMIT;
    EXCEPTION
    WHEN EXP_ERROR THEN
        RAISE_APPLICATION_ERROR(-20000, MSG);
        ROLLBACK;
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
        PROCEDIMIENTO');
        ROLLBACK;
  END SP_ELIMINAR;
```

```
--PROCEDURE SP_CONSULTAR
-- Consulta a los clientes ingresados a la base de datos.

  PROCEDURE SP_CONSULTAR
  (
    --CONSULTA POR ID UN CLIENTE
    P_ID       IN NUMBER   DEFAULT NULL,
    MY_CURSOR  IN OUT T_CURSOR
  )
  IS
  BEGIN
      OPEN MY_CURSOR FOR
      SELECT CODCLIENTE, CEDULA, APELLIDOS, NOMBRES, telcasa, teltrabajo,
celular, fechaatencion,
           dircasa, dirtrabajo, correo, codvendedor, codvehiculo, codbanco, precio, entrada,
           plazo, gastos, coddispositivo, codestado
      FROM   CLIENTES
      WHERE  CODCLIENTE = P_ID;
  END SP_CONSULTAR;

  --PROCEDURE SP_CONSULTASMIXTAS_FORMS
  --Consulta de clientes por opciones desde la ventana de clientes.
  /*Sirve para consultas por cedula, cliente o fecha. Cuando se desee realizar consultas
    lo que se hace es en base a la opción que recibe el parámetro OP, Si op=C realizara
    una búsqueda por cedula, si op=N realizará una consulta por Nombre y así con el
   resto de parámetros que reciba*/

  PROCEDURE SP_CONSULTASMIXTAS_FORMS
  (
    --Sirve para las consultas por cedula, cliente o fechas
    P_CEDULA         IN VARCHAR2  DEFAULT NULL,
    P_NOMBRES        IN VARCHAR2  DEFAULT NULL,
    P_APELLIDOS      IN VARCHAR2  DEFAULT NULL,
    --PARA FECHAS
    P_FECINI_IN      IN VARCHAR2  DEFAULT NULL,
    P_FECFIN_IN      IN VARCHAR2  DEFAULT NULL,

    --Tipo de consulta c=cedula, a=apellidos, n=nombres, f=fechas
    P_OP             IN CHAR      DEFAULT NULL,

    --Si incluye vendedor el id del mismo caso contrario p_vendedor = -1
     O SEA ADMINISTRADOR QUE PUEDE CONSULTAR A TODOS
    P_VENDEDOR       IN NUMBER    DEFAULT NULL,
    CLIENTES_DATA    IN OUT T_CLIENTE
  )
  IS
    ES_VENDEDOR      BOOLEAN      := FALSE;
    P_FECINI         DATE         := TO_DATE(P_FECINI_IN,'DD/MM/YYYY');
    P_FECFIN         DATE         := TO_DATE(P_FECFIN_IN,'DD/MM/YYYY');

    --Cursores de busqueda para el administrador
    --Todos
    CURSOR CUR_CLIENTE_AT IS
    SELECT codcliente,cedula,apellidos,nombres,telcasa,teltrabajo,celular,
         modelo,nombre,fechaatencion,estadogm,tipoclientegm,codvendedor,activo
    FROM   VISCONSULCLIENTES
    WHERE  ACTIVO = 1 ORDER BY APELLIDOS, NOMBRES;

    --Cedula
```

```
    CURSOR CUR_CLIENTE_AC IS
    SELECT codcliente,cedula,apellidos,nombres,telcasa,teltrabajo,celular,
        modelo,nombre,fechaatencion,estadogm,tipoclientegm,codvendedor,activo
    FROM   VISCONSULCLIENTES
    WHERE  ACTIVO = 1 AND CEDULA = P_CEDULA;

    --Apellidos
    CURSOR CUR_CLIENTE_AA IS
    SELECT codcliente,cedula,apellidos,nombres,telcasa,teltrabajo,celular,
        modelo,nombre,fechaatencion,estadogm,tipoclientegm,codvendedor,activo
    FROM   VISCONSULCLIENTES
    WHERE  ACTIVO = 1 AND APELLIDOS LIKE '%' || P_APELLIDOS || '%'
    ORDER  BY APELLIDOS;

    --Nombres
    CURSOR CUR_CLIENTE_AN IS
    SELECT codcliente,cedula,apellidos,nombres,telcasa,teltrabajo,celular,
        modelo,nombre,fechaatencion,estadogm,tipoclientegm,codvendedor,activo
    FROM   VISCONSULCLIENTES
    WHERE  ACTIVO = 1 AND NOMBRES LIKE '%' || P_NOMBRES || '%'
    ORDER  BY NOMBRES;

    --Por fechas
    CURSOR CUR_CLIENTE_AF IS
    SELECT codcliente,cedula,apellidos,nombres,telcasa,teltrabajo,celular,
        modelo,nombre,fechaatencion,estadogm,tipoclientegm,codvendedor,activo
    FROM   VISCONSULCLIENTES
    WHERE  ACTIVO = 1 AND FECHAATENCION BETWEEN P_FECINI AND
P_FECFIN
    ORDER  BY FECHAATENCION, APELLIDOS;

    --Cursores para vendedores
    --todos
    CURSOR CUR_CLIENTE_VT IS
    SELECT codcliente,cedula,apellidos,nombres,telcasa,teltrabajo,celular,
        modelo,nombre,fechaatencion,estadogm,tipoclientegm,codvendedor,activo
    FROM   VISCONSULCLIENTES
    WHERE  ACTIVO = 1 AND CODVENDEDOR = P_VENDEDOR ORDER BY
APELLIDOS, NOMBRES;

    --Cedulas
    CURSOR CUR_CLIENTE_VC IS
    SELECT codcliente,cedula,apellidos,nombres,telcasa,teltrabajo,celular,
        modelo,nombre,fechaatencion,estadogm,tipoclientegm,codvendedor,activo
    FROM   VISCONSULCLIENTES
    WHERE  ACTIVO = 1 AND CEDULA = P_CEDULA AND CODVENDEDOR =
P_VENDEDOR;

    --Apellidos
    CURSOR CUR_CLIENTE_VA IS
    SELECT codcliente,cedula,apellidos,nombres,telcasa,teltrabajo,celular,
        modelo,nombre,fechaatencion,estadogm,tipoclientegm,codvendedor,activo
    FROM   VISCONSULCLIENTES
    WHERE  ACTIVO = 1 AND APELLIDOS LIKE '%' || P_APELLIDOS || '%' AND
CODVENDEDOR = P_VENDEDOR
    ORDER  BY APELLIDOS;

    --Nombres
    CURSOR CUR_CLIENTE_VN IS
```

```
    SELECT codcliente,cedula,apellidos,nombres,telcasa,teltrabajo,celular,
        modelo,nombre,fechaatencion,estadogm,tipoclientegm,codvendedor,activo
    FROM   VISCONSULCLIENTES
    WHERE  ACTIVO = 1 AND NOMBRES LIKE '%' || P_NOMBRES || '%' AND
CODVENDEDOR = P_VENDEDOR
    ORDER  BY NOMBRES;

    --Fechas
    CURSOR CUR_CLIENTE_VF IS
    SELECT codcliente,cedula,apellidos,nombres,telcasa,teltrabajo,celular,
        modelo,nombre,fechaatencion,estadogm,tipoclientegm,codvendedor,activo
    FROM   VISCONSULCLIENTES
    WHERE  ACTIVO = 1 AND FECHAATENCION BETWEEN P_FECINI AND
P_FECFIN AND CODVENDEDOR = P_VENDEDOR
    ORDER  BY FECHAATENCION, APELLIDOS;
  BEGIN
      SELECT COUNT(CODUSUARIO) INTO CONTADOR
      FROM USUARIOS WHERE CARGO ='Vendedor' AND CODUSUARIO =
P_VENDEDOR;
      IF CONTADOR > 0 THEN
         ES_VENDEDOR := TRUE;
      END IF;
      IF ES_VENDEDOR = FALSE THEN

      --Administrador
        --TODOS
        IF P_OP = 'T' THEN
          OPEN CUR_CLIENTE_AT;
          CONTADOR := 1;
          LOOP
          FETCH CUR_CLIENTE_AT INTO
              CLIENTES_DATA(CONTADOR).CODCLIENTE,
              CLIENTES_DATA(CONTADOR).CEDULA,
              CLIENTES_DATA(CONTADOR).APELLIDOS,
              CLIENTES_DATA(CONTADOR).NOMBRES,
              CLIENTES_DATA(CONTADOR).TELCASA,
              CLIENTES_DATA(CONTADOR).TELTRABAJO,
              CLIENTES_DATA(CONTADOR).CELULAR,
              CLIENTES_DATA(CONTADOR).MODELO,
              CLIENTES_DATA(CONTADOR).NOMBRE,
              CLIENTES_DATA(CONTADOR).FECHAATENCION,
              CLIENTES_DATA(CONTADOR).ESTADOGM,
              CLIENTES_DATA(CONTADOR).TIPOCLIENTEGM,
              CLIENTES_DATA(CONTADOR).CODVENDEDOR,
              CLIENTES_DATA(CONTADOR).ACTIVO;
              EXIT WHEN CUR_CLIENTE_AT%NOTFOUND;
              CONTADOR := CONTADOR + 1;
          END LOOP;
          IF CUR_CLIENTE_AT%ISOPEN THEN
            CLOSE CUR_CLIENTE_AT;
          END IF;
        END IF;

        --Cedula
        IF P_OP = 'C' THEN
          OPEN CUR_CLIENTE_AC;
          CONTADOR := 1;
          LOOP
          FETCH CUR_CLIENTE_AC INTO
```

```
        CLIENTES_DATA(CONTADOR).CODCLIENTE,
        CLIENTES_DATA(CONTADOR).CEDULA,
        CLIENTES_DATA(CONTADOR).APELLIDOS,
        CLIENTES_DATA(CONTADOR).NOMBRES,
        CLIENTES_DATA(CONTADOR).TELCASA,
        CLIENTES_DATA(CONTADOR).TELTRABAJO,
        CLIENTES_DATA(CONTADOR).CELULAR,
        CLIENTES_DATA(CONTADOR).MODELO,
        CLIENTES_DATA(CONTADOR).NOMBRE,
        CLIENTES_DATA(CONTADOR).FECHAATENCION,
        CLIENTES_DATA(CONTADOR).ESTADOGM,
        CLIENTES_DATA(CONTADOR).TIPOCLIENTEGM,
        CLIENTES_DATA(CONTADOR).CODVENDEDOR,
        CLIENTES_DATA(CONTADOR).ACTIVO;
        EXIT WHEN CUR_CLIENTE_AC%NOTFOUND;
        CONTADOR := CONTADOR + 1;
    END LOOP;
    IF CUR_CLIENTE_AC%ISOPEN THEN
      CLOSE CUR_CLIENTE_AC;
    END IF;
  END IF;
  --Apellidos
  IF P_OP = 'A' THEN
    OPEN CUR_CLIENTE_AA;
    CONTADOR := 1;
    LOOP
    FETCH CUR_CLIENTE_AA INTO
        CLIENTES_DATA(CONTADOR).CODCLIENTE,
        CLIENTES_DATA(CONTADOR).CEDULA,
        CLIENTES_DATA(CONTADOR).APELLIDOS,
        CLIENTES_DATA(CONTADOR).NOMBRES,
        CLIENTES_DATA(CONTADOR).TELCASA,
        CLIENTES_DATA(CONTADOR).TELTRABAJO,
        CLIENTES_DATA(CONTADOR).CELULAR,
        CLIENTES_DATA(CONTADOR).MODELO,
        CLIENTES_DATA(CONTADOR).NOMBRE,
        CLIENTES_DATA(CONTADOR).FECHAATENCION,
        CLIENTES_DATA(CONTADOR).ESTADOGM,
        CLIENTES_DATA(CONTADOR).TIPOCLIENTEGM,
        CLIENTES_DATA(CONTADOR).CODVENDEDOR,
        CLIENTES_DATA(CONTADOR).ACTIVO;
        EXIT WHEN CUR_CLIENTE_AA%NOTFOUND;
        CONTADOR := CONTADOR + 1;
    END LOOP;
    IF CUR_CLIENTE_AA%ISOPEN THEN
      CLOSE CUR_CLIENTE_AA;
    END IF;
  END IF;

  --Nombres
  IF P_OP = 'N' THEN
    OPEN CUR_CLIENTE_AN;
    CONTADOR := 1;
    LOOP
    FETCH CUR_CLIENTE_AN INTO
        CLIENTES_DATA(CONTADOR).CODCLIENTE,
        CLIENTES_DATA(CONTADOR).CEDULA,
        CLIENTES_DATA(CONTADOR).APELLIDOS,
        CLIENTES_DATA(CONTADOR).NOMBRES,
```

```
          CLIENTES_DATA(CONTADOR).TELCASA,
          CLIENTES_DATA(CONTADOR).TELTRABAJO,
          CLIENTES_DATA(CONTADOR).CELULAR,
          CLIENTES_DATA(CONTADOR).MODELO,
          CLIENTES_DATA(CONTADOR).NOMBRE,
          CLIENTES_DATA(CONTADOR).FECHAATENCION,
          CLIENTES_DATA(CONTADOR).ESTADOGM,
          CLIENTES_DATA(CONTADOR).TIPOCLIENTEGM,
          CLIENTES_DATA(CONTADOR).CODVENDEDOR,
          CLIENTES_DATA(CONTADOR).ACTIVO;
          EXIT WHEN CUR_CLIENTE_AN%NOTFOUND;
          CONTADOR := CONTADOR + 1;
    END LOOP;
    IF CUR_CLIENTE_AN%ISOPEN THEN
      CLOSE CUR_CLIENTE_AN;
    END IF;
  END IF;

  --Fechas
  IF P_OP = 'F' THEN
    OPEN CUR_CLIENTE_AF;
    CONTADOR := 1;
    LOOP
    FETCH CUR_CLIENTE_AF INTO
          CLIENTES_DATA(CONTADOR).CODCLIENTE,
          CLIENTES_DATA(CONTADOR).CEDULA,
          CLIENTES_DATA(CONTADOR).APELLIDOS,
          CLIENTES_DATA(CONTADOR).NOMBRES,
          CLIENTES_DATA(CONTADOR).TELCASA,
          CLIENTES_DATA(CONTADOR).TELTRABAJO,
          CLIENTES_DATA(CONTADOR).CELULAR,
          CLIENTES_DATA(CONTADOR).MODELO,
          CLIENTES_DATA(CONTADOR).NOMBRE,
          CLIENTES_DATA(CONTADOR).FECHAATENCION,
          CLIENTES_DATA(CONTADOR).ESTADOGM,
          CLIENTES_DATA(CONTADOR).TIPOCLIENTEGM,
          CLIENTES_DATA(CONTADOR).CODVENDEDOR,
          CLIENTES_DATA(CONTADOR).ACTIVO;
          EXIT WHEN CUR_CLIENTE_AF%NOTFOUND;
          CONTADOR := CONTADOR + 1;
    END LOOP;
    IF CUR_CLIENTE_AF%ISOPEN THEN
      CLOSE CUR_CLIENTE_AF;
    END IF;
  END IF;
ELSE

--Vendedor
  --Todos
  IF P_OP = 'T' THEN
    OPEN CUR_CLIENTE_VT;
    CONTADOR := 1;
    LOOP
    FETCH CUR_CLIENTE_VT INTO
          CLIENTES_DATA(CONTADOR).CODCLIENTE,
          CLIENTES_DATA(CONTADOR).CEDULA,
          CLIENTES_DATA(CONTADOR).APELLIDOS,
          CLIENTES_DATA(CONTADOR).NOMBRES,
          CLIENTES_DATA(CONTADOR).TELCASA,
```

```
        CLIENTES_DATA(CONTADOR).TELTRABAJO,
        CLIENTES_DATA(CONTADOR).CELULAR,
        CLIENTES_DATA(CONTADOR).MODELO,
        CLIENTES_DATA(CONTADOR).NOMBRE,
        CLIENTES_DATA(CONTADOR).FECHAATENCION,
        CLIENTES_DATA(CONTADOR).ESTADOGM,
        CLIENTES_DATA(CONTADOR).TIPOCLIENTEGM,
        CLIENTES_DATA(CONTADOR).CODVENDEDOR,
        CLIENTES_DATA(CONTADOR).ACTIVO;
        EXIT WHEN CUR_CLIENTE_VT%NOTFOUND;
        CONTADOR := CONTADOR + 1;
  END LOOP;
  IF CUR_CLIENTE_VT%ISOPEN THEN
    CLOSE CUR_CLIENTE_VT;
  END IF;
END IF;

--Cedula
IF P_OP = 'C' THEN
  OPEN CUR_CLIENTE_VC;
  CONTADOR := 1;
  LOOP
  FETCH CUR_CLIENTE_VC INTO
        CLIENTES_DATA(CONTADOR).CODCLIENTE,
        CLIENTES_DATA(CONTADOR).CEDULA,
        CLIENTES_DATA(CONTADOR).APELLIDOS,
        CLIENTES_DATA(CONTADOR).NOMBRES,
        CLIENTES_DATA(CONTADOR).TELCASA,
        CLIENTES_DATA(CONTADOR).TELTRABAJO,
        CLIENTES_DATA(CONTADOR).CELULAR,
        CLIENTES_DATA(CONTADOR).MODELO,
        CLIENTES_DATA(CONTADOR).NOMBRE,
        CLIENTES_DATA(CONTADOR).FECHAATENCION,
        CLIENTES_DATA(CONTADOR).ESTADOGM,
        CLIENTES_DATA(CONTADOR).TIPOCLIENTEGM,
        CLIENTES_DATA(CONTADOR).CODVENDEDOR,
        CLIENTES_DATA(CONTADOR).ACTIVO;
        EXIT WHEN CUR_CLIENTE_VC%NOTFOUND;
        CONTADOR := CONTADOR + 1;
  END LOOP;
  IF CUR_CLIENTE_VC%ISOPEN THEN
    CLOSE CUR_CLIENTE_VC;
  END IF;
END IF;

--Apellidos
IF P_OP = 'A' THEN
  OPEN CUR_CLIENTE_VA;
  CONTADOR := 1;
  LOOP
  FETCH CUR_CLIENTE_VA INTO
        CLIENTES_DATA(CONTADOR).CODCLIENTE,
        CLIENTES_DATA(CONTADOR).CEDULA,
        CLIENTES_DATA(CONTADOR).APELLIDOS,
        CLIENTES_DATA(CONTADOR).NOMBRES,
        CLIENTES_DATA(CONTADOR).TELCASA,
        CLIENTES_DATA(CONTADOR).TELTRABAJO,
        CLIENTES_DATA(CONTADOR).CELULAR,
        CLIENTES_DATA(CONTADOR).MODELO,
```

```
        CLIENTES_DATA(CONTADOR).NOMBRE,
        CLIENTES_DATA(CONTADOR).FECHAATENCION,
        CLIENTES_DATA(CONTADOR).ESTADOGM,
        CLIENTES_DATA(CONTADOR).TIPOCLIENTEGM,
        CLIENTES_DATA(CONTADOR).CODVENDEDOR,
        CLIENTES_DATA(CONTADOR).ACTIVO;
        EXIT WHEN CUR_CLIENTE_VA%NOTFOUND;
        CONTADOR := CONTADOR + 1;
    END LOOP;
    IF CUR_CLIENTE_VA%ISOPEN THEN
      CLOSE CUR_CLIENTE_VA;
    END IF;
  END IF;

  --Nombres
  IF P_OP = 'N' THEN
    OPEN CUR_CLIENTE_VN;
    CONTADOR := 1;
    LOOP
    FETCH CUR_CLIENTE_VN INTO
        CLIENTES_DATA(CONTADOR).CODCLIENTE,
        CLIENTES_DATA(CONTADOR).CEDULA,
        CLIENTES_DATA(CONTADOR).APELLIDOS,
        CLIENTES_DATA(CONTADOR).NOMBRES,
        CLIENTES_DATA(CONTADOR).TELCASA,
        CLIENTES_DATA(CONTADOR).TELTRABAJO,
        CLIENTES_DATA(CONTADOR).CELULAR,
        CLIENTES_DATA(CONTADOR).MODELO,
        CLIENTES_DATA(CONTADOR).NOMBRE,
        CLIENTES_DATA(CONTADOR).FECHAATENCION,
        CLIENTES_DATA(CONTADOR).ESTADOGM,
        CLIENTES_DATA(CONTADOR).TIPOCLIENTEGM,
        CLIENTES_DATA(CONTADOR).CODVENDEDOR,
        CLIENTES_DATA(CONTADOR).ACTIVO;
        EXIT WHEN CUR_CLIENTE_VN%NOTFOUND;
        CONTADOR := CONTADOR + 1;
    END LOOP;
    IF CUR_CLIENTE_VN%ISOPEN THEN
      CLOSE CUR_CLIENTE_VN;
    END IF;
  END IF;

  --Fechas
  IF P_OP = 'F' THEN
    OPEN CUR_CLIENTE_VF;
    CONTADOR := 1;
    LOOP
    FETCH CUR_CLIENTE_VF INTO
        CLIENTES_DATA(CONTADOR).CODCLIENTE,
        CLIENTES_DATA(CONTADOR).CEDULA,
        CLIENTES_DATA(CONTADOR).APELLIDOS,
        CLIENTES_DATA(CONTADOR).NOMBRES,
        CLIENTES_DATA(CONTADOR).TELCASA,
        CLIENTES_DATA(CONTADOR).TELTRABAJO,
        CLIENTES_DATA(CONTADOR).CELULAR,
        CLIENTES_DATA(CONTADOR).MODELO,
        CLIENTES_DATA(CONTADOR).NOMBRE,
        CLIENTES_DATA(CONTADOR).FECHAATENCION,
        CLIENTES_DATA(CONTADOR).ESTADOGM,
```

```
            CLIENTES_DATA(CONTADOR).TIPOCLIENTEGM,
            CLIENTES_DATA(CONTADOR).CODVENDEDOR,
            CLIENTES_DATA(CONTADOR).ACTIVO;
            EXIT WHEN CUR_CLIENTE_VF%NOTFOUND;
            CONTADOR := CONTADOR + 1;
          END LOOP;
          IF CUR_CLIENTE_VF%ISOPEN THEN
            CLOSE CUR_CLIENTE_VF;
          END IF;
        END IF;
      END IF;
      EXCEPTION
        WHEN OTHERS THEN

        --Cerrar cursores de administrador.
        IF CUR_CLIENTE_AC%ISOPEN THEN
          CLOSE CUR_CLIENTE_AC;
        END IF;
        IF CUR_CLIENTE_AN%ISOPEN THEN
          CLOSE CUR_CLIENTE_AN;
        END IF;
        IF CUR_CLIENTE_AA%ISOPEN THEN
          CLOSE CUR_CLIENTE_AA;
        END IF;
        IF CUR_CLIENTE_AF%ISOPEN THEN
          CLOSE CUR_CLIENTE_AF;
        END IF;
        --CERRAR CURSORES DE VEND.
        IF CUR_CLIENTE_VC%ISOPEN THEN
          CLOSE CUR_CLIENTE_VC;
        END IF;
        IF CUR_CLIENTE_VN%ISOPEN THEN
          CLOSE CUR_CLIENTE_VN;
        END IF;
        IF CUR_CLIENTE_VA%ISOPEN THEN
          CLOSE CUR_CLIENTE_VA;
        END IF;
        IF CUR_CLIENTE_VF%ISOPEN THEN
          CLOSE CUR_CLIENTE_VF;
        END IF;
  END SP_CONSULTASMIXTAS_FORMS;
```

--PROCEDURE SP_CONSULTAGENERAL_FORMS

/ Permitirá realizar una consulta si el id del usuario es igual a vendedor se mostraran todos datos del cliente pero se ocultaran sus datos telefónicos para proteger la cartera del vendedor y aparecerá la palabra No disponible en lugar del número telefónico. */*

```
  PROCEDURE SP_CONSULTAGENERAL_FORMS
  (
    --Sirve para las consultas por cedula, cliente o fechas
    P_CEDULA        IN VARCHAR2  DEFAULT NULL,
    P_NOMBRES       IN VARCHAR2  DEFAULT NULL,
    P_APELLIDOS     IN VARCHAR2  DEFAULT NULL,
    --PARA FECHAS
    P_FECINI_IN     IN VARCHAR2  DEFAULT NULL,
    P_FECFIN_IN     IN VARCHAR2  DEFAULT NULL,

    --Tipo de consulta c=cedula, a=apellidos, n=nombres, f=fechas
```

```
    P_OP            IN CHAR      DEFAULT NULL,

    --Si incluye vendedor el id del mismo caso contrario p_vendedor = - 1 y
    --El administrador puede consultar a todos los vendedores
    P_VENDEDOR      IN NUMBER    DEFAULT NULL,
    CLIENTES_DATA   IN OUT T_CLIENTE
  )
  IS
    ES_VENDEDOR     BOOLEAN     := FALSE;
    P_FECINI        DATE        := TO_DATE(P_FECINI_IN,'DD/MM/YYYY');
    P_FECFIN        DATE        := TO_DATE(P_FECFIN_IN,'DD/MM/YYYY');

    --Cursores de busqueda para el administrador

    --Todos
    CURSOR CUR_CLIENTE_AT IS
    SELECT codcliente,cedula,apellidos,nombres,telcasa,teltrabajo,celular,
        modelo,nombre,fechaatencion,estadogm,tipoclientegm,codvendedor,activo
    FROM   VISCONSULCLIENTES
    WHERE  ACTIVO = 1 ORDER BY APELLIDOS, NOMBRES;
    --Cedula

    CURSOR CUR_CLIENTE_AC IS
    SELECT codcliente,cedula,apellidos,nombres,telcasa,teltrabajo,celular,
        modelo,nombre,fechaatencion,estadogm,tipoclientegm,codvendedor,activo
    FROM   VISCONSULCLIENTES
    WHERE  ACTIVO = 1 AND CEDULA = P_CEDULA;

    --Apellidos
    CURSOR CUR_CLIENTE_AA IS
    SELECT codcliente,cedula,apellidos,nombres,telcasa,teltrabajo,celular,
        modelo,nombre,fechaatencion,estadogm,tipoclientegm,codvendedor,activo
    FROM   VISCONSULCLIENTES
    WHERE  ACTIVO = 1 AND APELLIDOS LIKE '%' || P_APELLIDOS || '%'
    ORDER  BY APELLIDOS;

    --Nombres
    CURSOR CUR_CLIENTE_AN IS
    SELECT codcliente,cedula,apellidos,nombres,telcasa,teltrabajo,celular,
        modelo,nombre,fechaatencion,estadogm,tipoclientegm,codvendedor,activo
    FROM   VISCONSULCLIENTES
    WHERE  ACTIVO = 1 AND NOMBRES LIKE '%' || P_NOMBRES || '%'
    ORDER  BY NOMBRES;

    --Por fechas
    CURSOR CUR_CLIENTE_AF IS
    SELECT codcliente,cedula,apellidos,nombres,telcasa,teltrabajo,celular,
        modelo,nombre,fechaatencion,estadogm,tipoclientegm,codvendedor,activo
    FROM   VISCONSULCLIENTES
    WHERE  ACTIVO = 1 AND FECHAATENCION BETWEEN P_FECINI AND
P_FECFIN
    ORDER  BY FECHAATENCION, APELLIDOS;
  BEGIN
      SELECT COUNT(CODUSUARIO) INTO CONTADOR
      FROM USUARIOS WHERE CARGO ='Vendedor' AND CODUSUARIO =
P_VENDEDOR;
      IF CONTADOR > 0 THEN
          ES_VENDEDOR := TRUE;
      END IF;
```

```
--Administrador
-Todos
IF P_OP = 'T' THEN
  OPEN CUR_CLIENTE_AT;
  CONTADOR := 1;
  LOOP
  FETCH CUR_CLIENTE_AT INTO
      CLIENTES_DATA(CONTADOR).CODCLIENTE,
      CLIENTES_DATA(CONTADOR).CEDULA,
      CLIENTES_DATA(CONTADOR).APELLIDOS,
      CLIENTES_DATA(CONTADOR).NOMBRES,
      CLIENTES_DATA(CONTADOR).TELCASA,
      CLIENTES_DATA(CONTADOR).TELTRABAJO,
      CLIENTES_DATA(CONTADOR).CELULAR,
      CLIENTES_DATA(CONTADOR).MODELO,
      CLIENTES_DATA(CONTADOR).NOMBRE,
      CLIENTES_DATA(CONTADOR).FECHAATENCION,
      CLIENTES_DATA(CONTADOR).ESTADOGM,
      CLIENTES_DATA(CONTADOR).TIPOCLIENTEGM,
      CLIENTES_DATA(CONTADOR).CODVENDEDOR,
      CLIENTES_DATA(CONTADOR).ACTIVO;
      IF ES_VENDEDOR = TRUE AND
        CLIENTES_DATA(CONTADOR).CODVENDEDOR != P_VENDEDOR THEN
        CLIENTES_DATA(CONTADOR).TELCASA     := 'NO DISP.';
        CLIENTES_DATA(CONTADOR).TELTRABAJO  := 'NO DISP.';
        CLIENTES_DATA(CONTADOR).CELULAR     := 'NO DISP.';
      END IF;
      EXIT WHEN CUR_CLIENTE_AT%NOTFOUND;
      CONTADOR := CONTADOR + 1;
  END LOOP;
  IF CUR_CLIENTE_AT%ISOPEN THEN
    CLOSE CUR_CLIENTE_AT;
  END IF;
END IF;

--Cedula
IF P_OP = 'C' THEN
  OPEN CUR_CLIENTE_AC;
  CONTADOR := 1;
  LOOP
  FETCH CUR_CLIENTE_AC INTO
      CLIENTES_DATA(CONTADOR).CODCLIENTE,
      CLIENTES_DATA(CONTADOR).CEDULA,
      CLIENTES_DATA(CONTADOR).APELLIDOS,
      CLIENTES_DATA(CONTADOR).NOMBRES,
      CLIENTES_DATA(CONTADOR).TELCASA,
      CLIENTES_DATA(CONTADOR).TELTRABAJO,
      CLIENTES_DATA(CONTADOR).CELULAR,
      CLIENTES_DATA(CONTADOR).MODELO,
      CLIENTES_DATA(CONTADOR).NOMBRE,
      CLIENTES_DATA(CONTADOR).FECHAATENCION,
      CLIENTES_DATA(CONTADOR).ESTADOGM,
      CLIENTES_DATA(CONTADOR).TIPOCLIENTEGM,
      CLIENTES_DATA(CONTADOR).CODVENDEDOR,
      CLIENTES_DATA(CONTADOR).ACTIVO;
      IF ES_VENDEDOR = TRUE AND
        CLIENTES_DATA(CONTADOR).CODVENDEDOR != P_VENDEDOR THEN
        CLIENTES_DATA(CONTADOR).TELCASA     := 'NO DISP.';
```

```
        CLIENTES_DATA(CONTADOR).TELTRABAJO  := 'NO DISP.';
        CLIENTES_DATA(CONTADOR).CELULAR     := 'NO DISP.';
      END IF;
      EXIT WHEN CUR_CLIENTE_AC%NOTFOUND;
      CONTADOR := CONTADOR + 1;
  END LOOP;
  IF CUR_CLIENTE_AC%ISOPEN THEN
    CLOSE CUR_CLIENTE_AC;
  END IF;
END IF;

--Apellidos
IF P_OP = 'A' THEN
  OPEN CUR_CLIENTE_AA;
  CONTADOR := 1;
  LOOP
  FETCH CUR_CLIENTE_AA INTO
      CLIENTES_DATA(CONTADOR).CODCLIENTE,
      CLIENTES_DATA(CONTADOR).CEDULA,
      CLIENTES_DATA(CONTADOR).APELLIDOS,
      CLIENTES_DATA(CONTADOR).NOMBRES,
      CLIENTES_DATA(CONTADOR).TELCASA,
      CLIENTES_DATA(CONTADOR).TELTRABAJO,
      CLIENTES_DATA(CONTADOR).CELULAR,
      CLIENTES_DATA(CONTADOR).MODELO,
      CLIENTES_DATA(CONTADOR).NOMBRE,
      CLIENTES_DATA(CONTADOR).FECHAATENCION,
      CLIENTES_DATA(CONTADOR).ESTADOGM,
      CLIENTES_DATA(CONTADOR).TIPOCLIENTEGM,
      CLIENTES_DATA(CONTADOR).CODVENDEDOR,
      CLIENTES_DATA(CONTADOR).ACTIVO;
      IF ES_VENDEDOR = TRUE AND
        CLIENTES_DATA(CONTADOR).CODVENDEDOR != P_VENDEDOR THEN
        CLIENTES_DATA(CONTADOR).TELCASA     := 'NO DISP.';
        CLIENTES_DATA(CONTADOR).TELTRABAJO  := 'NO DISP.';
        CLIENTES_DATA(CONTADOR).CELULAR     := 'NO DISP.';
      END IF;
      EXIT WHEN CUR_CLIENTE_AA%NOTFOUND;
      CONTADOR := CONTADOR + 1;
  END LOOP;
  IF CUR_CLIENTE_AA%ISOPEN THEN
    CLOSE CUR_CLIENTE_AA;
  END IF;
END IF;

--Nombres
IF P_OP = 'N' THEN
  OPEN CUR_CLIENTE_AN;
  CONTADOR := 1;
  LOOP
  FETCH CUR_CLIENTE_AN INTO
      CLIENTES_DATA(CONTADOR).CODCLIENTE,
      CLIENTES_DATA(CONTADOR).CEDULA,
      CLIENTES_DATA(CONTADOR).APELLIDOS,
      CLIENTES_DATA(CONTADOR).NOMBRES,
      CLIENTES_DATA(CONTADOR).TELCASA,
      CLIENTES_DATA(CONTADOR).TELTRABAJO,
      CLIENTES_DATA(CONTADOR).CELULAR,
      CLIENTES_DATA(CONTADOR).MODELO,
```

```
        CLIENTES_DATA(CONTADOR).NOMBRE,
        CLIENTES_DATA(CONTADOR).FECHAATENCION,
        CLIENTES_DATA(CONTADOR).ESTADOGM,
        CLIENTES_DATA(CONTADOR).TIPOCLIENTEGM,
        CLIENTES_DATA(CONTADOR).CODVENDEDOR,
        CLIENTES_DATA(CONTADOR).ACTIVO;
        IF ES_VENDEDOR = TRUE AND
          CLIENTES_DATA(CONTADOR).CODVENDEDOR != P_VENDEDOR THEN
          CLIENTES_DATA(CONTADOR).TELCASA     := 'NO DISP.';
          CLIENTES_DATA(CONTADOR).TELTRABAJO  := 'NO DISP.';
          CLIENTES_DATA(CONTADOR).CELULAR     := 'NO DISP.';
        END IF;
        EXIT WHEN CUR_CLIENTE_AN%NOTFOUND;
        CONTADOR := CONTADOR + 1;
    END LOOP;
    IF CUR_CLIENTE_AN%ISOPEN THEN
      CLOSE CUR_CLIENTE_AN;
    END IF;
  END IF;

  --Fechas
  IF P_OP = 'F' THEN
    OPEN CUR_CLIENTE_AF;
    CONTADOR := 1;
    LOOP
    FETCH CUR_CLIENTE_AF INTO
        CLIENTES_DATA(CONTADOR).CODCLIENTE,
        CLIENTES_DATA(CONTADOR).CEDULA,
        CLIENTES_DATA(CONTADOR).APELLIDOS,
        CLIENTES_DATA(CONTADOR).NOMBRES,
        CLIENTES_DATA(CONTADOR).TELCASA,
        CLIENTES_DATA(CONTADOR).TELTRABAJO,
        CLIENTES_DATA(CONTADOR).CELULAR,
        CLIENTES_DATA(CONTADOR).MODELO,
        CLIENTES_DATA(CONTADOR).NOMBRE,
        CLIENTES_DATA(CONTADOR).FECHAATENCION,
        CLIENTES_DATA(CONTADOR).ESTADOGM,
        CLIENTES_DATA(CONTADOR).TIPOCLIENTEGM,
        CLIENTES_DATA(CONTADOR).CODVENDEDOR,
        CLIENTES_DATA(CONTADOR).ACTIVO;
        IF ES_VENDEDOR = TRUE AND
          CLIENTES_DATA(CONTADOR).CODVENDEDOR != P_VENDEDOR THEN
          CLIENTES_DATA(CONTADOR).TELCASA     := 'NO DISP.';
          CLIENTES_DATA(CONTADOR).TELTRABAJO  := 'NO DISP.';
          CLIENTES_DATA(CONTADOR).CELULAR     := 'NO DISP.';
        END IF;
        EXIT WHEN CUR_CLIENTE_AF%NOTFOUND;
        CONTADOR := CONTADOR + 1;
    END LOOP;
    IF CUR_CLIENTE_AF%ISOPEN THEN
      CLOSE CUR_CLIENTE_AF;
    END IF;
  END IF;
 EXCEPTION
    WHEN OTHERS THEN

    --Cerrar cursores de administrador.
    IF CUR_CLIENTE_AC%ISOPEN THEN
      CLOSE CUR_CLIENTE_AC;
```

```
        END IF;
        IF CUR_CLIENTE_AN%ISOPEN THEN
          CLOSE CUR_CLIENTE_AN;
        END IF;
        IF CUR_CLIENTE_AA%ISOPEN THEN
          CLOSE CUR_CLIENTE_AA;
        END IF;
        IF CUR_CLIENTE_AF%ISOPEN THEN
          CLOSE CUR_CLIENTE_AF;
        END IF;
 END SP_CONSULTAGENERAL_FORMS;
```

-- PROCEDURE SP_CONSULTARELIMINADOS_FORMS

--Consulta los clientes eliminados para la pantalla papalera de reciclaje de clientes, si el id del usuario que consulta (p_vendedor) resulta ser el id de un vendedor, muestra los clientes de ese vendedor solamente, si el id es de un administrador muestra todos los clientes eliminados/*

```
 PROCEDURE SP_CONSULTARELIMINADOS_FORMS
 (
  --CONSULTA POR VENDEDOR LOS CLIENTES ELIMINADOS
  --SI P_VENDEDOR = -1 ENTONCES TRAE DE TODOS LOS VENDEDORES
  P_VENDEDOR       IN NUMBER   DEFAULT NULL,
  CLI_DATA         IN OUT      T_CLI_ELI
 )
 IS
  TIPO VARCHAR2(20);
  CURSOR CUR_ADMIN IS
  SELECT * FROM VIS_CLIENTES_ELIMINADOS
  WHERE ACTIVO = 0 ORDER BY APELLIDOS, NOMBRES;
  CURSOR CUR_VEND IS
  SELECT * FROM VIS_CLIENTES_ELIMINADOS
  WHERE ACTIVO = 0 AND CODVENDEDOR = P_VENDEDOR
  ORDER BY APELLIDOS, NOMBRES;
 BEGIN
  SELECT CARGO INTO TIPO FROM USUARIOS
  WHERE CODUSUARIO = P_VENDEDOR;
  CONTADOR := 0;
  IF TIPO = 'Administrador' THEN
    FOR FILA IN CUR_ADMIN
    LOOP
       CONTADOR := CONTADOR + 1;
       CLI_DATA(CONTADOR) := FILA;
    END LOOP;
    IF CUR_ADMIN%ISOPEN THEN
      CLOSE CUR_ADMIN;
    END IF;
  ELSE
    FOR FILA IN CUR_VEND
    LOOP
       CONTADOR := CONTADOR + 1;
       CLI_DATA(CONTADOR) := FILA;
    END LOOP;
    IF CUR_VEND%ISOPEN THEN
      CLOSE CUR_VEND;
    END IF;
  END IF;
 EXCEPTION
```

```
  WHEN OTHERS THEN
     IF CUR_ADMIN%ISOPEN THEN
       CLOSE CUR_ADMIN;
     END IF;
     IF CUR_VEND%ISOPEN THEN
       CLOSE CUR_VEND;
     END IF;
     RAISE_APPLICATION_ERROR(-20000, 'NO SE PUDIERON CONSULTAR LOS
     CLIENTES ELIMINADOS');
     ROLLBACK;
END SP_CONSULTARELIMINADOS_FORMS;


-- PROCEDURE SP_GETCREDITOS_FORMS
-- Obtiene los créditos del cliente, dentro del paquete clientes

PROCEDURE SP_GETCREDITOS_FORMS
(
  P_ID            IN NUMBER   DEFAULT NULL,
  CREDITOS_DATA    IN OUT T_CREDITO
)
IS
    CURSOR CUR_CREDITOS IS
    SELECT
codcredito,fechaing,fechares,VENDEDOR,banco,estado,modelo,precio,entrada,plazo,dispositi
vo,gastos,codusuario,
          CODCLIENTE,CODBANCO,CODVEHICULO,ACTIVO
    FROM   VisConsulCreditos
    WHERE  codcliente = P_ID
    ORDER  BY fechaing, banco;
BEGIN
   CONTADOR := 1;
   FOR FILA IN CUR_CREDITOS
   LOOP
       CREDITOS_DATA(CONTADOR).CODCREDITO     := FILA.CODCREDITO;
       CREDITOS_DATA(CONTADOR).FECHAING       := FILA.FECHAING;
       CREDITOS_DATA(CONTADOR).FECHARES       := FILA.FECHARES;
       CREDITOS_DATA(CONTADOR).VENDEDOR       := FILA.VENDEDOR;
       CREDITOS_DATA(CONTADOR).BANCO          := FILA.BANCO;
       CREDITOS_DATA(CONTADOR).ESTADO         := FILA.ESTADO;
       CREDITOS_DATA(CONTADOR).MODELO         := FILA.MODELO;
       CREDITOS_DATA(CONTADOR).PRECIO         := FILA.PRECIO;
       CREDITOS_DATA(CONTADOR).ENTRADA        := FILA.ENTRADA;
       CREDITOS_DATA(CONTADOR).PLAZO          := FILA.PLAZO;
       CREDITOS_DATA(CONTADOR).DISPOSITIVO    := FILA.DISPOSITIVO;
       CREDITOS_DATA(CONTADOR).GASTOS         := FILA.GASTOS;
       CREDITOS_DATA(CONTADOR).CODUSUARIO     := FILA.CODUSUARIO;
       CREDITOS_DATA(CONTADOR).CODCLIENTE     := FILA.CODCLIENTE;
       CREDITOS_DATA(CONTADOR).CODBANCO       := FILA.CODBANCO;
       CREDITOS_DATA(CONTADOR).CODVEHICULO    := FILA.CODVEHICULO;
       CREDITOS_DATA(CONTADOR).ACTIVO         := FILA.ACTIVO;
       CONTADOR := CONTADOR + 1;
   END LOOP;
   IF CUR_CREDITOS%ISOPEN THEN
       CLOSE CUR_CREDITOS;
   END IF;
   EXCEPTION
     WHEN OTHERS THEN
     --CERRAR CURSORES DE ADM.
```

```
      IF CUR_CREDITOS%ISOPEN THEN
        CLOSE CUR_CREDITOS;
      END IF;
END SP_GETCREDITOS_FORMS;

--PROCEDURE SP_GETVENTAS_FORMS
-- Obtiene las ventas del cliente, dentro del paquete clientes

PROCEDURE SP_GETVENTAS_FORMS
(
  P_ID            IN NUMBER   DEFAULT NULL,
  VENTAS_DATA     IN OUT      T_VENTAS
)
IS
    CURSOR CUR_VENTAS IS
    SELECT *
    FROM   VisConsulVentas
    WHERE  codcliente = P_ID
    ORDER  BY fechafac, banco;
BEGIN
   OPEN CUR_VENTAS;
   CONTADOR := 1;
   FETCH CUR_VENTAS INTO VENTAS_DATA(CONTADOR);
   WHILE CUR_VENTAS%FOUND
   LOOP
       CONTADOR := CONTADOR + 1;
       FETCH CUR_VENTAS INTO VENTAS_DATA(CONTADOR);
       --EXIT WHEN CUR_VENTAS%NOTFOUND;
   END LOOP;
   IF CUR_VENTAS%ISOPEN THEN
       CLOSE CUR_VENTAS;
   END IF;
   EXCEPTION
      WHEN OTHERS THEN

      --Cerrar cursores de administrador.
      IF CUR_VENTAS%ISOPEN THEN
        CLOSE CUR_VENTAS;
      END IF;
END SP_GETVENTAS_FORMS;


--PROCEDURE SP_GETAGENDA_FORMS
-- Obtiene todos los datos de la table Agenda del Cliente

PROCEDURE SP_GETAGENDA_FORMS
(
 P_ID            IN NUMBER   DEFAULT NULL,
 AGENDA_DATA     IN OUT T_AGENDA
)
IS
CURSOR CUR_AGENDA IS
SELECT codmensaje, codusuario, codcliente, fecha, hora, mensaje, respuesta
FROM   Agenda
WHERE  CODCLIENTE = P_ID Order By Hora;
BEGIN
   CONTADOR := 0;
   FOR FILA IN CUR_AGENDA
   LOOP
```

```
        CONTADOR := CONTADOR + 1;
        AGENDA_DATA(CONTADOR) := FILA;
      END LOOP;
      IF CUR_AGENDA%ISOPEN THEN
        CLOSE CUR_AGENDA;
      END IF;
    EXCEPTION
      WHEN OTHERS THEN

      --Cerrar cursores de administrador.
      IF CUR_AGENDA%ISOPEN THEN
        CLOSE CUR_AGENDA;
      END IF;
    END SP_GETAGENDA_FORMS;
END PK_CLIENTES;
/
```

- ***PK_CREDITOS.***

Objetivo: Administrar los Créditos de los Clientes.

Contiene los Procedimientos Almacenados para la Creación, Modificación, Eliminación y Consulta de la Tabla CREDITOS.

```
prompt Creating package body PK_CREDITOS
prompt ================================

CREATE OR REPLACE Package Body USER15.PK_CREDITOS IS

    NEW_ID          NUMBER;
    MSG             VARCHAR2(100);
    CONTADOR        NUMBER;
    EXP_ERROR       EXCEPTION;
    V_CARGO         VARCHAR2(20);

-- PROCEDURE SP_INSERTAR
/* Recibe todos los parametros de la tabla Créditos para insertar todos los datos necesarios para la solicitud de crédito, devuelve la secuencia o id.*/

PROCEDURE SP_INSERTAR
(
      P_CODCLIENTE     IN NUMBER     DEFAULT NULL,
      P_CODUSUARIO     IN NUMBER     DEFAULT NULL,
      P_CODBANCO       IN NUMBER     DEFAULT NULL,
      P_CODVEHICULO    IN NUMBER     DEFAULT NULL,
      P_CODDISPOSITIVO IN NUMBER     DEFAULT NULL,
      P_ESTADO         IN VARCHAR2   DEFAULT NULL,
      P_PRECIO         IN NUMBER     DEFAULT NULL,
      P_ENTRADA        IN NUMBER     DEFAULT NULL,
      P_PLAZO          IN NUMBER     DEFAULT NULL,
      P_GASTOS         IN NUMBER     DEFAULT NULL,
      P_FECHAING       IN DATE       DEFAULT NULL,
      P_FECHARES       IN DATE       DEFAULT NULL,
      P_DOCSCLI        IN VARCHAR2   DEFAULT NULL,
      P_DOCSCON        IN VARCHAR2   DEFAULT NULL,
      P_COMENTARIO     IN CLOB       DEFAULT NULL,
      P_ID             OUT NUMBER
)
```

```
IS
BEGIN
   MSG:= 'NO SE PUDO INSERTAR CREDITO';
   SELECT COUNT(CODCREDITO) INTO CONTADOR
   FROM CREDITOS WHERE CODCLIENTE = P_CODCLIENTE AND CODBANCO =
P_CODBANCO AND CODVEHICULO = P_CODVEHICULO
   AND FECHAING BETWEEN (SYSDATE - 30) AND SYSDATE;
   IF CONTADOR > 0 THEN
     MSG := 'YA EXISTE UN CREDITO DEL CLIENTE POR ESTE VEHICULO CON
ESTE BANCO EN LOS ULTIMOS 30 DÍAS';
     RAISE EXP_ERROR;
   END IF;
   SELECT NVL(MAX(CODCREDITO) + 1, 1) INTO NEW_ID FROM CREDITOS;
   INSERT INTO CREDITOS
        (codcredito,codcliente,codusuario,codbanco,codvehiculo,coddispositivo,
         estado,precio,entrada,plazo,gastos,fechaing,horaing,fechares,
         docscli,docscon,comentario)
   VALUES
(NEW_ID,P_CODCLIENTE,P_CODUSUARIO,P_CODBANCO,P_CODVEHICULO,P_CO
DDISPOSITIVO,P_ESTADO,P_PRECIO,P_ENTRADA,P_PLAZO,P_GASTOS,P_FECHAI
NG,SYSDATE,P_FECHARES, P_DOCSCLI,P_DOCSCON,P_COMENTARIO);
   COMMIT;
   SELECT NEW_ID INTO P_ID FROM DUAL;
   EXCEPTION
   WHEN EXP_ERROR THEN
      RAISE_APPLICATION_ERROR(-20000, MSG);
      ROLLBACK;
   WHEN OTHERS THEN
      RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO: ' || SQLERRM);
      ROLLBACK;
END SP_INSERTAR;

--PROCEDURE SP_GUARDAR
-- Guarda todos los datos de la tabla Créditos

PROCEDURE SP_GUARDAR
(
      P_CODCREDITO     IN NUMBER      DEFAULT NULL,
      P_CODCLIENTE     IN NUMBER      DEFAULT NULL,
      P_CODUSUARIO     IN NUMBER      DEFAULT NULL,
      P_CODBANCO       IN NUMBER      DEFAULT NULL,
      P_CODVEHICULO    IN NUMBER      DEFAULT NULL,
      P_CODDISPOSITIVO IN NUMBER      DEFAULT NULL,
      P_ESTADO         IN VARCHAR2    DEFAULT NULL,
      P_PRECIO         IN NUMBER      DEFAULT NULL,
      P_ENTRADA        IN NUMBER      DEFAULT NULL,
      P_PLAZO          IN NUMBER      DEFAULT NULL,
      P_GASTOS         IN NUMBER      DEFAULT NULL,
      P_FECHAING       IN DATE        DEFAULT NULL,
      P_FECHARES       IN DATE        DEFAULT NULL,
      P_DOCSCLI        IN VARCHAR2    DEFAULT NULL,
      P_DOCSCON        IN VARCHAR2    DEFAULT NULL,
      P_COMENTARIO     IN CLOB        DEFAULT NULL
)
IS
BEGIN
   MSG:= 'NO SE PUDO GUARDAR CREDITO';
   SELECT COUNT(CODCREDITO) INTO CONTADOR
```

```
   FROM CREDITOS WHERE CODCLIENTE = P_CODCLIENTE AND CODBANCO =
P_CODBANCO AND CODVEHICULO = P_CODVEHICULO
   AND FECHAING BETWEEN (SYSDATE - 30) AND SYSDATE AND CODCREDITO
<> P_CODCREDITO;
   IF CONTADOR > 0 THEN
      MSG := 'YA EXISTE UN CREDITO DEL CLIENTE POR ESTE VEHICULO CON
ESTE BANCO EN ESTE MES';
      RAISE EXP_ERROR;
   END IF;
   UPDATE CREDITOS
   SET    codcliente      = P_CODCLIENTE,
          codusuario      = P_CODUSUARIO,
          codbanco        = P_CODBANCO,
          codvehiculo     = P_CODVEHICULO,
          coddispositivo  = P_CODDISPOSITIVO,
          estado          = P_ESTADO,
          precio          = P_PRECIO,
          entrada         = P_ENTRADA,
          plazo           = P_PLAZO,
          gastos          = P_GASTOS,
          fechaing        = P_FECHAING,
          fechares        = P_FECHARES,
          docscli         = P_DOCSCLI,
          docscon         = P_DOCSCON,
          comentario      = P_COMENTARIO
   WHERE  CODCREDITO      = P_CODCREDITO;
   COMMIT;
   EXCEPTION
   WHEN EXP_ERROR THEN
      RAISE_APPLICATION_ERROR(-20000, MSG);
      ROLLBACK;
   WHEN OTHERS THEN
      RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO: ' || SQLERRM);
      ROLLBACK;
END SP_GUARDAR;

--PROCEDURE SP_ELIMINAR
-- Elimina un crédito de un cliente.

PROCEDURE SP_ELIMINAR
(
       P_CODCREDITO     IN NUMBER     DEFAULT NULL
)
IS
BEGIN
   MSG:= 'NO SE PUDO ELIMINAR CREDITO';
   SELECT COUNT(CODVENTA) INTO CONTADOR FROM VENTAS
   WHERE CODCREDITO = P_CODCREDITO;

   IF CONTADOR > 0 THEN
      MSG := 'EL CREDITO QUE DESEA ELIMINAR SE ENCUENTRA FACTURADO, NO
SE PUEDE ELIMINAR';
      RAISE EXP_ERROR;
   END IF;
   DELETE FROM CREDITOS WHERE CODCREDITO = P_CODCREDITO;
   COMMIT;
   EXCEPTION
   WHEN EXP_ERROR THEN
```

```
        RAISE_APPLICATION_ERROR(-20000, MSG);
        ROLLBACK;
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO');
        ROLLBACK;
END SP_ELIMINAR;

-- PROCEDURE SP_CONSULTAR
--Consultar por id ó código del crédito.

PROCEDURE SP_CONSULTAR
(
        P_CODCREDITO     IN NUMBER      DEFAULT NULL,
        MY_CURSOR        IN OUT         T_CURSOR
)
IS
BEGIN
    OPEN MY_CURSOR FOR
    SELECT * FROM CREDITOS WHERE CODCREDITO = P_CODCREDITO;
END SP_CONSULTAR;
--PROCEDURE SP_CONSULTARTODOS
--Consultar creditos por fecha y vendedor

PROCEDURE SP_CONSULTARTODOS
(
        --PARA FILTRAR POR VENDEDOR
        P_CODVENDEDOR    IN NUMBER      DEFAULT NULL,
        P_FECINI         IN DATE        DEFAULT NULL,
        P_FECFIN         IN DATE        DEFAULT NULL,
        --INDICA SI SE CONSULTA POR FECHAS
        P_OPFEC          IN CHAR        DEFAULT NULL,
        MY_CURSOR        IN OUT         T_CURSOR
)
IS
BEGIN
IF NVL(P_CODVENDEDOR,-1) = -1 THEN
       IF P_OPFEC = 'S' THEN

         --Sin vendedor con fechas
         OPEN MY_CURSOR FOR
         SELECT A.codcredito, A.fechaing, A.fechares, B.APELLIDOS || ' ' || B.NOMBRES AS
CLIENTE,
              A.vendedor, A.banco, A.estado, A.modelo,
              A.precio, A.entrada, A.plazo, A.dispositivo, A.gastos
         FROM       VISCONSULCREDITOS A
         INNER  JOIN CLIENTES B ON A.CodCliente = B.CODCLIENTE
         WHERE  A.FECHAING BETWEEN P_FECINI AND P_FECFIN;
       ELSE

         --Sin vendedor sin fechas
         OPEN MY_CURSOR FOR
         SELECT A.codcredito, A.fechaing, A.fechares, B.APELLIDOS || ' ' || B.NOMBRES AS
CLIENTE,
              A.vendedor, A.banco, A.estado, A.modelo,
              A.precio, A.entrada, A.plazo, A.dispositivo, A.gastos
         FROM       VISCONSULCREDITOS A
         INNER  JOIN CLIENTES B ON A.CodCliente = B.CODCLIENTE;
       END IF;
```

```
    ELSE
      IF P_OPFEC = 'S' THEN

        --Con vendedor con fechas
        OPEN MY_CURSOR FOR
        SELECT A.codcredito, A.fechaing, A.fechares, B.APELLIDOS || ' ' || B.NOMBRES AS
CLIENTE,
             A.vendedor, A.banco, A.estado, A.modelo,
             A.precio, A.entrada, A.plazo, A.dispositivo, A.gastos
        FROM       VISCONSULCREDITOS A
        INNER  JOIN CLIENTES B ON A.CodCliente = B.CODCLIENTE
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        AND    A.FECHAING BETWEEN P_FECINI AND P_FECFIN;
      ELSE

        --Con vendedor sin fechas
        OPEN MY_CURSOR FOR
        SELECT A.codcredito, A.fechaing, A.fechares, B.APELLIDOS || ' ' || B.NOMBRES AS
CLIENTE,
             A.vendedor, A.banco, A.estado, A.modelo,
             A.precio, A.entrada, A.plazo, A.dispositivo, A.gastos
        FROM       VISCONSULCREDITOS A
        INNER  JOIN CLIENTES B ON A.CodCliente = B.CODCLIENTE
        WHERE  A.CODUSUARIO = P_CODVENDEDOR;
      END IF;
    END IF;

END;


--PROCEDURE SP_CONSULTARTODOS_FORMS
-- Consulta todos los vendedores

PROCEDURE SP_CONSULTARTODOS_FORMS
(
        --Para filtrar por vendedor
        P_CODVENDEDOR     IN NUMBER      DEFAULT NULL,
        P_FECINI          IN DATE        DEFAULT NULL,
        P_FECFIN          IN DATE        DEFAULT NULL,
        --INDICA SI SE CONSULTA POR FECHAS
        P_OPFEC           IN CHAR        DEFAULT NULL,
        CRED_DATA         IN OUT         T_CREDITOS
)

IS
        --Sin vendedor con fechas
        CURSOR CUR_01 IS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.FECHAING BETWEEN P_FECINI AND P_FECFIN
        ORDER  BY A.Cliente;

        --Sin vendedor sin fechas
        CURSOR CUR_02 IS
        SELECT  *
        FROM    VISCONSULCREDITOS_CLIENTES A
        ORDER BY A.Cliente;

        --Con vendedor con fechas
```

```
        CURSOR CUR_03 IS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        AND    A.FECHAING BETWEEN P_FECINI AND P_FECFIN
        ORDER BY A.Cliente;

        --Con vendedor sin fechas
        CURSOR CUR_04 IS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        ORDER BY A.Cliente;
BEGIN
    CONTADOR := 0;
    SELECT CARGO INTO V_CARGO FROM USUARIOS
    WHERE CODUSUARIO = P_CODVENDEDOR;
    IF V_CARGO = 'Administrador' THEN
      IF P_OPFEC = 'S' THEN

        --Sin vendedor con fechas
        FOR FILA IN CUR_01 LOOP
           CONTADOR := CONTADOR + 1;
           CRED_DATA(CONTADOR) := FILA;
        END LOOP;
        IF CUR_01%ISOPEN THEN
          CLOSE CUR_01;
        END IF;
      ELSE

        --Sin vendedor sin fechas
        FOR FILA IN CUR_02 LOOP
           CONTADOR := CONTADOR + 1;
           CRED_DATA(CONTADOR) := FILA;
        END LOOP;
        IF CUR_02%ISOPEN THEN
          CLOSE CUR_02;
        END IF;
      END IF;
    ELSE
      IF P_OPFEC = 'S' THEN

        --Con vendedor con fechas
        FOR FILA IN CUR_03 LOOP
           CONTADOR := CONTADOR + 1;
           CRED_DATA(CONTADOR) := FILA;
        END LOOP;
        IF CUR_03%ISOPEN THEN
          CLOSE CUR_03;
        END IF;
      ELSE

        --Con vendedor sin fechas
        FOR FILA IN CUR_04 LOOP
           CONTADOR := CONTADOR + 1;
           CRED_DATA(CONTADOR) := FILA;
        END LOOP;
        IF CUR_04%ISOPEN THEN
          CLOSE CUR_04;
```

```
        END IF;
      END IF;
    END IF;

END SP_CONSULTARTODOS_FORMS;


--PROCEDURE SP_CONSULTARPORVENDEDOR_FORMS
--Consulta por el nombre de un vendedor específico

PROCEDURE SP_CONSULTARPORVENDEDOR_FORMS
(
        --Para filtrar por vendedor
        P_CODVENDEDOR     IN NUMBER      DEFAULT NULL,
        P_FECINI          IN DATE        DEFAULT NULL,
        P_FECFIN          IN DATE        DEFAULT NULL,
        --INDICA SI SE CONSULTA POR FECHAS
        P_OPFEC           IN CHAR        DEFAULT NULL,
        CRED_DATA         IN OUT         T_CREDITOS
)

IS
        --Con vendedor con fechas
        CURSOR CUR_03 IS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        AND    A.FECHAING BETWEEN P_FECINI AND P_FECFIN
        ORDER BY A.Cliente;

        --Con vendedor sin fechas
        CURSOR CUR_04 IS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        ORDER BY A.Cliente;
BEGIN
   CONTADOR := 0;
   IF P_OPFEC = 'S' THEN

      --Con vendedor con fechas
      FOR FILA IN CUR_03 LOOP
         CONTADOR := CONTADOR + 1;
         CRED_DATA(CONTADOR) := FILA;
      END LOOP;
      IF CUR_03%ISOPEN THEN
        CLOSE CUR_03;
      END IF;
   ELSE

      --Con vendedor sin fechas
      FOR FILA IN CUR_04 LOOP
         CONTADOR := CONTADOR + 1;
         CRED_DATA(CONTADOR) := FILA;
      END LOOP;
      IF CUR_04%ISOPEN THEN
        CLOSE CUR_04;
      END IF;
   END IF;
```

```
END SP_CONSULTARPORVENDEDOR_FORMS;

 --PROCEDURE SP_CONSULTARPORBANCO
--Consultar por bancos (con fecha y vendedor opcionales)

PROCEDURE SP_CONSULTARPORBANCO_FORMS
(
        P_CODBANCO       IN NUMBER     DEFAULT NULL,
        --PARA FILTRAR POR VENDEDOR
        P_CODVENDEDOR    IN NUMBER     DEFAULT NULL,
        P_FECINI         IN DATE       DEFAULT NULL,
        P_FECFIN         IN DATE       DEFAULT NULL,
        --INDICA SI SE CONSULTA POR FECHAS
        P_OPFEC          IN CHAR       DEFAULT NULL,
        CRED_DATA        IN OUT        T_CREDITOS
)
IS
        --Sin vendedor con fechas
        CURSOR CUR_01 IS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.FECHAING BETWEEN P_FECINI AND P_FECFIN
        AND    A.CODBANCO = P_CODBANCO
        ORDER  BY A.Cliente;

        --Sin vendedor sin fechas
        CURSOR CUR_02 IS
        SELECT  *
        FROM    VISCONSULCREDITOS_CLIENTES A
        WHERE   A.CODBANCO = P_CODBANCO
        ORDER BY A.Cliente;

        --Con vendedor con fechas
        CURSOR CUR_03 IS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        AND    A.FECHAING BETWEEN P_FECINI AND P_FECFIN
        AND    A.CODBANCO = P_CODBANCO
        ORDER BY A.Cliente;

        --Con vendedor sin fechas
        CURSOR CUR_04 IS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        AND    A.CODBANCO = P_CODBANCO
        ORDER BY A.Cliente;
BEGIN
    CONTADOR := 0;
    SELECT CARGO INTO V_CARGO FROM USUARIOS
    WHERE CODUSUARIO = P_CODVENDEDOR;
    IF V_CARGO = 'Administrador' THEN
      IF P_OPFEC = 'S' THEN

         --Sin vendedor con fechas
         FOR FILA IN CUR_01 LOOP
            CONTADOR := CONTADOR + 1;
            CRED_DATA(CONTADOR) := FILA;
```

```
        END LOOP;
        IF CUR_01%ISOPEN THEN
          CLOSE CUR_01;
        END IF;
      ELSE

        --Sin vendedor sin fechas
        FOR FILA IN CUR_02 LOOP
           CONTADOR := CONTADOR + 1;
           CRED_DATA(CONTADOR) := FILA;
        END LOOP;
        IF CUR_02%ISOPEN THEN
          CLOSE CUR_02;
        END IF;
      END IF;
    ELSE
      IF P_OPFEC = 'S' THEN

        --Con vendedor con fechas
        FOR FILA IN CUR_03 LOOP
           CONTADOR := CONTADOR + 1;
           CRED_DATA(CONTADOR) := FILA;
        END LOOP;
        IF CUR_03%ISOPEN THEN
          CLOSE CUR_03;
        END IF;
      ELSE

        --Con vendedor sin fechas
        FOR FILA IN CUR_04 LOOP
           CONTADOR := CONTADOR + 1;
           CRED_DATA(CONTADOR) := FILA;
        END LOOP;
        IF CUR_04%ISOPEN THEN
          CLOSE CUR_04;
        END IF;
      END IF;
    END IF;
END SP_CONSULTARPORBANCO_FORMS;


--PROCEDURE SP_CONSULTARPORESTADO_FORMS
--Consultar por estados de credito (con fecha y vendedor opcionales)
/* Consulta según el estado en que se encuentra el crédito. Los estados son: Sin
   Respuesta, Aprobado, Pre-aprobado, Anulado, Con Respuesta*/

PROCEDURE SP_CONSULTARPORESTADO_FORMS
(
       P_ESTADO        IN VARCHAR2    DEFAULT NULL,
       --Para filtrar por vendedor
       P_CODVENDEDOR    IN NUMBER     DEFAULT NULL,
       P_FECINI         IN DATE       DEFAULT NULL,
       P_FECFIN         IN DATE       DEFAULT NULL,

       --Indica si se consulta por fechas
       P_OPFEC          IN CHAR       DEFAULT NULL,
       CRED_DATA        IN OUT        T_CREDITOS
)
```

```
IS
        --Sin vendedor con fechas
        CURSOR CUR_01 IS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.FECHAING BETWEEN P_FECINI AND P_FECFIN
        AND    A.Estado = P_ESTADO
        ORDER  BY A.Cliente;

        --Sin vendedor sin fechas
        CURSOR CUR_02 IS
        SELECT  *
        FROM    VISCONSULCREDITOS_CLIENTES A
        WHERE   A.Estado = P_ESTADO
        ORDER BY A.Cliente;

        --Con vendedor con fechas
        CURSOR CUR_03 IS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        AND    A.FECHAING BETWEEN P_FECINI AND P_FECFIN
        AND    A.Estado = P_ESTADO
        ORDER BY A.Cliente;

        --Con vendedor sin fechas
        CURSOR CUR_04 IS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        AND    A.Estado = P_ESTADO
        ORDER BY A.Cliente;
 BEGIN
    CONTADOR := 0;
    SELECT CARGO INTO V_CARGO FROM USUARIOS
    WHERE CODUSUARIO = P_CODVENDEDOR;
    IF V_CARGO = 'Administrador' THEN
      IF P_OPFEC = 'S' THEN

        --Sin vendedor con fechas
        FOR FILA IN CUR_01 LOOP
           CONTADOR := CONTADOR + 1;
           CRED_DATA(CONTADOR) := FILA;
        END LOOP;
        IF CUR_01%ISOPEN THEN
          CLOSE CUR_01;
        END IF;
      ELSE

        --Sin vendedor sin fechas
        FOR FILA IN CUR_02 LOOP
           CONTADOR := CONTADOR + 1;
           CRED_DATA(CONTADOR) := FILA;
        END LOOP;
        IF CUR_02%ISOPEN THEN
          CLOSE CUR_02;
        END IF;
      END IF;
    ELSE
```

```
      IF P_OPFEC = 'S' THEN

        --Con vendedor con fechas
        FOR FILA IN CUR_03 LOOP
           CONTADOR := CONTADOR + 1;
           CRED_DATA(CONTADOR) := FILA;
        END LOOP;
        IF CUR_03%ISOPEN THEN
          CLOSE CUR_03;
        END IF;
      ELSE

        --Con vendedor sin fechas
        FOR FILA IN CUR_04 LOOP
           CONTADOR := CONTADOR + 1;
           CRED_DATA(CONTADOR) := FILA;
        END LOOP;
        IF CUR_04%ISOPEN THEN
          CLOSE CUR_04;
        END IF;
      END IF;
    END IF;
END SP_CONSULTARPORESTADO_FORMS;

--PROCEDURE SP_CONSULTARPORMODELO_FORMS
--Consultar por modelo de vehiculo (con fecha y vendedor opcionales)

PROCEDURE SP_CONSULTARPORMODELO_FORMS
(
      P_CODVEHICULO    IN NUMBER     DEFAULT NULL,

      --Para filtrar por vendedor
      P_CODVENDEDOR    IN NUMBER     DEFAULT NULL,
      P_FECINI         IN DATE       DEFAULT NULL,
      P_FECFIN         IN DATE       DEFAULT NULL,

      --Indica si se consulta por fechas
      P_OPFEC          IN CHAR       DEFAULT NULL,
      CRED_DATA        IN OUT        T_CREDITOS
)
IS
      --Sin vendedor con fechas
      CURSOR CUR_01 IS
      SELECT *
      FROM   VISCONSULCREDITOS_CLIENTES A
      WHERE  A.FECHAING BETWEEN P_FECINI AND P_FECFIN
      AND    A.CodVehiculo = P_CODVEHICULO
      ORDER  BY A.Cliente;

      --Sin vendedor sin fechas
      CURSOR CUR_02 IS
      SELECT  *
      FROM    VISCONSULCREDITOS_CLIENTES A
      WHERE   A.CodVehiculo = P_CODVEHICULO
      ORDER BY A.Cliente;

      --Con vendedor con fechas
      CURSOR CUR_03 IS
      SELECT *
```

```
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        AND    A.FECHAING BETWEEN P_FECINI AND P_FECFIN
        AND    A.CodVehiculo = P_CODVEHICULO
        ORDER BY A.Cliente;

        --Con vendedor sin fechas
        CURSOR CUR_04 IS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        AND    A.CodVehiculo = P_CODVEHICULO
        ORDER BY A.Cliente;
BEGIN
    CONTADOR := 0;
    SELECT CARGO INTO V_CARGO FROM USUARIOS
    WHERE CODUSUARIO = P_CODVENDEDOR;
    IF V_CARGO = 'Administrador' THEN
      IF P_OPFEC = 'S' THEN

        --Sin vendedor con fechas
        FOR FILA IN CUR_01 LOOP
           CONTADOR := CONTADOR + 1;
           CRED_DATA(CONTADOR) := FILA;
        END LOOP;
        IF CUR_01%ISOPEN THEN
          CLOSE CUR_01;
        END IF;
      ELSE

        --Sin vendedor sin fechas
        FOR FILA IN CUR_02 LOOP
           CONTADOR := CONTADOR + 1;
           CRED_DATA(CONTADOR) := FILA;
        END LOOP;
        IF CUR_02%ISOPEN THEN
          CLOSE CUR_02;
        END IF;
      END IF;
    ELSE
      IF P_OPFEC = 'S' THEN

        --Con vendedor con fechas
        FOR FILA IN CUR_03 LOOP
           CONTADOR := CONTADOR + 1;
           CRED_DATA(CONTADOR) := FILA;
        END LOOP;
        IF CUR_03%ISOPEN THEN
          CLOSE CUR_03;
        END IF;
      ELSE

        --Con vendedor sin fechas
        FOR FILA IN CUR_04 LOOP
           CONTADOR := CONTADOR + 1;
           CRED_DATA(CONTADOR) := FILA;
        END LOOP;
        IF CUR_04%ISOPEN THEN
          CLOSE CUR_04;
```

```
        END IF;
      END IF;
    END IF;
END SP_CONSULTARPORMODELO_FORMS;

-- PROCEDURE SP_FACTURAR_CREDITO
-- Permite llenar la factura del cliente que aprobó el crédito

PROCEDURE SP_FACTURAR_CREDITO
(
        P_CODSEGURO      IN NUMBER      DEFAULT NULL,
        P_FACTURA        IN VARCHAR2    DEFAULT NULL,
        P_FECHAFAC       IN DATE        DEFAULT NULL,
        P_RAMV           IN VARCHAR2    DEFAULT NULL,
        P_MOTOR          IN VARCHAR2    DEFAULT NULL,
        P_CHASIS         IN VARCHAR2    DEFAULT NULL,
        P_COLOR          IN VARCHAR2    DEFAULT NULL,
        P_ANIO           IN NUMBER      DEFAULT NULL,
        P_ACCESORIOS     IN CLOB        DEFAULT NULL,
        P_CODCREDITO     IN NUMBER      DEFAULT NULL
)
IS
        V_ESTADO         VARCHAR2(20);
BEGIN
   SELECT COUNT(CODCREDITO) INTO CONTADOR FROM VENTAS WHERE
CODCREDITO = P_CODCREDITO;
   IF CONTADOR > 0 THEN
        MSG := 'EL CREDITO YA SE ENCUENTRA FACTURADO, NO SE PUEDE
FACTURAR 2 VECES';
        RAISE EXP_ERROR;
   END IF;
   SELECT ESTADO INTO V_ESTADO FROM CREDITOS WHERE CODCREDITO =
P_CODCREDITO;
   IF V_ESTADO <> 'Aprobado' THEN
        MSG := 'EL CREDITO NO SE PUEDE FACTURAR DEBIDO A QUE NO SE
ENCUENTRA APROBADO';
        RAISE EXP_ERROR;
   END IF;
   SELECT NVL(MAX(CODVENTA) + 1, 1) INTO NEW_ID FROM VENTAS;
   INSERT INTO ventas
   SELECT NEW_ID, codcliente, codusuario, codbanco,
          codvehiculo, coddispositivo, P_CODSEGURO,
          P_FACTURA, precio, entrada, plazo, gastos,
          P_FECHAFAC, NULL, P_RAMV, P_MOTOR, P_CHASIS,
          P_COLOR, P_ANIO, P_ACCESORIOS, P_CODCREDITO
   FROM   CREDITOS
   WHERE  CODCREDITO = P_CODCREDITO;
   COMMIT;
   EXCEPTION
   WHEN EXP_ERROR THEN
        RAISE_APPLICATION_ERROR(-20000, MSG);
        ROLLBACK;
   WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO: ' || SQLERRM);
        ROLLBACK;
END SP_FACTURAR_CREDITO;
END PK_CREDITOS;
```

❖ **PK_DEALER.**

Objetivo: Manejar los Datos de la Tabla Dealer que lleva el Nombre del Concesionario que usarà el sistema Terrasoft.

Este package sirve para asignarle un nombre al concesionario que adquiera y utilice el sistema Terrasoft.

```
prompt Creating package body PK_DEALER
prompt ==============================

CREATE OR REPLACE Package Body USER15.PK_DEALER IS

 MSG            VARCHAR(100);
 EXP_ERROR      EXCEPTION;

 --PROCEDURE SET_CONCESIONARIO
/* Graba y devuelve los datos como el nombre del concesionario que usará el sistema
   Terrasoft*/

 PROCEDURE SET_CONCESIONARIO
 (
       P_NOMBRE        IN VARCHAR2 DEFAULT NULL
 )
 IS
 BEGIN
    MSG:= 'NO SE PUDO ESTABLECER EL NOMBRE DEL CONCESIONARIO';
    DELETE FROM DEALER;

    --Tabla de un solo registro
    INSERT INTO DEALER (NOMBRE) VALUES (P_NOMBRE);
    COMMIT;
    EXCEPTION
       WHEN EXP_ERROR THEN
          RAISE_APPLICATION_ERROR(-20000, MSG);
          ROLLBACK;
       WHEN OTHERS THEN
          RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO');
          ROLLBACK;
 END SET_CONCESIONARIO;

 -- PROCEDURE GET_CONCESIONARIO
-- Obtiene los datos del concesionario.

 PROCEDURE GET_CONCESIONARIO
 (
       MY_CURSOR       OUT       T_CURSOR
 )
 IS
 BEGIN
```

```
    OPEN MY_CURSOR FOR
    SELECT NOMBRE FROM DEALER;

  END GET_CONCESIONARIO;
END PK_DEALER;
/
```

- ***PK_DISPOSITIVOS.***

Objetivo: Administrar dispositivos de seguridad

Contiene los Procedimientos Almacenados para la Creación, Modificación, Eliminación y Consulta de la Tabla DISPOSITIVOS, que almacena los datos de los Disp. de Seguridad que se pueden financiar en un crédito.

```
prompt Creating package body PK_DISPOSITIVOS
prompt ====================================

CREATE OR REPLACE Package Body USER15.PK_DISPOSITIVOS IS
    NEW_ID          NUMBER;
    MSG             VARCHAR2(100);
    CONTADOR        NUMBER;
    EXP_ERROR       EXCEPTION;

    -- PROCEDURE SP_INSERTAR
    --Permite inserter los datos del dispositvo como el nombre,precio y código.

    PROCEDURE SP_INSERTAR
    (
     P_NOMBRE         IN VARCHAR2 DEFAULT NULL,
     P_PRECIO         IN NUMBER   DEFAULT NULL,
     P_ID             OUT NUMBER
    )
    IS
    BEGIN
       MSG:= 'NO SE PUDO INSERTAR DISPOSITIVO DE SEGURIDAD';
       SELECT COUNT(*) INTO CONTADOR FROM DISPOSITIVOS
       WHERE NOMBRE LIKE P_NOMBRE AND ACTIVO = 1;
       IF CONTADOR > 0 THEN
         MSG := 'EL NOMBRE DE ESTE DISPOSITIVO YA EXISTE';
         RAISE EXP_ERROR;
       END IF;
       SELECT NVL(MAX(CODDISPOSITIVO) + 1, 1) INTO NEW_ID FROM
DISPOSITIVOS;
       INSERT INTO DISPOSITIVOS
       (coddispositivo,nombre,precio,activo)
       VALUES
       (NEW_ID,P_NOMBRE,P_PRECIO,1);
       SELECT NEW_ID INTO P_ID FROM DUAL;
       COMMIT;
       EXCEPTION
       WHEN EXP_ERROR THEN
          RAISE_APPLICATION_ERROR(-20000, MSG);
          ROLLBACK;
```

```
        WHEN OTHERS THEN
            RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO');
            ROLLBACK;
    END SP_INSERTAR;

    -- PROCEDURE SP_GUARDAR
    -- Guarda todos los datos del dispositivo en su tabla respectiva.

    PROCEDURE SP_GUARDAR
    (
     P_ID            IN NUMBER DEFAULT NULL,
     P_NOMBRE          IN VARCHAR2 DEFAULT NULL,
     P_PRECIO         IN NUMBER DEFAULT NULL,
     P_ACTIVO          IN NUMBER DEFAULT NULL
    )
    IS
    BEGIN
        MSG:= 'NO SE PUDO GUARDAR DISPOSITIVO DE SEGURIDAD';
        SELECT COUNT(*) INTO CONTADOR FROM DISPOSITIVOS
        WHERE CODDISPOSITIVO = P_ID;
        IF CONTADOR = 0 THEN
          MSG:= 'NO SE PUDO ENCONTRAR EL DISPOSITIVO A MODIFICAR';
          RAISE EXP_ERROR;
        END IF;
        SELECT COUNT(*) INTO CONTADOR FROM DISPOSITIVOS
        WHERE NOMBRE LIKE P_NOMBRE AND ACTIVO = 1 AND CODDISPOSITIVO
<> P_ID;
        IF CONTADOR > 0 THEN
          MSG:= 'EL NOMBRE YA ESTA ASIGNADO A OTRA MARCA';
          RAISE EXP_ERROR;
        END IF;
        UPDATE DISPOSITIVOS SET
        NOMBRE = P_NOMBRE,
        PRECIO = P_PRECIO,
        ACTIVO = P_ACTIVO
        WHERE CODDISPOSITIVO = P_ID;
        COMMIT;
        EXCEPTION
        WHEN EXP_ERROR THEN
            RAISE_APPLICATION_ERROR(-20000, MSG);
            ROLLBACK;
        WHEN OTHERS THEN
            RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO');
            ROLLBACK;
    END SP_GUARDAR;

    -- PROCEDURE SP_ELIMINAR
    --Elimina un dispositvo enviándolo a la papelera de reciclaje y asignándole un estado
      activo o inactivo.

    PROCEDURE SP_ELIMINAR
    (
     P_ID            IN NUMBER DEFAULT NULL,
     P_ACTIVO          IN NUMBER DEFAULT NULL
    )
    IS
    BEGIN
```

```
        MSG:= 'NO SE PUDO ELIMINAR DISPOSITIVO DE SEGURIDAD';
        SELECT COUNT(*) INTO CONTADOR FROM DISPOSITIVOS
        WHERE CODDISPOSITIVO = P_ID;
        IF CONTADOR = 0 THEN
          MSG:= 'NO SE PUDO ENCONTRAR EL DISPOSITIVO';
          RAISE EXP_ERROR;
        END IF;

        --Valido para eliminar y restaurar dependiendo del parametro
        UPDATE DISPOSITIVOS SET
        ACTIVO = P_ACTIVO
        WHERE CODDISPOSITIVO = P_ID;
        COMMIT;
        EXCEPTION
        WHEN EXP_ERROR THEN
            RAISE_APPLICATION_ERROR(-20000, MSG);
            ROLLBACK;
        WHEN OTHERS THEN
            RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO');
            ROLLBACK;
    END SP_ELIMINAR;

    --PROCEDURE SP_CONSULTAR
    -- Consulta un dispositivo según su código respectivo

    PROCEDURE SP_CONSULTAR
    (
     P_ID             IN NUMBER DEFAULT NULL,
     MY_CURSOR        IN OUT T_CURSOR
    )
    IS
    BEGIN
        OPEN MY_CURSOR FOR
        SELECT * FROM DISPOSITIVOS WHERE CODDISPOSITIVO = P_ID;
    END SP_CONSULTAR;


    --PROCEDURE SP_CONSULTARPORNOMBRE
    -- Consulta por un nombre específico de dispositivo

    PROCEDURE SP_CONSULTARPORNOMBRE
    (
     P_NOMBRE         IN VARCHAR2 DEFAULT NULL,
     MY_CURSOR        IN OUT T_CURSOR
    )
    IS
    BEGIN
        OPEN MY_CURSOR FOR
        SELECT * FROM DISPOSITIVOS WHERE NOMBRE LIKE '%' || P_NOMBRE || '%'
AND ACTIVO = 1
        ORDER BY NOMBRE;
    END SP_CONSULTARPORNOMBRE;

    --PROCEDURE SP_CONSULTARTODOS_FORMS
    --Consulta todos los dispositivos de la base de datos

    PROCEDURE SP_CONSULTARTODOS_FORMS
    (
```

```
        DISP_DATA   IN OUT T_DISPOSITIVO,
        P_ACTIVO    IN NUMBER     DEFAULT NULL,
        P_NOMBRE    IN VARCHAR2   DEFAULT NULL
       )
       IS
          CURSOR CUR_DISP IS
          SELECT * FROM DISPOSITIVOS WHERE ACTIVO = P_ACTIVO
          AND NOMBRE LIKE '%' || P_NOMBRE || '%'
          ORDER BY NOMBRE;
       BEGIN
          OPEN CUR_DISP;
          CONTADOR := 1;
          LOOP
          FETCH CUR_DISP INTO
              DISP_DATA(CONTADOR).CODDISPOSITIVO,
              DISP_DATA(CONTADOR).NOMBRE,
              DISP_DATA(CONTADOR).PRECIO,
              DISP_DATA(CONTADOR).ACTIVO;
              EXIT WHEN CUR_DISP%NOTFOUND;
              CONTADOR := CONTADOR + 1;
          END LOOP;
          IF CUR_DISP%ISOPEN THEN
            CLOSE CUR_DISP;
          END IF;
       EXCEPTION
          WHEN OTHERS THEN
          IF CUR_DISP%ISOPEN THEN
            CLOSE CUR_DISP;
          END IF;
       END SP_CONSULTARTODOS_FORMS;
  END PK_DISPOSITIVOS;
  /
```

- **PK_LOGIN.**

  Objetivo: Este package contiene los Procedimientos Almacenados para la comparación del ingreso de los usuarios con los almacenados en la Base de Datos.

```
prompt Creating package body PK_LOGIN
prompt ============================
CREATE OR REPLACE PACKAGE BODY USER15.PK_LOGIN IS

--PROCEDURE SP_LOGIN
/* Validará el estado y la clave del usuario.El estado serà igual a 1 si es Activo e igual a 0 si es
Inactivo.*/

PROCEDURE SP_LOGIN
  (
        MY_CURSOR OUT T_CURSOR,
        P_USUARIO VARCHAR2,
        P_CLAVE VARCHAR2
  )
  IS
  BEGIN
    OPEN MY_CURSOR FOR
    SELECT CODUSUARIO, USUARIO, CLAVE, NOMBRE, CARGO
    FROM USUARIOS WHERE ACTIVO=1 AND USUARIO = P_USUARIO AND CLAVE =
P_CLAVE;
```

```
 END SP_LOGIN;
END PK_LOGIN;
/
```

❖ ***PK_MARCAS.***

Objetivo: Administrar marcas de vehículos

Contiene los Procedimientos Almacenados para la Creación, Modificación, Eliminación y Consulta de la Tabla MARCAS, que almacena los datos de las marcas de Vehículos.

```
prompt Creating package body PK_MARCAS
prompt ==============================

CREATE OR REPLACE Package Body USER15.PK_MARCAS IS

    NEW_ID          NUMBER;
    MSG             VARCHAR2(100);
    CONTADOR        NUMBER;
    EXP_ERROR       EXCEPTION;




   --PROCEDURE SP_INSERTAR
   --Permite ingresar el nombre y el id o código de la marca de un vehículo

    PROCEDURE SP_INSERTAR
    (
     P_NOMBRE         IN VARCHAR2 DEFAULT NULL,
     P_ID             OUT NUMBER
    )
    IS
    BEGIN
        MSG:= 'NO SE PUDO INSERTAR MARCA DE VEHÍCULO';
        SELECT COUNT(*) INTO CONTADOR FROM marcas
        WHERE NOMBRE LIKE P_NOMBRE AND ACTIVO = 1;
        IF CONTADOR > 0 THEN
          MSG := 'EL NOMBRE DE ESTA MARCA YA EXISTE';
          RAISE EXP_ERROR;
        END IF;
        SELECT NVL(MAX(CODMARCA) + 1, 1) INTO NEW_ID FROM marcas;
        INSERT INTO marcas
        (codmarca,nombre,activo)
        VALUES
        (NEW_ID,P_NOMBRE,1);
        SELECT NEW_ID INTO P_ID FROM DUAL;
        COMMIT;
        EXCEPTION
        WHEN EXP_ERROR THEN
            RAISE_APPLICATION_ERROR(-20000, MSG);
            ROLLBACK;
        WHEN OTHERS THEN
            RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO');
            ROLLBACK;
    END SP_INSERTAR;
```

-- PROCEDURE SP_GUARDAR

-- Guardará la marca de un vehículo dentro de la tabla Marcas

```
PROCEDURE SP_GUARDAR
(
 P_ID            IN NUMBER DEFAULT NULL,
 P_NOMBRE          IN VARCHAR2 DEFAULT NULL,
 P_ACTIVO          IN NUMBER DEFAULT NULL
)
IS
BEGIN
    MSG:= 'NO SE PUDO GUARDAR MARCA DE VEHÍCULO';
    SELECT COUNT(*) INTO CONTADOR FROM marcas
    WHERE CODMARCA = P_ID;
    IF CONTADOR = 0 THEN
      MSG:= 'NO SE PUDO ENCONTRAR LA MARCA A MODIFICAR';
      RAISE EXP_ERROR;
    END IF;
    SELECT COUNT(*) INTO CONTADOR FROM marcas
    WHERE NOMBRE LIKE P_NOMBRE AND ACTIVO = 1 AND CODMARCA <>
P_ID;
    IF CONTADOR > 0 THEN
      MSG:= 'EL NOMBRE YA ESTA ASIGNADO A OTRA MARCA';
      RAISE EXP_ERROR;
    END IF;
    UPDATE MARCAS SET
    NOMBRE = P_NOMBRE,
    ACTIVO = P_ACTIVO
    WHERE CODMARCA = P_ID;
    COMMIT;
    EXCEPTION
    WHEN EXP_ERROR THEN
        RAISE_APPLICATION_ERROR(-20000, MSG);
        ROLLBACK;
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO');
        ROLLBACK;

END SP_GUARDAR;
```

-- PROCEDURE SP_ELIMINAR

/ Los datos no se eliminan de la base de datos mas bien se activan e inactivan. Cada tabla tiene su campo activo =0 e inactivo =1. La información que se elimina se va a una papelera de reciclaje y luego si desea restaurarlo también lo podrá hacer.*/*

```
PROCEDURE SP_ELIMINAR
(
 P_ID            IN NUMBER DEFAULT NULL,
 P_ACTIVO          IN NUMBER DEFAULT NULL
)
IS
BEGIN
    MSG:= 'NO SE PUDO ELIMINAR MARCA DE VEHÍCULO';
    SELECT COUNT(*) INTO CONTADOR FROM marcas
    WHERE CODMARCA = P_ID;
    IF CONTADOR = 0 THEN
      MSG:= 'NO SE PUDO ENCONTRAR LA MARCA';
      RAISE EXP_ERROR;
```

```
        END IF;
        SELECT COUNT(*) INTO CONTADOR FROM VEHICULOS
        WHERE CODMARCA = P_ID AND ACTIVO=1;
        IF CONTADOR > 0 AND P_ACTIVO = 0 THEN
          MSG:= 'NO SE PUEDE ELIMINAR LA MARCA DEBIDO A QUE HAY
VEHICULOS ACTIVOS ASIGNADOS';
          RAISE EXP_ERROR;
        END IF;

        --Valido para eliminar y restaurar dependiendo del parametro
        UPDATE MARCAS SET
        ACTIVO = P_ACTIVO
        WHERE CODMARCA = P_ID;
        COMMIT;
        EXCEPTION
        WHEN EXP_ERROR THEN
           RAISE_APPLICATION_ERROR(-20000, MSG);
           ROLLBACK;
        WHEN OTHERS THEN
           RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO');
           ROLLBACK;
    END SP_ELIMINAR;

   -- PROCEDURE SP_CONSULTAR
   -- Permite consultar según el código de la marca

    PROCEDURE SP_CONSULTAR
    (
     P_ID            IN NUMBER DEFAULT NULL,
     MY_CURSOR       IN OUT T_CURSOR
    )
    IS
    BEGIN
       OPEN MY_CURSOR FOR
       SELECT * FROM MARCAS WHERE CODMARCA = P_ID;
    END SP_CONSULTAR;




    -- PROCEDURE SP_CONSULTARPORNOMBRE
    -- Consulta por el nombre de una marca específica.

    PROCEDURE SP_CONSULTARPORNOMBRE
    (
     P_NOMBRE        IN VARCHAR2 DEFAULT NULL,
     MY_CURSOR       IN OUT T_CURSOR
    )
    IS
    BEGIN
       OPEN MY_CURSOR FOR
       SELECT * FROM MARCAS WHERE NOMBRE LIKE '%' || P_NOMBRE || '%' AND
ACTIVO = 1
       ORDER BY NOMBRE;
    END SP_CONSULTARPORNOMBRE;

    -- SP_CONSULTARTODOS_FORMS
```

```
-- Consulta todas las las marcas de los vehículos que se hayan registrado al sistema.

  PROCEDURE SP_CONSULTARTODOS_FORMS
 (
  MARCA_DATA  IN OUT T_MARCA,
  P_ACTIVO    IN NUMBER     DEFAULT NULL
 )
 IS
    CURSOR CUR_MARCAS IS
    SELECT * FROM MARCAS WHERE ACTIVO = P_ACTIVO
    ORDER BY NOMBRE;

 BEGIN
    OPEN CUR_MARCAS;
    CONTADOR := 1;
    LOOP
    FETCH CUR_MARCAS INTO
        MARCA_DATA(CONTADOR).CODMARCA,
        MARCA_DATA(CONTADOR).NOMBRE,
        MARCA_DATA(CONTADOR).ACTIVO;
        EXIT WHEN CUR_MARCAS%NOTFOUND;
        CONTADOR := CONTADOR + 1;
    END LOOP;
    IF CUR_MARCAS%ISOPEN THEN
      CLOSE CUR_MARCAS;
    END IF;
 EXCEPTION
    WHEN OTHERS THEN
    IF CUR_MARCAS%ISOPEN THEN
      CLOSE CUR_MARCAS;
    END IF;
 END SP_CONSULTARTODOS_FORMS;
END PK_MARCAS;
/
```

- **PK_SEGUROS.**

Objetivo: Administrar compañias de seguros de vehículos

Contiene los Procedimientos Almacenados para la Creación, Modificación, Eliminación y Consulta de la Tabla SEGUROS, que almacena los datos de las compañías aseguradoras de Vehículos.

```
prompt ================================

CREATE OR REPLACE PACKAGE BODY USER15.PK_SEGUROS IS
    NEW_ID          NUMBER;
    MSG             VARCHAR2(100);
    CONTADOR        NUMBER;
    EXP_ERROR       EXCEPTION;
```

```
--PROCEDURE SP_INSERTAR
-- Ingresa todos los datos de las compañías aseguradoradoras de vehículos

PROCEDURE SP_INSERTAR
(
 P_NOMBRE          IN VARCHAR2 DEFAULT NULL,
 P_ID              OUT NUMBER
)
IS
BEGIN
    MSG:= 'NO SE PUDO INSERTAR CIA. DE SEGUROS';
    SELECT COUNT(*) INTO CONTADOR FROM SEGUROS
    WHERE NOMBRE LIKE P_NOMBRE AND ACTIVO = 1;
    IF CONTADOR > 0 THEN
      MSG := 'EL NOMBRE DE ESTA ASEGURADORA YA EXISTE';
      RAISE EXP_ERROR;
    END IF;
    SELECT NVL(MAX(CODSEGURO) + 1, 1) INTO NEW_ID FROM SEGUROS;
    INSERT INTO SEGUROS
    (CODSEGURO,nombre,activo)
    VALUES
    (NEW_ID,P_NOMBRE,1);
    SELECT NEW_ID INTO P_ID FROM DUAL;
    COMMIT;
    EXCEPTION
    WHEN EXP_ERROR THEN
        RAISE_APPLICATION_ERROR(-20000, MSG);
        ROLLBACK;
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO');
        ROLLBACK;
  END SP_INSERTAR;

 -- PROCEDURE SP_GUARDAR
 -- Guarda el registro e ingreso de una compañía de seguros.

 PROCEDURE SP_GUARDAR
 (
  P_ID              IN NUMBER DEFAULT NULL,
  P_NOMBRE          IN VARCHAR2 DEFAULT NULL,
  P_ACTIVO          IN NUMBER DEFAULT NULL
 )
 IS
 BEGIN
    MSG:= 'NO SE PUDO GUARDAR CIA. DE SEGUROS';
    SELECT COUNT(*) INTO CONTADOR FROM SEGUROS
    WHERE CODSEGURO = P_ID;
    IF CONTADOR = 0 THEN
      MSG:= 'NO SE PUDO ENCONTRAR LA ASEGURADORA A MODIFICAR';
      RAISE EXP_ERROR;
    END IF;
    SELECT COUNT(*) INTO CONTADOR FROM SEGUROS
    WHERE NOMBRE LIKE P_NOMBRE AND ACTIVO = 1 AND CODSEGURO <>
P_ID;
    IF CONTADOR > 0 THEN
      MSG:= 'EL NOMBRE YA ESTA ASIGNADO A OTRA ASEGURADORA';
      RAISE EXP_ERROR;
```

```
        END IF;
        UPDATE SEGUROS SET
        NOMBRE = P_NOMBRE,
        ACTIVO = P_ACTIVO
        WHERE CODSEGURO = P_ID;
        COMMIT;
        EXCEPTION
        WHEN EXP_ERROR THEN
            RAISE_APPLICATION_ERROR(-20000, MSG);
            ROLLBACK;
        WHEN OTHERS THEN
            RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO');
            ROLLBACK;
    END SP_GUARDAR;

    -- PROCEDURE SP_ELIMINAR
    /*Valido para eliminar una Cía. Aseguradora y restaurar dependiendo del parámetro
      que recibe */

    PROCEDURE SP_ELIMINAR
    (
     P_ID            IN NUMBER DEFAULT NULL,
     P_ACTIVO        IN NUMBER DEFAULT NULL
    )
    IS
    BEGIN
        MSG:= 'NO SE PUDO ELIMINAR CIA. DE SEGUROS';
        SELECT COUNT(*) INTO CONTADOR FROM SEGUROS
        WHERE CODSEGURO = P_ID;
        IF CONTADOR = 0 THEN
          MSG:= 'NO SE PUDO ENCONTRAR LA ASEGURADORA';
          RAISE EXP_ERROR;
        END IF;
        UPDATE SEGUROS SET
        ACTIVO = P_ACTIVO
        WHERE CODSEGURO = P_ID;
        COMMIT;
        EXCEPTION
        WHEN EXP_ERROR THEN
            RAISE_APPLICATION_ERROR(-20000, MSG);
            ROLLBACK;
        WHEN OTHERS THEN
            RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO');
            ROLLBACK;
    END SP_ELIMINAR;

    -- PROCEDURE SP_CONSULTAR
    -- Consulta según el código de una Cía. Aseguradora.

    PROCEDURE SP_CONSULTAR
    (
     P_ID            IN NUMBER DEFAULT NULL,
     MY_CURSOR       IN OUT T_CURSOR
    )
    IS
    BEGIN
        OPEN MY_CURSOR FOR
```

```
        SELECT * FROM SEGUROS WHERE CODSEGURO = P_ID;
    END SP_CONSULTAR;



    -- PROCEDURE SP_CONSULTARPORNOMBRE
    -- Consulta por el nombre de una compañía aseguradora.

    PROCEDURE SP_CONSULTARPORNOMBRE
    (
     P_NOMBRE          IN VARCHAR2 DEFAULT NULL,
     MY_CURSOR         IN OUT T_CURSOR
    )
    IS
    BEGIN
        OPEN MY_CURSOR FOR
        SELECT * FROM SEGUROS WHERE NOMBRE LIKE '%' || P_NOMBRE || '%'
AND ACTIVO = 1
        ORDER BY NOMBRE;
    END SP_CONSULTARPORNOMBRE;

   -- PROCEDURE SP_CONSULTARTODOS_FORMS
   -- Consulta todas las Cías. Aseguradoras de vehículos


    PROCEDURE SP_CONSULTARTODOS_FORMS
    (
        DATA_SEGUROS IN OUT      T_SEGUROS,
        P_ACTIVO  IN NUMBER      DEFAULT NULL
    )
    IS
    CURSOR SEG_CURSOR IS
    SELECT * FROM SEGUROS WHERE
    ACTIVO = P_ACTIVO ORDER BY NOMBRE;
    BEGIN
        CONTADOR := 0;
        FOR FILA IN SEG_CURSOR
        LOOP
           CONTADOR := CONTADOR + 1;
           DATA_SEGUROS(CONTADOR) := FILA;
        END LOOP;
        IF SEG_CURSOR%ISOPEN THEN
          CLOSE SEG_CURSOR;
        END IF;
    END SP_CONSULTARTODOS_FORMS;
END PK_SEGUROS;
/
```

PK_SOLICITUD.

Objetivo: Maneja todos los datos de las solicitudes de crédito.
Contiene los Procedimientos Almacenados para la Creación, Modificación, Eliminación y Consulta de las Solicitudes de Crédito.

```
prompt Creating package body PK_SOLICITUD
prompt ================================

CREATE OR REPLACE PACKAGE BODY USER15.PK_SOLICITUD IS
```

```
MSG             VARCHAR2(100);
EXP_ERROR       EXCEPTION;

--PROCEDURE SP_SETACTIVOS
-- Permite grabar los datos de la tabla Activos.

 PROCEDURE SP_SETACTIVOS
(
  P_ID             IN NUMBER   DEFAULT NULL,
  v_casadir1       IN VARCHAR2 DEFAULT NULL,
  v_casadir2       IN VARCHAR2 DEFAULT NULL,
  v_casavalor1     IN NUMBER   DEFAULT NULL,
  v_casavalor2     IN NUMBER   DEFAULT NULL,
  v_depdir1        IN VARCHAR2 DEFAULT NULL,
  v_depdir2        IN VARCHAR2 DEFAULT NULL,
  v_depvalor1      IN NUMBER   DEFAULT NULL,
  v_depvalor2      IN NUMBER   DEFAULT NULL,
  v_vehmarca1      IN VARCHAR2 DEFAULT NULL,
  v_vehmarca2      IN VARCHAR2 DEFAULT NULL,
  v_vehmodelo1     IN VARCHAR2 DEFAULT NULL,
  v_vehmodelo2     IN VARCHAR2 DEFAULT NULL,
  v_vehanio1       IN VARCHAR2 DEFAULT NULL,
  v_vehanio2       IN VARCHAR2 DEFAULT NULL,
  v_vehvalor1      IN NUMBER   DEFAULT NULL,
  v_vehvalor2      IN NUMBER   DEFAULT NULL,
  v_inversion1     IN VARCHAR2 DEFAULT NULL,
  v_inversion2     IN VARCHAR2 DEFAULT NULL,
  v_plazo1         IN VARCHAR2 DEFAULT NULL,
  v_plazo2         IN VARCHAR2 DEFAULT NULL,
  v_monto1         IN NUMBER   DEFAULT NULL,
  v_monto2         IN NUMBER   DEFAULT NULL,
  v_mercaderia     IN NUMBER   DEFAULT NULL,
  v_muebles        IN NUMBER   DEFAULT NULL,
  v_acciones       IN NUMBER   DEFAULT NULL,
  v_maquinarias    IN NUMBER   DEFAULT NULL
)
IS
BEGIN
    IF FN_CLIENTE_EXISTE(P_ID => P_ID) = FALSE THEN
    --INSERT
      MSG:= 'UN ERROR HA OCURRIDO AL INSERTAR ACTIVOS';
      INSERT INTO activos
        (codcliente, casadir1, casadir2, casavalor1, casavalor2, depdir1, depdir2, depvalor1,
depvalor2, vehmarca1, vehmarca2, vehmodelo1, vehmodelo2, vehanio1, vehanio2, vehvalor1,
vehvalor2, inversion1, inversion2, plazo1, plazo2, monto1, monto2, mercaderia, muebles,
acciones, maquinarias)
        VALUES
        (P_ID, v_casadir1, v_casadir2, v_casavalor1, v_casavalor2, v_depdir1, v_depdir2,
v_depvalor1, v_depvalor2, v_vehmarca1, v_vehmarca2, v_vehmodelo1, v_vehmodelo2,
v_vehanio1, v_vehanio2, v_vehvalor1, v_vehvalor2, v_inversion1, v_inversion2, v_plazo1,
v_plazo2, v_monto1, v_monto2, v_mercaderia, v_muebles, v_acciones, v_maquinarias);
    ELSE

    --Update
      MSG:= 'UN ERROR HA OCURRIDO AL EDITAR ACTIVOS';
      UPDATE activos
        SET casadir1 = v_casadir1,
          casadir2 = v_casadir2,
          casavalor1 = v_casavalor1,
```

```
                casavalor2 = v_casavalor2,
                depdir1 = v_depdir1,
                depdir2 = v_depdir2,
                depvalor1 = v_depvalor1,
                depvalor2 = v_depvalor2,
                vehmarca1 = v_vehmarca1,
                vehmarca2 = v_vehmarca2,
                vehmodelo1 = v_vehmodelo1,
                vehmodelo2 = v_vehmodelo2,
                vehanio1 = v_vehanio1,
                vehanio2 = v_vehanio2,
                vehvalor1 = v_vehvalor1,
                vehvalor2 = v_vehvalor2,
                inversion1 = v_inversion1,
                inversion2 = v_inversion2,
                plazo1 = v_plazo1,
                plazo2 = v_plazo2,
                monto1 = v_monto1,
                monto2 = v_monto2,
                mercaderia = v_mercaderia,
                muebles = v_muebles,
                acciones = v_acciones,
                maquinarias = v_maquinarias
            WHERE CODCLIENTE = P_ID;
        END IF;
        COMMIT;
        EXCEPTION
        WHEN EXP_ERROR THEN
            RAISE_APPLICATION_ERROR(-20000, MSG);
            ROLLBACK;
        WHEN OTHERS THEN
            RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO AL
GRABAR SOLICITUD DATOS ACTIVOS');
            ROLLBACK;
    END SP_SETACTIVOS;
```

--PROCEDURE SP_GETACTIVOS_FORMS

-- Obtiene o devuelve los datos de la tabla Activos

```
PROCEDURE SP_GETACTIVOS_FORMS
    (
     P_ID             IN NUMBER   DEFAULT NULL,
     v_casadir1        OUT VARCHAR2 ,
     v_casadir2        OUT VARCHAR2 ,
     v_casavalor1       OUT NUMBER   ,
     v_casavalor2       OUT NUMBER   ,
     v_depdir1         OUT VARCHAR2 ,
     v_depdir2         OUT VARCHAR2 ,
     v_depvalor1        OUT NUMBER   ,
     v_depvalor2        OUT NUMBER   ,
     v_vehmarca1         OUT VARCHAR2 ,
     v_vehmarca2         OUT VARCHAR2 ,
     v_vehmodelo1        OUT VARCHAR2 ,
     v_vehmodelo2        OUT VARCHAR2 ,
     v_vehanio1         OUT VARCHAR2 ,
     v_vehanio2         OUT VARCHAR2 ,
     v_vehvalor1        OUT NUMBER   ,
     v_vehvalor2        OUT NUMBER   ,
     v_inversion1       OUT VARCHAR2 ,
```

```
  v_inversion2       OUT VARCHAR2 ,
  v_plazo1           OUT VARCHAR2 ,
  v_plazo2           OUT VARCHAR2 ,
  v_monto1           OUT NUMBER   ,
  v_monto2           OUT NUMBER   ,
  v_mercaderia       OUT NUMBER   ,
  v_muebles          OUT NUMBER   ,
  v_acciones         OUT NUMBER   ,
  v_maquinarias      OUT NUMBER
)
IS
CURSOR C_ACTIVOS IS
SELECT *
FROM   ACTIVOS
WHERE  CODCLIENTE = P_ID;
BEGIN
     CONTADOR := 0;
     FOR FILA IN C_ACTIVOS LOOP
        CONTADOR := CONTADOR + 1;
        v_casadir1 := FILA.casadir1;
        v_casadir2 := FILA.casadir2;
        v_casavalor1 := FILA.casavalor1;
        v_casavalor2 :=FILA.casavalor2;
        v_depdir1 := FILA.depdir1;
        v_depdir2 := FILA.depdir2;
        v_depvalor1 := FILA.depvalor1;
        v_depvalor2 := FILA.depvalor2;
        v_vehmarca1 := FILA.vehmarca1;
        v_vehmarca2 := FILA.vehmarca2;
        v_vehmodelo1 := FILA.vehmodelo1;
        v_vehmodelo2 := FILA.vehmodelo2;
        v_vehanio1 := FILA.vehanio1;
        v_vehanio2 := FILA.vehanio2;
        v_vehvalor1 := FILA.vehvalor1;
        v_vehvalor2 := FILA.vehvalor2;
        v_inversion1 := FILA.inversion1;
        v_inversion2 := FILA.inversion2;
        v_plazo1 := FILA.plazo1;
        v_plazo2 := FILA.plazo2;
        v_monto1 := FILA.monto1;
        v_monto2 := FILA.monto2;
        v_mercaderia := FILA.mercaderia;
        v_muebles := FILA.muebles;
        v_acciones := FILA.acciones;
        v_maquinarias := FILA.maquinarias;
     END LOOP;
     IF C_ACTIVOS%ISOPEN THEN
       CLOSE C_ACTIVOS;
     END IF;
END;

-- PROCEDURE SP_SETCONYUGE
-- Permite grabar los datos de la tabla Conyuge

PROCEDURE SP_SETCONYUGE
(
  P_ID              IN NUMBER   DEFAULT NULL,
  v_cedula          IN VARCHAR2 DEFAULT NULL,
  v_apellidos       IN VARCHAR2 DEFAULT NULL,
```

```
  v_nombres          IN VARCHAR2 DEFAULT NULL,
  v_nacionalidad     IN VARCHAR2 DEFAULT NULL,
  v_fechanac         IN DATE     DEFAULT NULL,
  v_tipolaboral      IN VARCHAR2 DEFAULT NULL,
  v_sepbienes        IN NUMBER   DEFAULT NULL,
  v_celular          IN VARCHAR2 DEFAULT NULL,
  v_profesion        IN VARCHAR2 DEFAULT NULL,
  v_estudios         IN VARCHAR2 DEFAULT NULL
)
IS
BEGIN
   IF FN_CLIENTE_EXISTE(P_ID => P_ID) = FALSE THEN

   --Insert
      MSG:= 'UN ERROR HA OCURRIDO AL INSERTAR CONYUGE';
      INSERT INTO conyuges
        (codcliente, cedula, apellidos, nombres, nacionalidad, fechanac, tipolaboral,
sepbienes, celular, profesion, estudios)
      VALUES
        (P_ID, v_cedula, v_apellidos, v_nombres, v_nacionalidad, v_fechanac,
v_tipolaboral, v_sepbienes, v_celular, v_profesion, v_estudios);
   ELSE

   --Update
      MSG:= 'UN ERROR HA OCURRIDO AL EDITAR CONYUGE';
      UPDATE CONYUGES
      SET    cedula = v_cedula,
             apellidos = v_apellidos,
             nombres = v_nombres,
             nacionalidad = v_nacionalidad,
             fechanac = v_fechanac,
             tipolaboral = v_tipolaboral,
             sepbienes = v_sepbienes,
             celular = v_celular,
             profesion = v_profesion,
             estudios = v_estudios
      WHERE  CODCLIENTE = P_ID;
   END IF;
   COMMIT;
   EXCEPTION
   WHEN EXP_ERROR THEN
       RAISE_APPLICATION_ERROR(-20000, MSG);
       ROLLBACK;
   WHEN OTHERS THEN
       RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO AL
GRABAR SOLICITUD DATOS CONYUGE');
       ROLLBACK;
END SP_SETCONYUGE;

-- PROCEDURE SP_GETCONYUGE_FORMS
-- Obtiene los datos del conyuge del cliente

PROCEDURE SP_GETCONYUGE_FORMS
(
  P_ID             IN NUMBER   DEFAULT NULL,
  v_cedula          OUT VARCHAR2,
  v_apellidos       OUT VARCHAR2,
  v_nombres         OUT VARCHAR2,
  v_nacionalidad    OUT VARCHAR2,
```

```
  v_fechanac         OUT DATE,
  v_tipolaboral      OUT VARCHAR2,
  v_sepbienes        OUT NUMBER,
  v_celular          OUT VARCHAR2,
  v_profesion        OUT VARCHAR2,
  v_estudios         OUT VARCHAR2
)
IS
CURSOR C_CONYUGE IS
SELECT * FROM CONYUGES
WHERE CODCLIENTE = P_ID;
   BEGIN
    FOR FILA IN C_CONYUGE LOOP
       v_cedula := FILA.CEDULA;
       v_apellidos := FILA.APELLIDOS;
       v_nombres := FILA.NOMBRES;
       v_nacionalidad := FILA.NACIONALIDAD;
       v_fechanac := FILA.FECHANAC;
       v_tipolaboral := FILA.TIPOLABORAL;
       v_sepbienes := FILA.SEPBIENES;
       v_celular := FILA.CELULAR;
       v_profesion := FILA.PROFESION;
       v_estudios := FILA.ESTUDIOS;
    END LOOP;
    IF C_CONYUGE%ISOPEN THEN
      CLOSE C_CONYUGE;
    END IF;
END;
```

--PROCEDURE SP_SETGASTOS

- Devuelve los Gastos del Cliente

```
PROCEDURE SP_SETGASTOS
(
 P_ID              IN NUMBER   DEFAULT NULL,
 v_servicios       IN NUMBER   DEFAULT NULL,
 v_alimentacion    IN NUMBER   DEFAULT NULL,
 v_educacion       IN NUMBER   DEFAULT NULL,
 v_prestamos       IN NUMBER   DEFAULT NULL,
 v_tarjetas        IN NUMBER   DEFAULT NULL,
 v_cuentas         IN NUMBER   DEFAULT NULL,
 v_otros           IN NUMBER   DEFAULT NULL
)
IS
BEGIN
   IF FN_CLIENTE_EXISTE(P_ID => P_ID) = FALSE THEN

   --Insert
      MSG:= 'UN ERROR HA OCURRIDO AL INSERTAR GASTOS';
      INSERT INTO gastos
        (codcliente, servicios, alimentacion, educacion, prestamos, tarjetas, cuentas,
         otros)
      VALUES
        (P_ID, v_servicios, v_alimentacion, v_educacion, v_prestamos, v_tarjetas,
         v_cuentas, v_otros);
   ELSE

   --Update
      MSG:= 'UN ERROR HA OCURRIDO AL EDITAR GASTOS';
```

```
        UPDATE GASTOS
        SET    servicios = v_servicios,
               alimentacion = v_alimentacion,
               educacion = v_educacion,
               prestamos = v_prestamos,
               tarjetas = v_tarjetas,
               cuentas = v_cuentas,
               otros = v_otros
        WHERE  CODCLIENTE = P_ID;
    END IF;
    COMMIT;
    EXCEPTION
    WHEN EXP_ERROR THEN
        RAISE_APPLICATION_ERROR(-20000, MSG);
        ROLLBACK;
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO AL
        GRABAR SOLICITUD DATOS GASTOS');
        ROLLBACK;
END SP_SETGASTOS;
```

```
-- PROCEDURE SP_GETGASTOS_FORMS
-- Obtiene los datos de los Gastos del Cliente

PROCEDURE SP_GETGASTOS_FORMS
(
  P_ID              IN NUMBER   DEFAULT NULL,
  v_servicios       OUT NUMBER,
  v_alimentacion    OUT NUMBER,
  v_educacion       OUT NUMBER,
  v_prestamos       OUT NUMBER,
  v_tarjetas        OUT NUMBER,
  v_cuentas         OUT NUMBER,
  v_otros           OUT NUMBER,
  v_total           OUT NUMBER
)
IS
  CURSOR C_GASTOS IS
  SELECT * FROM GASTOS
  WHERE CODCLIENTE = P_ID;
BEGIN
  FOR FILA IN C_GASTOS LOOP
     v_servicios := FILA.SERVICIOS;
     v_alimentacion := FILA.ALIMENTACION;
     v_educacion := FILA.EDUCACION;
     v_prestamos := FILA.PRESTAMOS;
     v_tarjetas  := FILA.TARJETAS;
     v_cuentas := FILA.CUENTAS;
     v_otros := FILA.OTROS;
     v_total := nvl(v_servicios,0) +
              nvl(v_alimentacion,0) +
              nvl(v_educacion,0) +
              nvl(v_prestamos,0) +
              nvl(v_tarjetas,0) +
```

```
                nvl(v_cuentas,0) +
                nvl(v_otros,0);
      END LOOP;
      IF C_GASTOS%ISOPEN THEN
        CLOSE C_GASTOS;
      END IF;
    END;

 -- PROCEDURE SP_SETPARIENTES
 --Devuelve los datos de los Parientes del Cliente
    PROCEDURE SP_SETPARIENTES
    (
      P_ID             IN NUMBER   DEFAULT NULL,
      v_nombre1          IN VARCHAR2 DEFAULT NULL,
      v_nombre2          IN VARCHAR2 DEFAULT NULL,
      v_telefono1        IN VARCHAR2 DEFAULT NULL,
      v_telefono2        IN VARCHAR2 DEFAULT NULL,
      v_relacion1        IN VARCHAR2 DEFAULT NULL,
      v_relacion2        IN VARCHAR2 DEFAULT NULL,
      v_direccion1       IN VARCHAR2 DEFAULT NULL,
      v_direccion2       IN VARCHAR2 DEFAULT NULL,
      v_ciudad1          IN VARCHAR2 DEFAULT NULL,
      v_ciudad2          IN VARCHAR2 DEFAULT NULL
    )
    IS
    BEGIN
      IF FN_CLIENTE_EXISTE(P_ID => P_ID) = FALSE THEN

      --Insert
        MSG:= 'UN ERROR HA OCURRIDO AL INSERTAR PARIENTES';
        INSERT INTO parientes
          (codcliente, nombre1, nombre2, telefono1, telefono2, relacion1, relacion2,
direccion1, direccion2, ciudad1, ciudad2)
        VALUES
          (P_ID, v_nombre1, v_nombre2, v_telefono1, v_telefono2, v_relacion1,
v_relacion2, v_direccion1, v_direccion2, v_ciudad1, v_ciudad2);
      ELSE

      --Update
        MSG:= 'UN ERROR HA OCURRIDO AL EDITAR PARIENTES';
        UPDATE PARIENTES
        SET    nombre1 = v_nombre1,
               nombre2 = v_nombre2,
               telefono1 = v_telefono1,
               telefono2 = v_telefono2,
               relacion1 = v_relacion1,
               relacion2 = v_relacion2,
               direccion1 = v_direccion1,
               direccion2 = v_direccion2,
               ciudad1 = v_ciudad1,
               ciudad2 = v_ciudad2
        WHERE  CODCLIENTE = P_ID;
      END IF;
      COMMIT;
      EXCEPTION
      WHEN EXP_ERROR THEN
         RAISE_APPLICATION_ERROR(-20000, MSG);
         ROLLBACK;
      WHEN OTHERS THEN
```

```
            RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO AL
GRABAR SOLICITUD DATOS PARIENTES');
            ROLLBACK;
     END SP_SETPARIENTES;
```

-- PROCEDURE SP_GETPARIENTES_FORMS

-- Obtiene los datos necesarios de la tabla parientes para la solicitud de crédito

```
     PROCEDURE SP_GETPARIENTES_FORMS
     (
       P_ID              IN NUMBER   DEFAULT NULL,
       v_nombre1          OUT VARCHAR2,
       v_nombre2          OUT VARCHAR2,
       v_telefono1        OUT VARCHAR2,
       v_telefono2        OUT VARCHAR2,
       v_relacion1        OUT VARCHAR2,
       v_relacion2        OUT VARCHAR2,
       v_direccion1       OUT VARCHAR2,
       v_direccion2       OUT VARCHAR2,
       v_ciudad1          OUT VARCHAR2,
       v_ciudad2          OUT VARCHAR2
     )
     IS
     CURSOR C_PARIENTES IS
     SELECT * FROM PARIENTES
     WHERE CODCLIENTE = P_ID;
     BEGIN
        FOR FILA IN C_PARIENTES LOOP
           v_nombre1 := FILA.NOMBRE1;
           v_nombre2 := FILA.NOMBRE2;
           v_telefono1 := FILA.TELEFONO1;
           v_telefono2 := FILA.TELEFONO2;
           v_relacion1 := FILA.RELACION1;
           v_relacion2 := FILA.RELACION2;
           v_direccion1 := FILA.DIRECCION1;
           v_direccion2 := FILA.DIRECCION2;
           v_ciudad1 := FILA.CIUDAD1;
           v_ciudad2 := FILA.CIUDAD2;
        END LOOP;
        IF C_PARIENTES%ISOPEN THEN
          CLOSE C_PARIENTES;
        END IF;
     END SP_GETPARIENTES_FORMS;
```

--PROCEDURE SP_SETPASIVOS

-- Devuelve los datos de los Pasivos de los clientes

```
     PROCEDURE SP_SETPASIVOS
     (
       P_ID              IN NUMBER   DEFAULT NULL,
       v_banco1            IN VARCHAR2 DEFAULT NULL,
       v_banco2            IN VARCHAR2 DEFAULT NULL,
       v_banco3            IN VARCHAR2 DEFAULT NULL,
       v_banco4            IN VARCHAR2 DEFAULT NULL,
       v_plazo1           IN VARCHAR2 DEFAULT NULL,
       v_plazo2           IN VARCHAR2 DEFAULT NULL,
       v_plazo3           IN VARCHAR2 DEFAULT NULL,
       v_plazo4           IN VARCHAR2 DEFAULT NULL,
       v_pago1            IN NUMBER   DEFAULT NULL,
```

```
    v_pago2          IN NUMBER   DEFAULT NULL,
    v_pago3          IN NUMBER   DEFAULT NULL,
    v_pago4          IN NUMBER   DEFAULT NULL,
    v_saldoa1        IN NUMBER   DEFAULT NULL,
    v_saldoa2        IN NUMBER   DEFAULT NULL,
    v_saldoa3        IN NUMBER   DEFAULT NULL,
    v_saldoa4        IN NUMBER   DEFAULT NULL,
    v_saldoi1        IN NUMBER   DEFAULT NULL,
    v_saldoi2        IN NUMBER   DEFAULT NULL,
    v_saldoi3        IN NUMBER   DEFAULT NULL,
    v_saldoi4        IN NUMBER   DEFAULT NULL
  )
  IS
  BEGIN
     IF FN_CLIENTE_EXISTE(P_ID => P_ID) = FALSE THEN

     --Insert
        MSG:= 'UN ERROR HA OCURRIDO AL INSERTAR PASIVOS';
        INSERT INTO pasivos
          (codcliente, banco1, banco2, banco3, banco4, plazo1, plazo2, plazo3, plazo4,
pago1, pago2, pago3, pago4, saldoa1, saldoa2, saldoa3, saldoa4, saldoi1, saldoi2, saldoi3,
saldoi4)
        VALUES
          (P_ID, v_banco1, v_banco2, v_banco3, v_banco4, v_plazo1, v_plazo2, v_plazo3,
v_plazo4, v_pago1, v_pago2, v_pago3, v_pago4, v_saldoa1, v_saldoa2, v_saldoa3, v_saldoa4,
v_saldoi1, v_saldoi2, v_saldoi3, v_saldoi4);
     ELSE

     --Update
        MSG:= 'UN ERROR HA OCURRIDO AL EDITAR PASIVOS';
        UPDATE PASIVOS
        SET    banco1 = v_banco1,
               banco2 = v_banco2,
               banco3 = v_banco3,
               banco4 = v_banco4,
               plazo1 = v_plazo1,
               plazo2 = v_plazo2,
               plazo3 = v_plazo3,
               plazo4 = v_plazo4,
               pago1 = v_pago1,
               pago2 = v_pago2,
               pago3 = v_pago3,
               pago4 = v_pago4,
               saldoa1 = v_saldoa1,
               saldoa2 = v_saldoa2,
               saldoa3 = v_saldoa3,
               saldoa4 = v_saldoa4,
               saldoi1 = v_saldoi1,
               saldoi2 = v_saldoi2,
               saldoi3 = v_saldoi3,
               saldoi4 = v_saldoi4
          WHERE  CODCLIENTE = P_ID;
     END IF;
     COMMIT;
     EXCEPTION
     WHEN EXP_ERROR THEN
         RAISE_APPLICATION_ERROR(-20000, MSG);
         ROLLBACK;
     WHEN OTHERS THEN
```

```
        RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO AL
GRABAR SOLICITUD DATOS PASIVOS');
        ROLLBACK;
    END SP_SETPASIVOS;

    -- PROCEDURE SP_GETPASIVOS_FORMS
    -- Obtiene los datos de los pasivos de los clientes

    PROCEDURE SP_GETPASIVOS_FORMS
    (
     P_ID           IN NUMBER   DEFAULT NULL,
     v_banco1         OUT VARCHAR2,
     v_banco2         OUT VARCHAR2,
     v_banco3         OUT VARCHAR2,
     v_banco4         OUT VARCHAR2,
     v_plazo1        OUT VARCHAR2,
     v_plazo2        OUT VARCHAR2,
     v_plazo3        OUT VARCHAR2,
     v_plazo4        OUT VARCHAR2,
     v_pago1          OUT NUMBER,
     v_pago2          OUT NUMBER,
     v_pago3          OUT NUMBER,
     v_pago4          OUT NUMBER,
     v_saldoa1        OUT NUMBER,
     v_saldoa2        OUT NUMBER,
     v_saldoa3        OUT NUMBER,
     v_saldoa4        OUT NUMBER,
     v_saldoi1       OUT NUMBER,
     v_saldoi2       OUT NUMBER,
     v_saldoi3       OUT NUMBER,
     v_saldoi4        OUT NUMBER
    )
    IS
    CURSOR C_PASIVOS IS
    SELECT * FROM PASIVOS
    WHERE CODCLIENTE = P_ID;
    BEGIN
       FOR FILA IN C_PASIVOS LOOP
          v_banco1 := FILA.BANCO1;
          v_banco2 := FILA.BANCO2;
          v_banco3 := FILA.BANCO3;
          v_banco4 := FILA.BANCO4;
          v_plazo1 := FILA.PLAZO1;
          v_plazo2 := FILA.PLAZO2;
          v_plazo3 := FILA.PLAZO3;
          v_plazo4 := FILA.PLAZO4;
          v_pago1 := FILA.PAGO1;
          v_pago2 := FILA.PAGO2;
          v_pago3 := FILA.PAGO3;
          v_pago4 := FILA.PAGO4;
          v_saldoa1 := FILA.SALDOA1;
          v_saldoa2 := FILA.SALDOA2;
          v_saldoa3 := FILA.SALDOA3;
          v_saldoa4 := FILA.SALDOA4;
          v_saldoi1 := FILA.SALDOI1;
          v_saldoi2 := FILA.SALDOI2;
          v_saldoi3 := FILA.SALDOI3;
          v_saldoi4 := FILA.SALDOI4;
       END LOOP;
```

```
        IF C_PASIVOS%ISOPEN THEN
          CLOSE C_PASIVOS;
        END IF;
    END SP_GETPASIVOS_FORMS;

   -- PROCEDURE SP_SETREFBANCARIAS
   -- Devuelve los datos de las referencias bancarias del cliente

   PROCEDURE SP_SETREFBANCARIAS
    (
     P_ID             IN NUMBER   DEFAULT NULL,
     v_numero1          IN VARCHAR2 DEFAULT NULL,
     v_numero2          IN VARCHAR2 DEFAULT NULL,
     v_numero3          IN VARCHAR2 DEFAULT NULL,
     v_tipo1          IN VARCHAR2 DEFAULT NULL,
     v_tipo2          IN VARCHAR2 DEFAULT NULL,
     v_tipo3          IN VARCHAR2 DEFAULT NULL,
     v_banco1           IN VARCHAR2 DEFAULT NULL,
     v_banco2           IN VARCHAR2 DEFAULT NULL,
     v_banco3           IN VARCHAR2 DEFAULT NULL
    )
    IS
    BEGIN
        IF FN_CLIENTE_EXISTE(P_ID => P_ID) = FALSE THEN

        --Insert
           MSG:= 'UN ERROR HA OCURRIDO AL INSERTAR REFERENCIAS
BANCARIAS';
           INSERT INTO refbancarias
             (codcliente, numero1, numero2, numero3, tipo1, tipo2, tipo3, banco1, banco2,
banco3)
           VALUES
             (P_ID, v_numero1, v_numero2, v_numero3, v_tipo1, v_tipo2, v_tipo3, v_banco1,
v_banco2, v_banco3);
        ELSE
        --UPDATE
           MSG:= 'UN ERROR HA OCURRIDO AL EDITAR REFERENCIAS
BANCARIAS';
           UPDATE REFBANCARIAS
           SET    numero1 = v_numero1,
                  numero2 = v_numero2,
                  numero3 = v_numero3,
                  tipo1 = v_tipo1,
                  tipo2 = v_tipo2,
                  tipo3 = v_tipo3,
                  banco1 = v_banco1,
                  banco2 = v_banco2,
                  banco3 = v_banco3
           WHERE  CODCLIENTE = P_ID;
        END IF;
        COMMIT;
        EXCEPTION
        WHEN EXP_ERROR THEN
           RAISE_APPLICATION_ERROR(-20000, MSG);
           ROLLBACK;
        WHEN OTHERS THEN
           RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO AL
GRABAR SOLICITUD DATOS BANCOS');
           ROLLBACK;
```

```
END SP_SETREFBANCARIAS;

--PROCEDURE SP_GETREFBANCARIAS_FORMS
-- Obtiene los datos de las referencias bancarias del cliente

PROCEDURE SP_GETREFBANCARIAS_FORMS
(
   P_ID             IN NUMBER   DEFAULT NULL,
   v_numero1           OUT VARCHAR2,
   v_numero2           OUT VARCHAR2,
   v_numero3           OUT VARCHAR2,
   v_tipo1           OUT VARCHAR2,
   v_tipo2           OUT VARCHAR2,
   v_tipo3           OUT VARCHAR2,
   v_banco1            OUT VARCHAR2,
   v_banco2            OUT VARCHAR2,
   v_banco3            OUT VARCHAR2
)
IS
CURSOR C_REFBCOS IS
SELECT * FROM REFBANCARIAS
WHERE CODCLIENTE = P_ID;
BEGIN
    FOR FILA IN C_REFBCOS LOOP
      v_numero1 := FILA.NUMERO1;
      v_numero2 := FILA.NUMERO2;
      v_numero3 := FILA.NUMERO3;
      v_tipo1 := FILA.Tipo1;
      v_tipo2 := FILA.Tipo2;
      v_tipo3 := FILA.Tipo3;
      v_banco1 := FILA.BANCO1;
      v_banco2 := FILA.BANCO2;
      v_banco3 := FILA.BANCO3;
    END LOOP;
    IF C_REFBCOS%ISOPEN THEN
      CLOSE C_REFBCOS;
    END IF;
END SP_GETREFBANCARIAS_FORMS;

-- PROCEDURE SP_SETREFCOMERCIALES
-- Muestra los  datos de las referencias comerciales del cliente.

PROCEDURE SP_SETREFCOMERCIALES
(
  P_ID             IN NUMBER   DEFAULT NULL,
  v_nombre1          IN VARCHAR2 DEFAULT NULL,
  v_nombre2          IN VARCHAR2 DEFAULT NULL,
  v_telefono1        IN VARCHAR2 DEFAULT NULL,
  v_telefono2        IN VARCHAR2 DEFAULT NULL,
  v_anio1           IN NUMBER   DEFAULT NULL,
  v_anio2           IN NUMBER   DEFAULT NULL,
  v_monto1            IN NUMBER   DEFAULT NULL,
  v_monto2            IN NUMBER   DEFAULT NULL
)
IS
BEGIN
    IF FN_CLIENTE_EXISTE(P_ID => P_ID) = FALSE THEN
    --INSERT
```

```
          MSG:= 'UN ERROR HA OCURRIDO AL INSERTAR REFERENCIAS
COMERCIALES';
          INSERT INTO refcomerciales
            (codcliente, nombre1, nombre2, telefono1, telefono2, anio1, anio2, monto1,
monto2)
          VALUES
            (P_ID, v_nombre1, v_nombre2, v_telefono1, v_telefono2, v_anio1, v_anio2,
v_monto1, v_monto2);
        ELSE

        --Update
          MSG:= 'UN ERROR HA OCURRIDO AL EDITAR REFERENCIAS
COMERCIALES';
          UPDATE REFCOMERCIALES
          SET    nombre1 = v_nombre1,
                 nombre2 = v_nombre2,
                 telefono1 = v_telefono1,
                 telefono2 = v_telefono2,
                 anio1 = v_anio1,
                 anio2 = v_anio2,
                 monto1 = v_monto1,
                 monto2 = v_monto2
          WHERE  CODCLIENTE = P_ID;
        END IF;
        COMMIT;
        EXCEPTION
        WHEN EXP_ERROR THEN
           RAISE_APPLICATION_ERROR(-20000, MSG);
           ROLLBACK;
        WHEN OTHERS THEN
           RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO AL
GRABAR SOLICITUD DATOS REF. COMERCIALES');
           ROLLBACK;
    END SP_SETREFCOMERCIALES;

    -- PROCEDURE SP_GETREFCOMERCIALES_FORMS
    -- Obtiene los datos de las referencias comerciales de los clientes

    PROCEDURE SP_GETREFCOMERCIALES_FORMS
    (
      P_ID             IN NUMBER   DEFAULT NULL,
      v_nombre1          OUT VARCHAR2,
      v_nombre2          OUT VARCHAR2,
      v_telefono1        OUT VARCHAR2,
      v_telefono2        OUT VARCHAR2,
      v_anio1           OUT NUMBER,
      v_anio2           OUT NUMBER,
      v_monto1           OUT NUMBER,
      v_monto2           OUT NUMBER
    )
    IS
    CURSOR C_REFCOM IS
    SELECT * FROM   REFCOMERCIALES
    WHERE  CODCLIENTE = P_ID;
    BEGIN
       FOR FILA IN C_REFCOM LOOP
         v_nombre1 := FILA.NOMBRE1;
         v_nombre2 := FILA.NOMBRE2;
         v_telefono1 := FILA.TELEFONO1;
```

```
        v_telefono2 := FILA.TELEFONO2;
        v_anio1 := FILA.ANIO1;
        v_anio2 := FILA.ANIO2;
        v_monto1 := FILA.MONTO1;
        v_monto2 := FILA.MONTO2;
      END LOOP;
      IF C_REFCOM%ISOPEN THEN
        CLOSE C_REFCOM;
      END IF;
    END SP_GETREFCOMERCIALES_FORMS;

  --PROCEDURE SP_SETREFTARJETAS
  -- Devuelve los datos de las tarjetas de crédito del cliente

    PROCEDURE SP_SETREFTARJETAS
    (
     P_ID             IN NUMBER   DEFAULT NULL,
     v_numero1          IN VARCHAR2 DEFAULT NULL,
     v_numero2          IN VARCHAR2 DEFAULT NULL,
     v_numero3          IN VARCHAR2 DEFAULT NULL,
     v_tipo1          IN VARCHAR2 DEFAULT NULL,
     v_tipo2          IN VARCHAR2 DEFAULT NULL,
     v_tipo3          IN VARCHAR2 DEFAULT NULL,
     v_banco1           IN VARCHAR2 DEFAULT NULL,
     v_banco2           IN VARCHAR2 DEFAULT NULL,
     v_banco3           IN VARCHAR2 DEFAULT NULL
    )
    IS
    BEGIN
       IF FN_CLIENTE_EXISTE(P_ID => P_ID) = FALSE THEN

       --Insert
          MSG:= 'UN ERROR HA OCURRIDO AL INSERTAR TARJETAS DE CREDITO';
          INSERT INTO reftarjetas
            (codcliente, numero1, numero2, numero3, tipo1, tipo2, tipo3, banco1, banco2,
banco3)
          VALUES
            (P_ID, v_numero1, v_numero2, v_numero3, v_tipo1, v_tipo2, v_tipo3, v_banco1,
v_banco2, v_banco3);
       ELSE

       --Update
          MSG:= 'UN ERROR HA OCURRIDO AL EDITAR TARJETAS DE CREDITO';
          UPDATE reftarjetas
          SET    numero1 = v_numero1,
                 numero2 = v_numero2,
                 numero3 = v_numero3,
                 tipo1 = v_tipo1,
                 tipo2 = v_tipo2,
                 tipo3 = v_tipo3,
                 banco1 = v_banco1,
                 banco2 = v_banco2,
                 banco3 = v_banco3
          WHERE  CODCLIENTE = P_ID;
       END IF;
       COMMIT;
       EXCEPTION
       WHEN EXP_ERROR THEN
          RAISE_APPLICATION_ERROR(-20000, MSG);
```

```
            ROLLBACK;
        WHEN OTHERS THEN
            RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO AL
GRABAR SOLICITUD DATOS TARJETAS');
            ROLLBACK;
    END SP_SETREFTARJETAS;

    -- PROCEDURE SP_GETREFTARJETAS
    -- Obtiene los datos de las referencias de las Tarjetas de crédito del cliente.

    PROCEDURE SP_GETREFTARJETAS
    (
      P_ID              IN NUMBER   DEFAULT NULL,
      MY_CURSOR         IN OUT      T_CURSOR
    )
    IS
    BEGIN
        OPEN MY_CURSOR FOR
        SELECT numero1, numero2, numero3, tipo1, tipo2, tipo3, banco1, banco2, banco3
        FROM   REFTARJETAS
        WHERE  CODCLIENTE = P_ID;
    END SP_GETREFTARJETAS;

    -- PROCEDURE SP_GETREFTARJETAS_FORMS
    -- Devuelve los datos de las referencias de las tarjetas
    PROCEDURE SP_GETREFTARJETAS_FORMS
    (
      P_ID              IN NUMBER   DEFAULT NULL,
      v_numero1           OUT VARCHAR2,
      v_numero2           OUT VARCHAR2,
      v_numero3           OUT VARCHAR2,
      v_tipo1           OUT VARCHAR2,
      v_tipo2           OUT VARCHAR2,
      v_tipo3           OUT VARCHAR2,
      v_banco1            OUT VARCHAR2,
      v_banco2            OUT VARCHAR2,
      v_banco3            OUT VARCHAR2
    )
    IS
      CURSOR C_REFTDC IS
      SELECT *
      FROM   REFTARJETAS
      WHERE  CODCLIENTE = P_ID;
    BEGIN
        FOR FILA IN C_REFTDC LOOP
          v_numero1 := FILA.NUMERO1;
          v_numero2 := FILA.NUMERO2;
          v_numero3 := FILA.NUMERO3;
          v_tipo1 := FILA.TIPO1;
          v_tipo2 := FILA.TIPO2;
          v_tipo3 := FILA.TIPO3;
          v_banco1 := FILA.BANCO1;
          v_banco2 := FILA.BANCO2;
          v_banco3 := FILA.BANCO3;
        END LOOP;
        IF C_REFTDC%ISOPEN THEN
          CLOSE C_REFTDC;
        END IF;
    END SP_GETREFTARJETAS_FORMS;
```

--PROCEDURE SP_SETTRABAJOCLIENTE

--Muestra los datos del trabajo del cliente

```
    PROCEDURE SP_SETTRABAJOCLIENTE
    (
     P_ID             IN NUMBER   DEFAULT NULL,
     v_empresa          IN VARCHAR2 DEFAULT NULL,
     v_actividad        IN VARCHAR2 DEFAULT NULL,
     v_ciudad           IN VARCHAR2 DEFAULT NULL,
     v_direccion        IN VARCHAR2 DEFAULT NULL,
     v_cargo            IN VARCHAR2 DEFAULT NULL,
     v_anios            IN NUMBER   DEFAULT NULL,
     v_meses            IN NUMBER   DEFAULT NULL,
     v_ingresos         IN NUMBER   DEFAULT NULL,
     v_otrosvalor       IN NUMBER   DEFAULT NULL,
     v_otrosdetalle     IN VARCHAR2 DEFAULT NULL,
     v_empresaant       IN VARCHAR2 DEFAULT NULL,
     v_telanterior      IN VARCHAR2 DEFAULT NULL,
     v_antiguedad       IN VARCHAR2 DEFAULT NULL,
     v_cargoant         IN VARCHAR2 DEFAULT NULL
    )
    IS
    BEGIN
       IF FN_CLIENTE_EXISTE(P_ID => P_ID) = FALSE THEN
       --Insert
          MSG:= 'UN ERROR HA OCURRIDO AL INSERTAR TRABAJO DEL CLIENTE';
          INSERT INTO trabajocliente
            (codcliente, empresa, actividad, ciudad, direccion, cargo, anios, meses, ingresos,
otrosvalor, otrosdetalle, empresaant, telanterior, antiguedad, cargoant)
          VALUES
            (P_ID, v_empresa, v_actividad, v_ciudad, v_direccion, v_cargo, v_anios, v_meses,
v_ingresos, v_otrosvalor, v_otrosdetalle, v_empresaant, v_telanterior, v_antiguedad,
v_cargoant);
       ELSE

       --Update
          MSG:= 'UN ERROR HA OCURRIDO AL EDITAR TRABAJO DEL CLIENTE';
          UPDATE TRABAJOCLIENTE
          SET    empresa = v_empresa,
                 actividad = v_actividad,
                 ciudad = v_ciudad,
                 direccion = v_direccion,
                 cargo = v_cargo,
                 anios = v_anios,
                 meses = v_meses,
                 ingresos = v_ingresos,
                 otrosvalor = v_otrosvalor,
                 otrosdetalle = v_otrosdetalle,
                 empresaant = v_empresaant,
                 telanterior = v_telanterior,
                 antiguedad = v_antiguedad,
                 cargoant = v_cargoant
          WHERE  CODCLIENTE = P_ID;
       END IF;
       COMMIT;
       EXCEPTION
       WHEN EXP_ERROR THEN
          RAISE_APPLICATION_ERROR(-20000, MSG);
```

```
            ROLLBACK;
          WHEN OTHERS THEN
            RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO AL
GRABAR SOLICITUD DATOS TRABAJO CLIENTE');
            ROLLBACK;
      END SP_SETTRABAJOCLIENTE;

      -- PROCEDURE SP_GETTRABAJOCLIENTE_FORMS
      -- Obtiene los datos del trabajo del Cliente que aplica una solicitud de crédito

      PROCEDURE SP_GETTRABAJOCLIENTE_FORMS
      (
       P_ID              IN NUMBER   DEFAULT NULL,
       v_empresa          OUT VARCHAR2,
       v_actividad        OUT VARCHAR2,
       v_ciudad           OUT VARCHAR2,
       v_direccion        OUT VARCHAR2,
       v_cargo            OUT VARCHAR2,
       v_anios            OUT NUMBER,
       v_meses            OUT NUMBER,
       v_ingresos         OUT NUMBER,
       v_otrosvalor       OUT NUMBER,
       v_otrosdetalle     OUT VARCHAR2,
       v_empresaant       OUT VARCHAR2,
       v_telanterior      OUT VARCHAR2,
       v_antiguedad       OUT VARCHAR2,
       v_cargoant         OUT VARCHAR2
      )
      IS
      CURSOR C_TRABCLI IS
      SELECT * FROM   TRABAJOCLIENTE
      WHERE  CODCLIENTE = P_ID;
      BEGIN
         FOR FILA IN C_TRABCLI LOOP
           v_empresa := FILA.EMPRESA;
           v_actividad := FILA.ACTIVIDAD;
           v_ciudad := FILA.CIUDAD;
           v_direccion := FILA.DIRECCION;
           v_cargo := FILA.CARGO;
           v_anios := NVL(FILA.ANIOS,0);
           v_meses := NVL(FILA.MESES,0);
           v_ingresos := NVL(FILA.INGRESOS,0);
           v_otrosvalor := NVL(FILA.OTROSVALOR,0);
           v_otrosdetalle := FILA.OTROSDETALLE;
           v_empresaant := FILA.EMPRESAANT;
           v_telanterior := FILA.TELANTERIOR;
           v_antiguedad := FILA.ANTIGUEDAD;
           v_cargoant := FILA.CARGOANT;
         END LOOP;
         IF C_TRABCLI%ISOPEN THEN
           CLOSE C_TRABCLI;
         END IF;
      END SP_GETTRABAJOCLIENTE_FORMS;

      -- PROCEDURE SP_SETTRABAJOCONYUGE
      --Muestra o devuelve los datos del trabajo del cónyuge

      PROCEDURE SP_SETTRABAJOCONYUGE
      (
```

```
    P_ID              IN NUMBER   DEFAULT NULL,
    v_empresa         IN VARCHAR2 DEFAULT NULL,
    v_actividad       IN VARCHAR2 DEFAULT NULL,
    v_ciudad          IN VARCHAR2 DEFAULT NULL,
    v_direccion       IN VARCHAR2 DEFAULT NULL,
    v_cargo           IN VARCHAR2 DEFAULT NULL,
    v_anios           IN NUMBER   DEFAULT NULL,
    v_meses           IN NUMBER   DEFAULT NULL,
    v_ingresos        IN NUMBER   DEFAULT NULL,
    v_otrosvalor      IN NUMBER   DEFAULT NULL,
    v_otrosdetalle    IN VARCHAR2 DEFAULT NULL,
    v_empresaant      IN VARCHAR2 DEFAULT NULL,
    v_telanterior     IN VARCHAR2 DEFAULT NULL,
    v_antiguedad      IN VARCHAR2 DEFAULT NULL,
    v_cargoant        IN VARCHAR2 DEFAULT NULL,
    v_teltrabajo      IN VARCHAR2 DEFAULT NULL,
    v_dirtrabajo      IN VARCHAR2 DEFAULT NULL
  )
  IS
  BEGIN
    IF FN_CLIENTE_EXISTE(P_ID => P_ID) = FALSE THEN

    --Insert
      MSG:= 'UN ERROR HA OCURRIDO AL INSERTAR TRABAJO DEL
CONYUGE';
      INSERT INTO trabajoconyuge
        (codcliente, empresa, actividad, ciudad, direccion, cargo, anios, meses, ingresos,
otrosvalor, otrosdetalle, empresaant, telanterior, antiguedad, cargoant, teltrabajo, dirtrabajo)
      VALUES
        (P_ID, v_empresa, v_actividad, v_ciudad, v_direccion, v_cargo, v_anios, v_meses,
v_ingresos, v_otrosvalor, v_otrosdetalle, v_empresaant, v_telanterior, v_antiguedad,
v_cargoant, v_teltrabajo, v_dirtrabajo);
    ELSE

    --Update
      MSG:= 'UN ERROR HA OCURRIDO AL EDITAR TRABAJO DEL CONYUGE';
      UPDATE TRABAJOCONYUGE
      SET    empresa = v_empresa,
             actividad = v_actividad,
             ciudad = v_ciudad,
             direccion = v_direccion,
             cargo = v_cargo,
             anios = v_anios,
             meses = v_meses,
             ingresos = v_ingresos,
             otrosvalor = v_otrosvalor,
             otrosdetalle = v_otrosdetalle,
             empresaant = v_empresaant,
             telanterior = v_telanterior,
             antiguedad = v_antiguedad,
             cargoant = v_cargoant,
             teltrabajo = v_teltrabajo,
             dirtrabajo = v_dirtrabajo
      WHERE  CODCLIENTE = P_ID;
    END IF;
    COMMIT;
    EXCEPTION
    WHEN EXP_ERROR THEN
       RAISE_APPLICATION_ERROR(-20000, MSG);
```

```
            ROLLBACK;
        WHEN OTHERS THEN
            RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO AL
GRABAR SOLICITUD DATOS TRABAJO CONYUGE');
            ROLLBACK;
    END SP_SETTRABAJOCONYUGE;
```

-- PROCEDURE SP_GETTRABAJOCONYUGE_FORMS

-- Obtiene los datos del trabajo del conyuge que aplica una solicitud de crédito

```
    PROCEDURE SP_GETTRABAJOCONYUGE_FORMS
    (
     P_ID             IN NUMBER   DEFAULT NULL,
     v_empresa          OUT VARCHAR2,
     v_actividad        OUT VARCHAR2,
     v_ciudad           OUT VARCHAR2,
     v_direccion        OUT VARCHAR2,
     v_cargo            OUT VARCHAR2,
     v_anios            OUT NUMBER,
     v_meses            OUT NUMBER,
     v_ingresos         OUT NUMBER,
     v_otrosvalor       OUT NUMBER,
     v_otrosdetalle     OUT VARCHAR2,
     v_empresaant       OUT VARCHAR2,
     v_telanterior      OUT VARCHAR2,
     v_antiguedad       OUT VARCHAR2,
     v_cargoant         OUT VARCHAR2,
     v_teltrabajo       OUT VARCHAR2,
     v_dirtrabajo       OUT VARCHAR2
    )
    IS
    CURSOR C_TRABCON IS
    SELECT * FROM   TRABAJOCONYUGE
    WHERE  CODCLIENTE = P_ID;
    BEGIN
       FOR FILA IN C_TRABCON LOOP
         v_empresa := FILA.EMPRESA;
         v_actividad := FILA.ACTIVIDAD;
         v_ciudad := FILA.CIUDAD;
         v_direccion := FILA.DIRECCION;
         v_cargo := FILA.CARGO;
         v_anios := NVL(FILA.ANIOS,0);
         v_meses := NVL(FILA.MESES,0);
         v_ingresos := NVL(FILA.INGRESOS,0);
         v_otrosvalor := NVL(FILA.OTROSVALOR,0);
         v_otrosdetalle := FILA.OTROSDETALLE;
         v_empresaant := FILA.EMPRESAANT;
         v_telanterior := FILA.TELANTERIOR;
         v_antiguedad := FILA.ANTIGUEDAD;
         v_cargoant := FILA.CARGOANT;
         v_teltrabajo := FILA.TELTRABAJO;
         v_dirtrabajo := FILA.DIRTRABAJO;
       END LOOP;
       IF C_TRABCON%ISOPEN THEN
         CLOSE C_TRABCON;
       END IF;
    END SP_GETTRABAJOCONYUGE_FORMS;
```

--PROCEDURE SP_SETVIVIENDA

```
-- Devuelde los datos de las Viviendas del cliente
    PROCEDURE SP_SETVIVIENDA
    (
      P_ID              IN  NUMBER,
      V_CIUDAD          IN     VARCHAR2,
      V_SECTOR          IN     VARCHAR2,
      V_ANIOS           IN     NUMBER,
      V_MESES           IN     NUMBER,
      V_TIPOVIVIENDA       IN VARCHAR2,
      V_VALORALQUILER         IN   NUMBER,
      V_ARRENDADOR        IN       VARCHAR2,
      V_TELARRENDADOR         IN   VARCHAR2,
      V_ANTIGUEDAD        IN       VARCHAR2,
      V_DIRANTERIOR       IN VARCHAR2,
      V_TELANTERIOR       IN VARCHAR2
    )
    IS
    BEGIN
       IF FN_CLIENTE_EXISTE(P_ID => P_ID) = FALSE THEN
        --INSERT
          MSG:= 'UN ERROR HA OCURRIDO AL INSERTAR DATOS DE LA
VIVIENDA';
          INSERT INTO VIVIENDAS
               (codcliente, ciudad, sector, anios, meses, tipovivienda, valoralquiler, arrendador,
telarrendador, antiguedad, diranterior, telanterior)
          VALUES
               (P_ID, v_ciudad, v_sector, v_anios, v_meses, v_tipovivienda, v_valoralquiler,
v_arrendador, v_telarrendador, v_antiguedad, v_diranterior, v_telanterior);
        ELSE

        --Update
          MSG:= 'UN ERROR HA OCURRIDO AL EDITAR DATOS DE LA VIVIENDA';
          UPDATE VIVIENDAS
          SET    ciudad = v_ciudad,
               sector = v_sector,
               anios = v_anios,
               meses = v_meses,
               tipovivienda = v_tipovivienda,
               valoralquiler = v_valoralquiler,
               arrendador = v_arrendador,
               telarrendador = v_telarrendador,
               antiguedad = v_antiguedad,
               diranterior = v_diranterior,
               telanterior = v_telanterior
          WHERE  CODCLIENTE = P_ID;
        END IF;
        COMMIT;
        EXCEPTION
        WHEN EXP_ERROR THEN
           RAISE_APPLICATION_ERROR(-20000, MSG);
           ROLLBACK;
        WHEN OTHERS THEN
           RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO AL
GRABAR SOLICITUD DATOS VIVIENDA');
           ROLLBACK;
    END SP_SETVIVIENDA;

   -- PROCEDURE SP_GETVIVIENDA_FORMS
   -- Obtiene los datos del cliente en lo referente a la Vivienda
```

```
PROCEDURE SP_GETVIVIENDA_FORMS
(
  P_ID              IN  NUMBER,
  V_CIUDAD              OUT  VARCHAR2,
  V_SECTOR              OUT  VARCHAR2,
  V_ANIOS               OUT  NUMBER,
  V_MESES               OUT  NUMBER,
  V_TIPOVIVIENDA        OUT       VARCHAR2,
  V_VALORALQUILER           OUT         NUMBER,
  V_ARRENDADOR          OUT      VARCHAR2,
  V_TELARRENDADOR           OUT         VARCHAR2,
  V_ANTIGUEDAD          OUT      VARCHAR2,
  V_DIRANTERIOR         OUT       VARCHAR2,
  V_TELANTERIOR         OUT       VARCHAR2
)
IS
CURSOR C_VIVIENDAS IS
SELECT * FROM VIVIENDAS
WHERE CODCLIENTE = P_ID;
BEGIN
   FOR FILA IN C_VIVIENDAS LOOP
     V_CIUDAD          := FILA.CIUDAD;
     V_SECTOR          := FILA.SECTOR;
     V_ANIOS   := NVL(FILA.ANIOS,0);
     V_MESES  := NVL(FILA.MESES,0);
     V_TIPOVIVIENDA        := FILA.TIPOVIVIENDA;
     V_VALORALQUILER   := FILA.VALORALQUILER;
     V_ARRENDADOR          := FILA.ARRENDADOR;
     V_TELARRENDADOR   := FILA.TELARRENDADOR;
     V_ANTIGUEDAD:= FILA.ANTIGUEDAD;
     V_DIRANTERIOR         := FILA.DIRANTERIOR;
     V_TELANTERIOR         := FILA.TELANTERIOR;
   END LOOP;

   IF C_VIVIENDAS%ISOPEN THEN
     CLOSE C_VIVIENDAS;
   END IF;
END SP_GETVIVIENDA_FORMS;
```

-- PROCEDURE SP_SETCLIENTE

-- Devuelve los datos de los clientes a utilizar en la solicitud de crédito

```
PROCEDURE SP_SETCLIENTE
(
   P_ID              IN  NUMBER    DEFAULT NULL,
   V_CEDULA              IN VARCHAR2   DEFAULT NULL,
   V_APELLIDOS           IN VARCHAR2   DEFAULT NULL,
   V_NOMBRES             IN VARCHAR2   DEFAULT NULL,
   V_TELCASA             IN VARCHAR2   DEFAULT NULL,
   V_TELTRABAJO          IN VARCHAR2   DEFAULT NULL,
   V_CELULAR             IN VARCHAR2   DEFAULT NULL,
   V_DIRCASA             IN VARCHAR2   DEFAULT NULL,
   V_DIRTRABAJO          IN VARCHAR2   DEFAULT NULL,
   V_NACIONALIDAD        IN VARCHAR2   DEFAULT NULL,
```

```
    V_FECHANAC          IN DATE      DEFAULT NULL,
    V_TIPOLABORAL       IN VARCHAR2  DEFAULT NULL,
    V_ESTADOCIVIL       IN VARCHAR2  DEFAULT NULL,
    V_PROFESION         IN VARCHAR2  DEFAULT NULL,
    V_SEXO              IN VARCHAR2  DEFAULT NULL,
    V_ESTUDIOS          IN VARCHAR2  DEFAULT NULL,
    V_CARGAS            IN VARCHAR2  DEFAULT NULL
  )
  IS
  BEGIN
    UPDATE CLIENTES SET
      CEDULA = V_CEDULA,
      APELLIDOS = V_APELLIDOS,
      NOMBRES = V_NOMBRES,
      TELCASA = V_TELCASA,
      TELTRABAJO = V_TELTRABAJO,
      CELULAR = V_CELULAR,
      DIRCASA = V_DIRCASA,
      DIRTRABAJO = V_DIRTRABAJO,
      NACIONALIDAD = V_NACIONALIDAD,
      FECHANAC = V_FECHANAC,
      TIPOLABORAL = V_TIPOLABORAL,
      ESTADOCIVIL = V_ESTADOCIVIL,
      PROFESION = V_PROFESION,
      SEXO = V_SEXO,
      ESTUDIOS = V_ESTUDIOS,
      CARGAS = V_CARGAS
    WHERE CODCLIENTE = P_ID;
    COMMIT;
    EXCEPTION
    WHEN OTHERS THEN
       RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO AL
GRABAR SOLICITUD DATOS CLIENTE');
       ROLLBACK;
  END SP_SETCLIENTE;

 -- PROCEDURE SP_GETCLIENTE_FORMS
  /*Obtiene los datos de los clientes que son necesarios para llenar la solicitud de crédito*/

  PROCEDURE SP_GETCLIENTE_FORMS
  (
    P_ID              IN  NUMBER,
    V_CEDULA          OUT VARCHAR2,
    V_APELLIDOS       OUT VARCHAR2,
    V_NOMBRES         OUT VARCHAR2,
    V_TELCASA         OUT VARCHAR2,
    V_TELTRABAJO      OUT VARCHAR2,
    V_CELULAR         OUT VARCHAR2,
    V_DIRCASA         OUT VARCHAR2,
    V_DIRTRABAJO      OUT VARCHAR2,
    V_CODVEHICULO     OUT NUMBER,
    V_PRECIO          OUT NUMBER,
    V_ENTRADA         OUT NUMBER,
    V_PLAZO           OUT NUMBER,
    V_GASTOS          OUT NUMBER,
    V_CODDISPOSITIVO  OUT NUMBER,
    V_NACIONALIDAD    OUT VARCHAR2,
    V_FECHANAC        OUT DATE,
    V_TIPOLABORAL     OUT VARCHAR2,
```

```
        V_ESTADOCIVIL          OUT VARCHAR2,
        V_PROFESION            OUT VARCHAR2,
        V_SEXO                 OUT VARCHAR2,
        V_ESTUDIOS             OUT VARCHAR2,
        V_CARGAS               OUT VARCHAR2,
        --EXTRAS
        V_MODELO               OUT VARCHAR2,
        V_DISPOSITIVO          OUT VARCHAR2
    )
    IS
    CURSOR C_CLIENTE IS
    SELECT
        C.CEDULA, C.APELLIDOS, C.NOMBRES, C.TELCASA,
        C.TELTRABAJO, C.CELULAR, C.DIRCASA, C.DIRTRABAJO,
C.CODVEHICULO,
        C.PRECIO, C.ENTRADA, C.PLAZO, C.GASTOS,
        C.CODDISPOSITIVO, C.NACIONALIDAD, C.FECHANAC, C.TIPOLABORAL,
        C.ESTADOCIVIL, C.PROFESION, C.SEXO, C.ESTUDIOS, C.CARGAS,
        V.MODELO, D.NOMBRE AS DISPOSITIVO
    FROM  CLIENTES C INNER JOIN VEHICULOS V
    ON    C.CODVEHICULO = V.CODVEHICULO
    INNER JOIN DISPOSITIVOS D
    ON    C.CODDISPOSITIVO = D.CODDISPOSITIVO
    WHERE C.CODCLIENTE = P_ID;
    BEGIN
       FOR FILA IN C_CLIENTE LOOP
          V_CEDULA := FILA.CEDULA;
          V_APELLIDOS := FILA.APELLIDOS;
          V_NOMBRES := FILA.NOMBRES;
          V_TELCASA := FILA.TELCASA;
          V_TELTRABAJO := FILA.TELTRABAJO;
          V_CELULAR := FILA.CELULAR;
          V_DIRCASA := FILA.DIRCASA;
          V_DIRTRABAJO := FILA.DIRTRABAJO;
          V_CODVEHICULO := FILA.CODVEHICULO;
          V_PRECIO := FILA.PRECIO;
          V_ENTRADA := FILA.ENTRADA;
          V_PLAZO := FILA.PLAZO;
          V_GASTOS := FILA.GASTOS;
          V_CODDISPOSITIVO := FILA.CODDISPOSITIVO;
          V_NACIONALIDAD := FILA.NACIONALIDAD;
          V_FECHANAC := FILA.FECHANAC;
          V_TIPOLABORAL := FILA.TIPOLABORAL;
          V_ESTADOCIVIL := FILA.ESTADOCIVIL;
          V_PROFESION := FILA.PROFESION;
          V_SEXO := FILA.SEXO;
          V_ESTUDIOS := FILA.ESTUDIOS;
          V_CARGAS := FILA.CARGAS;
          V_MODELO := FILA.MODELO;
          V_DISPOSITIVO := FILA.DISPOSITIVO;
       END LOOP;
       IF C_CLIENTE%ISOPEN THEN
         CLOSE C_CLIENTE;
       END IF;
    END SP_GETCLIENTE_FORMS;
END PK_SOLICITUD;
/
```

PK_USUARIOS.
Objetivo: Permitirà mediante procedimientos almacenados la Creación, Modificación, Eliminación y Consulta de los Usuarios.

prompt **Creating package body PK_USUARIOS**
prompt ==========

```
CREATE OR REPLACE PACKAGE BODY USER15.PK_USUARIOS AS
  --Declaracion de variables al inicio
  R NUMBER;
  MSG VARCHAR2(100);
  C NUMBER;
  CONTADOR NUMBER;
  NEW_ID NUMBER;
  EXP_ERROR EXCEPTION;

  --Con esto cargo un usuario en base a un id
  PROCEDURE SP_CARGAR (MY_CURSOR IN OUT T_CURSOR, P_ID NUMBER)
  IS
  BEGIN
    OPEN MY_CURSOR FOR
    SELECT CODUSUARIO, USUARIO, CLAVE, NOMBRE, CARGO, ACTIVO
    FROM USUARIOS WHERE CODUSUARIO = P_ID;
  END SP_CARGAR;

-- PROCEDURE SP_ELIMINAR
--Proc. Para eliminar logicamente al usuario
--Elimina un usuario del sistema.
  PROCEDURE SP_ELIMINAR (P_ID NUMBER, P_ACTIVO IN NUMBER DEFAULT
NULL)
  IS
  BEGIN
    SELECT COUNT (USUARIO) INTO C FROM USUARIOS WHERE CODUSUARIO =
P_ID;
    IF (C = 0) THEN
       MSG:= 'EL NOMBRE YA ESTA ASIGNADO A OTRO BANCO';
       RAISE EXP_ERROR;
    ELSE
      UPDATE USUARIOS SET ACTIVO = P_ACTIVO WHERE CODUSUARIO = P_ID;
      COMMIT;
    END IF;
    EXCEPTION
      WHEN EXP_ERROR THEN
          RAISE_APPLICATION_ERROR(-20000, MSG);
          ROLLBACK;
      WHEN OTHERS THEN
          RAISE_APPLICATION_ERROR(-20000, 'ERROR: ' || SQLERRM);
          ROLLBACK;
  END SP_ELIMINAR;

-- PROCEDURE SP_INSERTAR
 --Procedimiento para ingresar usuario nuevo
/*Pemite ingresar el nombre, el id y el rol del usuario dentro del sistema*/

SP_INSERTAR(P_LOGIN VARCHAR2, P_NOMBRE VARCHAR2, P_ROL VARCHAR2,
MY_CURSOR OUT NUMBER)
  IS
```

```
  BEGIN
    R:=0;

    --Si id=0 inserto
    SELECT COUNT(USUARIO) INTO C FROM USER15.USUARIOS WHERE USUARIO
LIKE P_LOGIN;
    IF (C > 0) THEN
      MSG:='EL INICIO DE SESIÓN YA EXISTE';
      R:=1;
    END IF;
    SELECT COUNT(USUARIO) INTO C FROM USER15.USUARIOS WHERE NOMBRE
LIKE P_NOMBRE;
    IF (C > 0) THEN
      MSG:='EL NOMBRE DEL USUARIO YA EXISTE';
      R:=1;
    END IF;

    --R=0 ok // r=1 error
    IF (R=0) THEN
      SELECT MAX(CODUSUARIO)+1 INTO NEW_ID FROM USUARIOS;
      NEW_ID:= NVL(NEW_ID,1);
      INSERT INTO USER15.USUARIOS(CODUSUARIO, USUARIO, NOMBRE, CARGO,
      ACTIVO)
      VALUES(NEW_ID, P_LOGIN, P_NOMBRE, P_ROL, 1);
      COMMIT;
      SELECT NEW_ID INTO MY_CURSOR FROM DUAL;
    ELSE
      RAISE EXP_ERROR;
    END IF;
    EXCEPTION
      WHEN EXP_ERROR THEN
          RAISE_APPLICATION_ERROR(-20000, MSG);
          ROLLBACK;
      WHEN OTHERS THEN
          RAISE_APPLICATION_ERROR(-20000, 'ERROR: ' || SQLERRM);
          ROLLBACK;
  END SP_INSERTAR;

--PROCEDURE SP_GUARDAR
 --Guarda y actualiza la base de datos

PROCEDURE SP_GUARDAR (P_ID NUMBER, P_LOGIN VARCHAR2, P_NOMBRE
VARCHAR2, P_ROL VARCHAR2, P_ACTIVO NUMBER)
  IS
  BEGIN
    R:=0;
    --Si id<>0 actualizo

    SELECT COUNT(USUARIO) INTO C FROM USER15.USUARIOS WHERE
CODUSUARIO = P_ID;
    IF (C = 0) THEN
      MSG:='EL USUARIO NO SE PUDO ENCONTRAR';
      R:=1;
    END IF;
    SELECT COUNT (USUARIO) INTO C FROM USER15.USUARIOS WHERE USUARIO
LIKE P_LOGIN AND CODUSUARIO <> P_ID;
    IF (C > 0) THEN
      MSG:='EL INICIO DE SESION YA ESTA TOMADO POR OTRO USUARIO';
      R:=1;
```

```
   END IF;
   SELECT COUNT(USUARIO) INTO C FROM USER15.USUARIOS WHERE NOMBRE
LIKE P_NOMBRE AND CODUSUARIO <> P_ID;
   IF (C > 0) THEN
    MSG:='EL NOMBRE DE USUARIO YA ESTA TOMADO POR OTRO USUARIO';
    R:=1;
   END IF;
   IF (R=0) THEN
    UPDATE USUARIOS SET
    USUARIO = P_LOGIN, NOMBRE = P_NOMBRE,
    CARGO = P_ROL, ACTIVO = P_ACTIVO
    WHERE CODUSUARIO = P_ID;
   ELSE
    RAISE EXP_ERROR;
   END IF;
   EXCEPTION
    WHEN EXP_ERROR THEN
        RAISE_APPLICATION_ERROR(-20000, MSG);
        ROLLBACK;
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20000, 'ERROR: ' || SQLERRM);
        ROLLBACK;
 END SP_GUARDAR;
```

--PROCEDURE SP_CONSULTAR

/ Permite realizar una consulta por usuario*/*

```
 PROCEDURE SP_CONSULTAR(MY_CURSOR IN OUT T_CURSOR, P_TODOS
NUMBER)
 IS
 BEGIN
   IF (P_TODOS=1) THEN
    OPEN MY_CURSOR FOR
    SELECT * FROM USER15.USUARIOS WHERE ACTIVO=1 ORDER BY USUARIO;
   ELSE
    OPEN MY_CURSOR FOR
    SELECT * FROM USER15.USUARIOS ORDER BY USUARIO;
   END IF;
 END SP_CONSULTAR;
```

--PROCEDURE SP_CONSULTARPORNOMBRE

/ Consulta por nombre de un usuario específico*/*

```
PROCEDURE SP_CONSULTARPORNOMBRE(MY_CURSOR IN OUT T_CURSOR,
P_NOMBRE VARCHAR2, P_OPCION VARCHAR2)
 IS
 BEGIN
   IF (P_OPCION='N') THEN
    OPEN MY_CURSOR FOR
    SELECT * FROM USER15.USUARIOS
    WHERE NOMBRE LIKE P_NOMBRE
    ORDER BY NOMBRE;
   ELSE
    OPEN MY_CURSOR FOR
    SELECT * FROM USER15.USUARIOS
    WHERE USUARIO LIKE P_NOMBRE
    ORDER BY USUARIO;
   END IF;
 END SP_CONSULTARPORNOMBRE;
```

PROCEDURE SP_CAMBIARCLAVE
-- Permite cambiar la clave de acceso del usuario.

```
 PROCEDURE SP_CAMBIARCLAVE(P_ID NUMBER, P_OLDCLAVE VARCHAR2,
P_NEWCLAVE VARCHAR2)
  IS
  BEGIN
    MSG := 'LA CLAVE ANTERIOR ES INCORRECTA, NO PUEDE CAMBIAR SU
CLAVE';
    SELECT COUNT(CODUSUARIO) INTO R FROM USUARIOS WHERE CLAVE =
P_OLDCLAVE AND CODUSUARIO = P_ID;
    IF R = 0 THEN
      RAISE EXP_ERROR;
    ELSE
      UPDATE USUARIOS SET CLAVE = P_NEWCLAVE WHERE CODUSUARIO =
P_ID;
      COMMIT;
    END IF;
    EXCEPTION
     WHEN EXP_ERROR THEN
         RAISE_APPLICATION_ERROR(-20000, MSG);
         ROLLBACK;
     WHEN OTHERS THEN
         RAISE_APPLICATION_ERROR(-20000, 'ERROR: ' || SQLERRM);
         ROLLBACK;
  END SP_CAMBIARCLAVE;
```

--PROCEDURE SP_ASIGNARCLAVE
/ Permite asignar una clave de acceso al usuario*/*

```
  PROCEDURE SP_ASIGNARCLAVE(P_ID IN NUMBER, P_CLAVE IN VARCHAR2)
  IS
  BEGIN
     UPDATE USUARIOS SET CLAVE = P_CLAVE WHERE CODUSUARIO = P_ID;
     COMMIT;
  END SP_ASIGNARCLAVE;
```

PROCEDURE SP_CONSULTARTODOS_FORMS
/ Permite consultar a todos los usuarios del sistema.*/*

```
  PROCEDURE SP_CONSULTARTODOS_FORMS
  (
   USUARIOS_DATA IN OUT T_USUARIO,
   P_ACTIVO      IN NUMBER       DEFAULT NULL,
   P_LOGIN       IN VARCHAR2     DEFAULT NULL
  )
  IS
   CURSOR CUR_USU IS
   SELECT * FROM USUARIOS WHERE ACTIVO = P_ACTIVO
   AND USUARIO LIKE '%' || P_LOGIN || '%'
   ORDER BY USUARIO;
  BEGIN
     OPEN CUR_USU;
        CONTADOR := 1;
        LOOP
        FETCH CUR_USU INTO
            USUARIOS_DATA(CONTADOR).CODUSUARIO,
            USUARIOS_DATA(CONTADOR).USUARIO,
```

```
            USUARIOS_DATA(CONTADOR).CLAVE,
            USUARIOS_DATA(CONTADOR).NOMBRE,
            USUARIOS_DATA(CONTADOR).CARGO,
            USUARIOS_DATA(CONTADOR).ACTIVO;
            EXIT WHEN CUR_USU%NOTFOUND;
            CONTADOR := CONTADOR + 1;
        END LOOP;
        IF CUR_USU%ISOPEN THEN
          CLOSE CUR_USU;
        END IF;
    EXCEPTION
        WHEN OTHERS THEN
        IF CUR_USU%ISOPEN THEN
          CLOSE CUR_USU;
        END IF;
  END SP_CONSULTARTODOS_FORMS;
END;
/
```

- ***PK_VEHICULOS.***

Objetivo: Administrar vehiculos. Contiene los Procedimientos Almacenados para la Creación, Modificación, Eliminación y Consulta de la Tabla VEHICULOS, que almacena los datos de los Vehículos, que se pueden emplear en Créditos y Ventas

```
prompt Creating package body PK_VEHICULOS
prompt ================================

CREATE OR REPLACE Package Body USER15.PK_VEHICULOS IS

    NEW_ID          NUMBER;
    MSG             VARCHAR2(100);
    CONTADOR        NUMBER;
    EXP_ERROR       EXCEPTION;

   -- PROCEDURE SP_INSERTAR
   -- Ingresa el regisro de un vehículo a la base de datos.

    PROCEDURE SP_INSERTAR
    (
      P_CODMARCA          IN NUMBER   DEFAULT NULL,
      P_MODELO            IN VARCHAR2 DEFAULT NULL,
      P_PRECIO            IN NUMBER   DEFAULT NULL,
      P_ANIO              IN NUMBER   DEFAULT NULL,
      P_TIPO              IN VARCHAR2 DEFAULT NULL,
      P_TONELAJE          IN NUMBER   DEFAULT NULL,
      P_KMS               IN NUMBER   DEFAULT NULL,
      P_CODCATEGORIA      IN NUMBER   DEFAULT NULL,
      P_ID                OUT NUMBER
    )
    IS
    BEGIN
        MSG:= 'NO SE PUDO INSERTAR VEHICULO';
        SELECT NVL(MAX(CODVEHICULO) + 1, 1) INTO NEW_ID FROM
VEHICULOS;
```

```
        INSERT INTO VEHICULOS
(CODVEHICULO,CODMARCA,MODELO,PRECIO,ANIO,TIPO,TONELAJE,KILOMETR
OS,CODCATEGORIA,ACTIVO)
        VALUES
(NEW_ID,P_CODMARCA,P_MODELO,P_PRECIO,P_ANIO,P_TIPO,P_TONELAJE,P_KM
S,P_CODCATEGORIA,1);
        SELECT NEW_ID INTO P_ID FROM DUAL;
        COMMIT;
        EXCEPTION
        WHEN EXP_ERROR THEN
            RAISE_APPLICATION_ERROR(-20000, MSG);
            ROLLBACK;
        WHEN OTHERS THEN
            RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO');
            ROLLBACK;
    END SP_INSERTAR;

    -- PROCEDURE SP_GUARDAR
    -- Guarda el vehículo que ha sido registrado correctamente al sistema

    PROCEDURE SP_GUARDAR
    (
      P_ID              IN NUMBER   DEFAULT NULL,
      P_CODMARCA        IN NUMBER   DEFAULT NULL,
      P_MODELO          IN VARCHAR2 DEFAULT NULL,
      P_PRECIO          IN NUMBER   DEFAULT NULL,
      P_ANIO            IN NUMBER   DEFAULT NULL,
      P_TIPO            IN VARCHAR2 DEFAULT NULL,
      P_TONELAJE        IN NUMBER   DEFAULT NULL,
      P_KMS             IN NUMBER   DEFAULT NULL,
      P_CODCATEGORIA    IN NUMBER   DEFAULT NULL,
      P_ACTIVO          IN NUMBER   DEFAULT NULL
    )
    IS
    BEGIN
        MSG:= 'NO SE PUDO GUARDAR VEHICULO';
        SELECT COUNT(*) INTO CONTADOR FROM VEHICULOS
        WHERE CODVEHICULO = P_ID;
        IF CONTADOR = 0 THEN
          MSG:= 'NO SE PUDO ENCONTRAR EL VEHICULO A MODIFICAR';
          RAISE EXP_ERROR;
        END IF;
        UPDATE VEHICULOS SET
        codmarca     = P_CODMARCA,
        modelo       = P_MODELO,
        precio       = P_PRECIO,
        anio         = P_ANIO,
        tipo         = P_TIPO,
        tonelaje     = P_TONELAJE,
        kilometros   = P_KMS,
        codcategoria = P_CODCATEGORIA,
        activo       = P_ACTIVO
        WHERE CODVEHICULO = P_ID;
        COMMIT;
        EXCEPTION
        WHEN EXP_ERROR THEN
            RAISE_APPLICATION_ERROR(-20000, MSG);
            ROLLBACK;
```

```
        WHEN OTHERS THEN
            RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO');
            ROLLBACK;
    END SP_GUARDAR;

    -- PROCEDURE SP_ELIMINAR
    /*Elimina un vehículo del sistema y actualiza la base de datos enviando el vehículo
eliminado a la papelera de reciclaje como estado inactivo*/

    PROCEDURE SP_ELIMINAR
    (
     P_ID            IN NUMBER DEFAULT NULL,
     P_ACTIVO        IN NUMBER DEFAULT NULL
    )
    IS
    BEGIN
        MSG:= 'NO SE PUDO ELIMINAR VEHICULO';
        SELECT COUNT(*) INTO CONTADOR FROM VEHICULOS
        WHERE CODVEHICULO = P_ID;
        IF CONTADOR = 0 THEN
          MSG:= 'NO SE PUDO ENCONTRAR EL VEHICULO';
          RAISE EXP_ERROR;
        END IF;

        --Valido para eliminar y restaurar dependiendo del parametro
        UPDATE VEHICULOS SET
        ACTIVO = P_ACTIVO
        WHERE CODVEHICULO = P_ID;
        COMMIT;
        EXCEPTION
        WHEN EXP_ERROR THEN
            RAISE_APPLICATION_ERROR(-20000, MSG);
            ROLLBACK;
        WHEN OTHERS THEN
            RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO');
            ROLLBACK;
    END SP_ELIMINAR;

    -- PROCEDURE SP_CONSULTAR
    -- Consulta de vehículos según el parametro de ingreso que realice el usuario

    PROCEDURE SP_CONSULTAR
    (
     P_ID            IN NUMBER DEFAULT NULL,
     MY_CURSOR       IN OUT T_CURSOR
    )
    IS
    BEGIN
        OPEN MY_CURSOR FOR
        SELECT * FROM VEHICULOS WHERE CODVEHICULO = P_ID;
    END SP_CONSULTAR;

    --PROCEDURE SP_CONSULTARPORMODELO
    -- Consulta por el modelo de un vehículo determinado

    PROCEDURE SP_CONSULTARPORMODELO
    (
```

```
    P_MODELO              IN VARCHAR2 DEFAULT NULL,
    P_CODCATEGORIA        IN NUMBER   DEFAULT NULL,
    MY_CURSOR IN OUT       T_CURSOR
  )
  IS
  BEGIN
     IF NVL(P_CODCATEGORIA,0) = 0 THEN
       OPEN MY_CURSOR FOR
       SELECT * FROM VISVEHICULOS
       WHERE MODELO LIKE '%' || P_MODELO || '%' AND ACTIVO = 1
       ORDER BY MODELO;
     ELSE
       OPEN MY_CURSOR FOR
       SELECT * FROM VISVEHICULOS
       WHERE MODELO LIKE '%' || P_MODELO || '%' AND ACTIVO = 1 AND
CODCATEGORIA = P_CODCATEGORIA
       ORDER BY MODELO;
     END IF;
  END SP_CONSULTARPORMODELO;

  -- PROCEDURE SP_CONSULTARPORMARCA
  -- Consulta por la marca de un determinado vehículo

  PROCEDURE SP_CONSULTARPORMARCA
  (
    P_CODMARCA            IN NUMBER   DEFAULT NULL,
    MY_CURSOR IN OUT       T_CURSOR
  )
  IS
  BEGIN
     OPEN MY_CURSOR FOR
     SELECT * FROM VISVEHICULOS
     WHERE CODMARCA = P_CODMARCA AND ACTIVO = 1
     ORDER BY MODELO;
  END SP_CONSULTARPORMARCA;

  -- PROCEDURE SP_CONSULTARPORTIPO
  -- Consulta por el tipo o modelo de un vehículo.

  PROCEDURE SP_CONSULTARPORTIPO
  (
    P_TIPO              IN VARCHAR2 DEFAULT NULL,
    MY_CURSOR IN OUT       T_CURSOR
  )
  IS
  BEGIN
     OPEN MY_CURSOR FOR
     SELECT * FROM VISVEHICULOS
     WHERE TIPO = P_TIPO AND ACTIVO = 1
     ORDER BY MODELO;
  END SP_CONSULTARPORTIPO;

  --PROCEDURE SP_CONSULTARPORPRECIOS
  -- Consulta por un precio determinado de un vehículo

  PROCEDURE SP_CONSULTARPORPRECIOS
  (
    P_PRECIO1             IN NUMBER   DEFAULT NULL,
    P_PRECIO2             IN NUMBER   DEFAULT NULL,
```

```
  MY_CURSOR IN OUT         T_CURSOR
)
IS
BEGIN
    OPEN MY_CURSOR FOR
    SELECT * FROM VISVEHICULOS
    WHERE PRECIO BETWEEN P_PRECIO1 AND P_PRECIO2 AND ACTIVO = 1
    ORDER BY MARCA, MODELO;
END SP_CONSULTARPORPRECIOS;

--PROCEDURE SP_CONSULTARTODOS_FORMS
-- Consulta todos los vehículos que se han registrado en la base de datos.

PROCEDURE SP_CONSULTARTODOS_FORMS
(
  CAR_DATA  IN OUT T_VEHICULO,
  P_ACTIVO  IN NUMBER     DEFAULT NULL,
  P_MODELO  IN VARCHAR2   DEFAULT NULL
)
IS
CURSOR CUR_VEHICULOS IS
SELECT * FROM VISVEHICULOS WHERE ACTIVO = P_ACTIVO
AND MODELO LIKE '%' || P_MODELO || '%'
ORDER BY MODELO;
BEGIN
    OPEN CUR_VEHICULOS;
    CONTADOR := 1;
    LOOP
    FETCH CUR_VEHICULOS INTO
        CAR_DATA(CONTADOR).CODVEHICULO,
        CAR_DATA(CONTADOR).CODMARCA,
        CAR_DATA(CONTADOR).CODCATEGORIA,
        CAR_DATA(CONTADOR).MARCA,
        CAR_DATA(CONTADOR).MODELO,
        CAR_DATA(CONTADOR).PRECIO,
        CAR_DATA(CONTADOR).ANIO,
        CAR_DATA(CONTADOR).CATEGORIA,
        CAR_DATA(CONTADOR).TIPO,
        CAR_DATA(CONTADOR).TONELAJE,
        CAR_DATA(CONTADOR).KILOMETROS,
        CAR_DATA(CONTADOR).ACTIVO;
        EXIT WHEN CUR_VEHICULOS%NOTFOUND;
        CONTADOR := CONTADOR + 1;
    END LOOP;
    IF CUR_VEHICULOS%ISOPEN THEN
      CLOSE CUR_VEHICULOS;
    END IF;
EXCEPTION
WHEN OTHERS THEN
    IF CUR_VEHICULOS%ISOPEN THEN
      CLOSE CUR_VEHICULOS;
    END IF;
END SP_CONSULTARTODOS_FORMS;
END PK_VEHICULOS;
/
```

- ***PK_VENTAS.***

  Objetivo : Administrar las Ventas de los Clientes

Contiene los Procedimientos Almacenados para la Creación, Modificación, Eliminación y Consulta de la Tabla Ventas.

```
prompt Creating package body PK_VENTAS
prompt ============================

CREATE OR REPLACE Package Body USER15.PK_VENTAS IS
     NEW_ID          NUMBER;
     MSG             VARCHAR2(100);
     CONTADOR        NUMBER;
     EXP_ERROR       EXCEPTION;
     V_CARGO         VARCHAR2(20);

--PROCEDURE SP_INSERTAR
-- Ingreso de todos los datos obligatorios para llenar la    --tabla Ventas.

PROCEDURE SP_INSERTAR

(
       P_CODCLIENTE     IN NUMBER     DEFAULT NULL,
       P_CODUSUARIO     IN NUMBER     DEFAULT NULL,
       P_CODBANCO       IN NUMBER     DEFAULT NULL,
       P_CODVEHICULO    IN NUMBER     DEFAULT NULL,
       P_CODDISPOSITIVO IN NUMBER     DEFAULT NULL,
       P_CODSEGURO      IN NUMBER     DEFAULT NULL,
       P_FACTURA        IN VARCHAR2   DEFAULT NULL,
       P_PRECIO         IN NUMBER     DEFAULT NULL,
       P_ENTRADA        IN NUMBER     DEFAULT NULL,
       P_PLAZO          IN NUMBER     DEFAULT NULL,
       P_GASTOS         IN NUMBER     DEFAULT NULL,
       P_FECHAFAC       IN DATE       DEFAULT NULL,
       P_FECHAENT       IN DATE       DEFAULT NULL,
       P_RAMV           IN VARCHAR2   DEFAULT NULL,
       P_MOTOR          IN VARCHAR2   DEFAULT NULL,
       P_CHASIS         IN VARCHAR2   DEFAULT NULL,
       P_COLOR          IN VARCHAR2   DEFAULT NULL,
       P_ANIO           IN NUMBER     DEFAULT NULL,
       P_ACCESORIOS     IN CLOB       DEFAULT NULL,
       P_CODCREDITO     IN NUMBER     DEFAULT NULL,
       P_ID             OUT           NUMBER
)
IS
BEGIN
   MSG:= 'NO SE PUDO INSERTAR FACTURA';
   SELECT COUNT(CODVENTA) INTO CONTADOR
   FROM VENTAS WHERE CODCREDITO = NVL(P_CODCREDITO,-1);
   IF CONTADOR > 0 AND P_CODCREDITO <> 0 THEN
     MSG := 'YA EXISTE UN VENTA VINCULADA A ESTE CREDITO, NO SE PUEDE
      VOLVER A FACTURAR';
     RAISE EXP_ERROR;
   END IF;
   SELECT COUNT(CODVENTA) INTO CONTADOR
   FROM VENTAS WHERE FACTURA = P_FACTURA;
   IF CONTADOR > 0 THEN
     MSG := 'YA EXISTE UNA FACTURA CON ESTE NUMERO, INGRESE OTRO
NUMERO DE FACTURA';
     RAISE EXP_ERROR;
   END IF;
   SELECT NVL(MAX(CODVENTA) + 1, 1) INTO NEW_ID FROM VENTAS;
```

```
   INSERT INTO VENTAS
    (codventa, codcliente, codusuario, codbanco, codvehiculo, coddispositivo, codseguro,
     factura, precio, entrada, plazo, gastos, fechafac, fechaent, ramv, motor, chasis,
     color, anio, accesorios, codcredito)
   VALUES
    (NEW_ID, P_CODCLIENTE, P_CODUSUARIO, P_CODBANCO, P_CODVEHICULO,
P_CODDISPOSITIVO, P_CODSEGURO,
     P_FACTURA, P_PRECIO, P_ENTRADA, P_PLAZO, P_GASTOS, P_FECHAFAC,
P_FECHAENT, P_RAMV, P_MOTOR,
     P_CHASIS, P_COLOR, P_ANIO, P_ACCESORIOS, P_CODCREDITO);
   COMMIT;
   SELECT NEW_ID INTO P_ID FROM DUAL;
   EXCEPTION
   WHEN EXP_ERROR THEN
      RAISE_APPLICATION_ERROR(-20000, MSG);
      ROLLBACK;
   WHEN OTHERS THEN
      RAISE_APPLICATION_ERROR(-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO' || SQLERRM);
      ROLLBACK;
END SP_INSERTAR;
```

--PROCEDURE SP_GUARDAR

--Guardar cambios y actualiza la Base de Datos con los datos insertados.

```
      PROCEDURE SP_GUARDAR  (
       P_CODVENTA       IN NUMBER     DEFAULT NULL,
       P_CODCLIENTE     IN NUMBER     DEFAULT NULL,
       P_CODUSUARIO     IN NUMBER     DEFAULT NULL,
       P_CODBANCO       IN NUMBER     DEFAULT NULL,
       P_CODVEHICULO    IN NUMBER     DEFAULT NULL,
       P_CODDISPOSITIVO IN NUMBER     DEFAULT NULL,
       P_CODSEGURO      IN NUMBER     DEFAULT NULL,
       P_FACTURA        IN VARCHAR2   DEFAULT NULL,
       P_PRECIO         IN NUMBER     DEFAULT NULL,
       P_ENTRADA        IN NUMBER     DEFAULT NULL,
       P_PLAZO          IN NUMBER     DEFAULT NULL,
       P_GASTOS         IN NUMBER     DEFAULT NULL,
       P_FECHAFAC       IN DATE       DEFAULT NULL,
       P_FECHAENT       IN DATE       DEFAULT NULL,
       P_RAMV           IN VARCHAR2   DEFAULT NULL,
       P_MOTOR          IN VARCHAR2   DEFAULT NULL,
       P_CHASIS         IN VARCHAR2   DEFAULT NULL,
       P_COLOR          IN VARCHAR2   DEFAULT NULL,
       P_ANIO           IN NUMBER     DEFAULT NULL,
       P_ACCESORIOS     IN CLOB       DEFAULT NULL
)
IS
BEGIN
   MSG:= 'NO SE PUDO GUARDAR FACTURA';
   SELECT COUNT(CODVENTA) INTO CONTADOR
   FROM VENTAS WHERE FACTURA = P_FACTURA AND CODVENTA <>
P_CODVENTA;
   IF CONTADOR > 0 THEN
     MSG := 'YA EXISTE UNA FACTURA CON ESTE NUMERO, INGRESE OTRO
NUMERO DE FACTURA';
     RAISE EXP_ERROR;
   END IF;
   UPDATE VENTAS
```

```
   SET    codcliente     = p_codcliente,
          codusuario     = p_codusuario,
          codbanco       = p_codbanco,
          codvehiculo    = p_codvehiculo,
          coddispositivo = p_coddispositivo,
          codseguro      = p_codseguro,
          factura        = p_factura,
          precio         = p_precio,
          entrada        = p_entrada,
          plazo          = p_plazo,
          gastos         = p_gastos,
          fechafac       = p_fechafac,
          fechaent       = p_fechaent,
          ramv           = p_ramv,
          motor          = p_motor,
          chasis         = p_chasis,
          color          = p_color,
          anio           = p_anio,
          accesorios     = p_accesorios
    where CODVENTA       = P_CODVENTA;
   COMMIT;
   EXCEPTION
   WHEN EXP_ERROR THEN
      RAISE_APPLICATION_ERROR(-20000, MSG);
      ROLLBACK;
   WHEN OTHERS THEN
      RAISE_APPLICATION_ERROR (-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO' || SQLERRM);
      ROLLBACK;
END SP_GUARDAR;
```

--PROCEDURE SP_ELIMINAR

*/*Los datos no se eliminan de la base de datos mas bien se activan e inactivan. Cada tabla tiene su campo activo =0 e inactivo =1.*

*/*La información que se elimina se va a una papelera de reciclaje y luego si desea restaurarlo también lo podrá hacer.*/*

```
PROCEDURE SP_ELIMINAR
(
       P_CODVENTA     IN NUMBER     DEFAULT NULL
)
IS
BEGIN
   MSG:= 'NO SE PUDO ELIMINAR FACTURA';
   DELETE FROM VENTAS WHERE CODVENTA = P_CODVENTA;
   COMMIT;
   EXCEPTION
   WHEN EXP_ERROR THEN
      RAISE_APPLICATION_ERROR (-20000, MSG);
      ROLLBACK;
   WHEN OTHERS THEN
      RAISE_APPLICATION_ERROR (-20000, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO');
      ROLLBACK;
END SP_ELIMINAR;
```

--PROCEDURE SP_CONSULTAR

--Consultar por id

/ Permite realizar consultas a las ventas realizadas y almacenadas en la base de datos.*/*

```
PROCEDURE SP_CONSULTAR
(
        P_CODVENTA       IN NUMBER      DEFAULT NULL,
        MY_CURSOR        IN OUT         T_CURSOR
)
IS
BEGIN
    OPEN MY_CURSOR FOR
    SELECT * FROM VENTAS WHERE CODVENTA = P_CODVENTA;
END SP_CONSULTAR;

--PROCEDURE SP_CONSULTARTODOS_FORMS
--Consultar creditos por fecha y vendedor
/* Permite realizar una consulta por todos los parámetros del form ventas.*/

PROCEDURE SP_CONSULTARTODOS_FORMS
(
        --Para filtrar por vendedor
        P_CODVENDEDOR     IN NUMBER      DEFAULT NULL,
        P_FECINI          IN DATE        DEFAULT NULL,
        P_FECFIN          IN DATE        DEFAULT NULL,

        --Indica si se consulta por fechas
        P_OPFEC           IN CHAR        DEFAULT NULL,
        DATA_VENTAS       IN OUT         T_VENTAS
)
IS
        --Sin vendedor con fechas
        CURSOR CUR_01 IS
        SELECT *
        FROM   VISCONSULVENTAS_CLIENTES A
        WHERE  A.FECHAFAC BETWEEN P_FECINI AND P_FECFIN;

        --Sin vendedor sin fechas
        CURSOR CUR_02 IS
        SELECT *
        FROM   VISCONSULVENTAS_CLIENTES A;

        --Con vendedor con fechas
        CURSOR CUR_03 IS
        SELECT *
        FROM   VISCONSULVENTAS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        AND    A.FECHAFAC BETWEEN P_FECINI AND P_FECFIN;

        --Con vendedor sin fechas
        CURSOR CUR_04 IS
        SELECT *
        FROM   VISCONSULVENTAS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR;
BEGIN
    CONTADOR := 0;
    SELECT CARGO INTO V_CARGO FROM USUARIOS
    WHERE CODUSUARIO = P_CODVENDEDOR;
    IF V_CARGO = 'Administrador' THEN
      IF P_OPFEC = 'S' THEN

         --Sin vendedor con fechas
```

```
        FOR FILA IN CUR_01 LOOP
           CONTADOR := CONTADOR + 1;
           DATA_VENTAS(CONTADOR) := FILA;
        END LOOP;
        IF CUR_01%ISOPEN THEN
          CLOSE CUR_01;
        END IF;
      ELSE

        --Sin vendedor sin fechas
        FOR FILA IN CUR_02 LOOP
           CONTADOR := CONTADOR + 1;
           DATA_VENTAS(CONTADOR) := FILA;
        END LOOP;
        IF CUR_02%ISOPEN THEN
          CLOSE CUR_02;
        END IF;
      END IF;
    ELSE
      IF P_OPFEC = 'S' THEN

        --Con vendedor con fechas
        FOR FILA IN CUR_03 LOOP
           CONTADOR := CONTADOR + 1;
           DATA_VENTAS(CONTADOR) := FILA;
        END LOOP;
        IF CUR_03%ISOPEN THEN
          CLOSE CUR_03;
        END IF;
      ELSE

        --Con vendedor sin fechas
        FOR FILA IN CUR_04 LOOP
           CONTADOR := CONTADOR + 1;
           DATA_VENTAS(CONTADOR) := FILA;
        END LOOP;
        IF CUR_04%ISOPEN THEN
          CLOSE CUR_04;
        END IF;
      END IF;
    END IF;
END SP_CONSULTARTODOS_FORMS;

--PROCEDURE SP_CONSULTARPORVENDEDOR_FORMS
-- Consulta por el código de un vendedor determinado

PROCEDURE SP_CONSULTARPORVENDEDOR_FORMS
(
       --Para filtrar por vendedor
       P_CODVENDEDOR    IN NUMBER     DEFAULT NULL,
       P_FECINI         IN DATE       DEFAULT NULL,
       P_FECFIN         IN DATE       DEFAULT NULL,

       --Indica si se consulta por fechas
       P_OPFEC          IN CHAR       DEFAULT NULL,
       DATA_VENTAS      IN OUT        T_VENTAS
)
IS
     --Con vendedor con fechas
```

```
        CURSOR CUR_03 IS
        SELECT *
        FROM   VISCONSULVENTAS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        AND    A.FECHAFAC BETWEEN P_FECINI AND P_FECFIN;

        --Con vendedor sin fechas
        CURSOR CUR_04 IS
        SELECT *
        FROM   VISCONSULVENTAS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR;
BEGIN
    CONTADOR := 0;
    SELECT CARGO INTO V_CARGO FROM USUARIOS
    WHERE CODUSUARIO = P_CODVENDEDOR;
    IF P_OPFEC = 'S' THEN

       --Con vendedor con fechas
       FOR FILA IN CUR_03 LOOP
          CONTADOR := CONTADOR + 1;
          DATA_VENTAS(CONTADOR) := FILA;
       END LOOP;
       IF CUR_03%ISOPEN THEN
         CLOSE CUR_03;
       END IF;
    ELSE

       --Con vendedor sin fechas
       FOR FILA IN CUR_04 LOOP
          CONTADOR := CONTADOR + 1;
          DATA_VENTAS(CONTADOR) :=  FILA;
       END LOOP;
       IF CUR_04%ISOPEN THEN
         CLOSE CUR_04;
       END IF;
    END IF;
END SP_CONSULTARPORVENDEDOR_FORMS;


-- PROCEDURE SP_CONSULTARPORBANCO_FORMS
  /*Permite consultar por un banco determinado y filtra la información por vendedor y por
fecha*/

PROCEDURE SP_CONSULTARPORBANCO_FORMS
(
        P_CODBANCO       IN NUMBER      DEFAULT NULL,

        --Para filtrar por vendedor
        P_CODVENDEDOR    IN NUMBER      DEFAULT NULL,
        P_FECINI         IN DATE        DEFAULT NULL,
        P_FECFIN         IN DATE        DEFAULT NULL,

        --Indica si se consulta por fechas
        P_OPFEC          IN CHAR        DEFAULT NULL,
        DATA_VENTAS      IN OUT         T_VENTAS
)
IS

        --Sin vendedor con fechas
```

```
        CURSOR CUR_01 IS
        SELECT *
        FROM   VISCONSULVENTAS_CLIENTES A
        WHERE  A.FECHAFAC BETWEEN P_FECINI AND P_FECFIN
        AND    A.Codbanco = P_CODBANCO;

        --Sin vendedor sin fechas
        CURSOR CUR_02 IS
        SELECT *
        FROM   VISCONSULVENTAS_CLIENTES A
        WHERE  A.Codbanco = P_CODBANCO;

        --Con vendedor con fechas
        CURSOR CUR_03 IS
        SELECT *
        FROM   VISCONSULVENTAS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        AND    A.FECHAFAC BETWEEN P_FECINI AND P_FECFIN
        AND    A.Codbanco = P_CODBANCO;

        --Con vendedor sin fechas
        CURSOR CUR_04 IS
        SELECT *
        FROM   VISCONSULVENTAS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        AND    A.Codbanco = P_CODBANCO;
BEGIN
    CONTADOR := 0;
    SELECT CARGO INTO V_CARGO FROM USUARIOS
    WHERE CODUSUARIO = P_CODVENDEDOR;
    IF V_CARGO = 'Administrador' THEN
      IF P_OPFEC = 'S' THEN

        --Sin vendedor con fechas
        FOR FILA IN CUR_01 LOOP
           CONTADOR := CONTADOR + 1;
           DATA_VENTAS(CONTADOR) := FILA;
        END LOOP;
        IF CUR_01%ISOPEN THEN
          CLOSE CUR_01;
        END IF;
      ELSE

        --Sin vendedor sin fechas
        FOR FILA IN CUR_02 LOOP
           CONTADOR := CONTADOR + 1;
           DATA_VENTAS(CONTADOR) := FILA;
        END LOOP;
        IF CUR_02%ISOPEN THEN
          CLOSE CUR_02;
        END IF;
      END IF;
    ELSE
      IF P_OPFEC = 'S' THEN

        --Con vendedor con fechas
        FOR FILA IN CUR_03 LOOP
           CONTADOR := CONTADOR + 1;
           DATA_VENTAS(CONTADOR) := FILA;
```

```
        END LOOP;
        IF CUR_03%ISOPEN THEN
          CLOSE CUR_03;
        END IF;
      ELSE

        --Con vendedor sin fechas
        FOR FILA IN CUR_04 LOOP
          CONTADOR := CONTADOR + 1;
          DATA_VENTAS(CONTADOR) := FILA;
        END LOOP;
        IF CUR_04%ISOPEN THEN
          CLOSE CUR_04;
        END IF;
      END IF;
    END IF;
END SP_CONSULTARPORBANCO_FORMS;

--PROCEDURE SP_CONSULTARPORMODELO_FORMS
-- Consulta por el modelo de un vehículo específico.

PROCEDURE SP_CONSULTARPORMODELO_FORMS
(
      P_CODVEHICULO    IN NUMBER     DEFAULT NULL,

      --Para filtrar por vendedor
      P_CODVENDEDOR    IN NUMBER     DEFAULT NULL,
      P_FECINI         IN DATE       DEFAULT NULL,
      P_FECFIN         IN DATE       DEFAULT NULL,

      --Indica si se consulta por fechas
      P_OPFEC          IN CHAR       DEFAULT NULL,
      DATA_VENTAS      IN OUT        T_VENTAS
)
IS
      --Sin vendedor con fechas
      CURSOR CUR_01 IS
      SELECT *
      FROM   VISCONSULVENTAS_CLIENTES A
      WHERE  A.FECHAFAC BETWEEN P_FECINI AND P_FECFIN
      AND    A.Codvehiculo = P_CODVEHICULO;

      --Sin vendedor sin fechas
      CURSOR CUR_02 IS
      SELECT *
      FROM   VISCONSULVENTAS_CLIENTES A
      WHERE  A.Codvehiculo = P_CODVEHICULO;

      --Con vendedor con fechas
      CURSOR CUR_03 IS
      SELECT *
      FROM   VISCONSULVENTAS_CLIENTES A
      WHERE  A.CODUSUARIO = P_CODVENDEDOR
      AND    A.FECHAFAC BETWEEN P_FECINI AND P_FECFIN
      AND    A.Codvehiculo = P_CODVEHICULO;

      --Con vendedor sin fechas
      CURSOR CUR_04 IS
      SELECT *
```

```
        FROM   VISCONSULVENTAS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        AND    A.Codvehiculo = P_CODVEHICULO;

BEGIN
    CONTADOR := 0;
    SELECT CARGO INTO V_CARGO FROM USUARIOS
    WHERE CODUSUARIO = P_CODVENDEDOR;
    IF V_CARGO = 'Administrador' THEN
      IF P_OPFEC = 'S' THEN

        --Sin vendedor con fechas
        FOR FILA IN CUR_01 LOOP
           CONTADOR := CONTADOR + 1;
           DATA_VENTAS(CONTADOR) := FILA;
        END LOOP;
        IF CUR_01%ISOPEN THEN
          CLOSE CUR_01;
        END IF;
      ELSE

        --Sin vendedor sin fechas
        FOR FILA IN CUR_02 LOOP
           CONTADOR := CONTADOR + 1;
           DATA_VENTAS(CONTADOR) := FILA;
        END LOOP;
        IF CUR_02%ISOPEN THEN
          CLOSE CUR_02;
        END IF;
      END IF;
    ELSE
      IF P_OPFEC = 'S' THEN

        --Con vendedor con fechas
        FOR FILA IN CUR_03 LOOP
           CONTADOR := CONTADOR + 1;
           DATA_VENTAS(CONTADOR) := FILA;
        END LOOP;
        IF CUR_03%ISOPEN THEN
          CLOSE CUR_03;
        END IF;
      ELSE


        --Con vendedor sin fechas
        FOR FILA IN CUR_04 LOOP
           CONTADOR := CONTADOR + 1;
           DATA_VENTAS(CONTADOR) := FILA;
        END LOOP;
        IF CUR_04%ISOPEN THEN
          CLOSE CUR_04;
        END IF;
      END IF;
    END IF;
END SP_CONSULTARPORMODELO_FORMS;
END PK_VENTAS;
/
```

❖ SP_DEALER.

Objetivo: Este procedimiento es para asignarle el nombre del concesionario donde se va a ejecutarse el sistema.

```
prompt Creating procedure SP_DEALER
prompt ============================

CREATE OR REPLACE PROCEDURE USER15.SP_DEALER
(
 P_NOMBRE IN DEALER.NOMBRE%TYPE
)
IS
BEGIN
    DELETE FROM DEALER;
    INSERT INTO DEALER (NOMBRE) VALUES (P_NOMBRE);
    COMMIT;
Exception
    WHEN OTHERS THEN
        ROLLBACK;
END SP_DEALER;
```

❖ PK_LOGIN

```
CREATE OR REPLACE PACKAGE BODY PK_LOGIN IS

 PROCEDURE SP_LOGIN
 (
        MY_CURSOR OUT T_CURSOR,
        P_USUARIO VARCHAR2,
        P_CLAVE VARCHAR2
 )
 IS
  V_DEALER      VARCHAR2(40);
 BEGIN
  SELECT NOMBRE INTO V_DEALER FROM DEALER WHERE ROWNUM=1;

  OPEN MY_CURSOR FOR
  SELECT CODUSUARIO, USUARIO, CLAVE, NOMBRE, CARGO, MAIL, V_DEALER
AS DEALER
  FROM USUARIOS WHERE ACTIVO=1 AND upper(USUARIO) = upper(P_USUARIO)
AND CLAVE = pk_utilitarios.encrypt(LOWER(P_CLAVE));

 EXCEPTION
  WHEN NO_DATA_FOUND THEN
      RAISE_APPLICATION_ERROR(-20000, 'NO SE ENCUENTRA EL
CONCESIONARIO CONFIGURADO');
  WHEN TOO_MANY_ROWS THEN
      RAISE_APPLICATION_ERROR(-20000,'HAY MAS DE UN CONCESIONARIO
CONFIGURADO');

 END SP_LOGIN;

 --RECIBE USUARIO Y CLAVE
```

```
  --VALIDA LOGIN MEDIANTE CLAVE ENCRIPTADA CON pk_utilitarios.encrypt()
  --DEVUELVE NOMBRE DEL USUARIO, NOMBRE DEL CONCESIONARIO Y ID DEL
USUARIO
 PROCEDURE SP_LOGIN_FORMS
 (
    P_USUARIO       IN  VARCHAR2,
    P_CLAVE         IN  VARCHAR2,
    P_NOMBRE        OUT VARCHAR2,
    P_CARGO         OUT VARCHAR2,
    P_ID            OUT NUMBER,
    P_DEALER        OUT VARCHAR2
 )
 IS
 BEGIN
    SELECT CODUSUARIO, NOMBRE, CARGO INTO P_ID, P_NOMBRE, P_CARGO
    FROM   USUARIOS WHERE UPPER(USUARIO) = UPPER(P_USUARIO) AND
CLAVE = pk_utilitarios.encrypt(LOWER(P_CLAVE))
    AND    ACTIVO = 1;

    SELECT NOMBRE INTO P_DEALER FROM DEALER WHERE ROWNUM=1;

 EXCEPTION
  WHEN NO_DATA_FOUND THEN
     RAISE_APPLICATION_ERROR(-20000, 'USUARIO O PASSWORD
INCORRECTOS, O NO HAY CONCESIONARIO CONFIGURADO');
  WHEN TOO_MANY_ROWS THEN
     RAISE_APPLICATION_ERROR(-20000,'HAY MAS DE UN USUARIO
CONFIGURADO CON EL MISMO USUARIO O CONTRASEÑA');

 END SP_LOGIN_FORMS;

END PK_LOGIN;
```

❖ PK_UTILITARIOS

```
CREATE OR REPLACE PACKAGE BODY PK_UTILITARIOS AS

 -- key must be exactly 8 bytes long
 c_encrypt_key varchar2(8) := 'key45678';

 MSG             VARCHAR2(100);
 EXP_ERROR       EXCEPTION;

 FUNCTION ENCRYPT (i_password varchar2)
 RETURN VARCHAR2 IS
  v_encrypted_val varchar2(38);
  v_data         varchar2(38);
 BEGIN
   -- Input data must have a length divisible by eight
   v_data := RPAD(i_password,(TRUNC(LENGTH(i_password)/8)+1)*8,CHR(0));

   DBMS_OBFUSCATION_TOOLKIT.DESENCRYPT(
     input_string     => v_data,
     key_string       => c_encrypt_key,
     encrypted_string => v_encrypted_val);
```

```
  RETURN v_encrypted_val;
END ENCRYPT;

FUNCTION DECRYPT (i_password varchar2)
RETURN VARCHAR2 IS
 v_decrypted_val varchar2(38);
BEGIN
  DBMS_OBFUSCATION_TOOLKIT.DESDECRYPT(
    input_string     => i_password,
    key_string       => c_encrypt_key,
    decrypted_string => v_decrypted_val);
  RETURN v_decrypted_val;
END DECRYPT;

--PROCEDIMIENTO PARA MANDAR MAILS
PROCEDURE ENVIAR_MAIL_CREDITO_APROBADO
(
  P_IDCREDITO     IN NUMBER
)
IS
  V_SERVER        VARCHAR2(64) := '192.168.1.6';
  V_DE            VARCHAR2(64) := 'info_creditos@digisoft.com';
  V_PARA          VARCHAR2(64);
  MAIL_CONN       UTL_SMTP.connection;
  V_CUERPO        VARCHAR2(700);
  V_DEALER        VARCHAR2(40);
  crlf            CONSTANT VARCHAR2(2) := CHR(13) || CHR(10);

  CURSOR CUR_CREDITO IS
  SELECT A.Estado, A.VENDEDOR, A.CLIENTE, A.BANCO, A.MODELO, A.PRECIO,
A.ENTRADA, A.PLAZO, B.CORREO, C.MAIL, C.TELEFONO
  FROM   visconsulcreditos_clientes A INNER JOIN CLIENTES B
  ON     A.CodCliente = B.CODCLIENTE
  INNER JOIN USUARIOS C
  ON     A.CodUsuario = C.CODUSUARIO
  WHERE  A.CodCredito = P_IDCREDITO;

  DATOS_MAIL      CUR_CREDITO%ROWTYPE;

BEGIN
  --LLENO LAS VARIABLES NECESARIAS
  FOR FILA IN CUR_CREDITO LOOP
    DATOS_MAIL := FILA;
  END LOOP;

  SELECT NOMBRE INTO V_DEALER FROM DEALER WHERE ROWNUM=1;

  IF DATOS_MAIL.CORREO IS NULL THEN
    MSG := 'EL CLIENTE NO POSEE CORREO';
    RAISE EXP_ERROR;
  END IF;

  IF LENGTH(DATOS_MAIL.CORREO) = 0 THEN
    MSG := 'EL CLIENTE NO POSEE CORREO';
    RAISE EXP_ERROR;
  END IF;

  IF DATOS_MAIL.ESTADO != 'Aprobado' THEN
    MSG := 'EL CREDITO NO ESTÁ APROBADO, TAL VEZ ALGUIEN MAS CAMBIO SU
```

```
ESTADO';
      RAISE EXP_ERROR;
    END IF;

    V_PARA := DATOS_MAIL.CORREO;

    --ESCRIBO EL CUERPO DEL MENSAJE
    V_CUERPO := 'Estimado(a) Cliente ' || DATOS_MAIL.CLIENTE || crlf || crlf;
    V_CUERPO := V_CUERPO || 'La Presente es para comunicarle que ud. ha sido APROBADO(A)
para un crédito bajo las siguientes características' || crlf || crlf;
    V_CUERPO := V_CUERPO || ' + Banco: ' || DATOS_MAIL.Banco || crlf;
    V_CUERPO := V_CUERPO || ' + Vehículo Modelo: ' || DATOS_MAIL.MODELO || crlf;
    V_CUERPO := V_CUERPO || ' + Precio: ' || DATOS_MAIL.Precio || crlf;
    V_CUERPO := V_CUERPO || ' + Plazo de: ' || DATOS_MAIL.PLAZO || ' Meses' || crlf || crlf;
    V_CUERPO := V_CUERPO || 'Para más información comuníquese con su Asesor(a) Comercial: ' ||
crlf;
    V_CUERPO := V_CUERPO || DATOS_MAIL.VENDEDOR || crlf;
    V_CUERPO := V_CUERPO || 'Mail: ' || DATOS_MAIL.MAIL || crlf;
    V_CUERPO := V_CUERPO || 'Telf.: ' || DATOS_MAIL.TELEFONO || crlf || crlf;
    V_CUERPO := V_CUERPO || 'Atte.' || crlf;
    V_CUERPO := V_CUERPO || V_DEALER;

    --ABRO LA CONEXION
    MAIL_CONN := UTL_SMTP.open_connection(V_SERVER, 25);
    --ASIGNO ATRIBUTOS AL MAIL
    UTL_SMTP.helo(MAIL_CONN, V_SERVER);
    UTL_SMTP.mail(MAIL_CONN, V_DE);
    UTL_SMTP.rcpt(MAIL_CONN, V_PARA);
    UTL_SMTP.open_data(MAIL_CONN);

    --ASIGNO CUERPO AL MAIL
    UTL_SMTP.write_data(MAIL_CONN, V_CUERPO);

    --CIERRO LA CONEXION
    UTL_SMTP.close_data(MAIL_CONN);
    UTL_SMTP.quit(MAIL_CONN);


    EXCEPTION
      WHEN EXP_ERROR THEN
         RAISE_APPLICATION_ERROR(-20002, MSG);
      WHEN OTHERS THEN
         RAISE_APPLICATION_ERROR(-20002, 'UN ERROR HA OCURRIDO EN EL
PROCEDIMIENTO: ' || SQLERRM);

  END ENVIAR_MAIL_CREDITO_APROBADO;

  --FUNCION VALIDACION DE CEDULA
  FUNCTION VALIDARCEDULA (V_CEDULA IN VARCHAR2)
  RETURN BOOLEAN
  IS
    RESULT BOOLEAN;
    CONT   NUMBER;
    SUMA   NUMBER := 0;
    DECENA NUMBER := 10;
    VALOR  NUMBER;
  BEGIN
     --PARA CUANDO EL CLIENTE NO CONOCE LA CEDULA O NO LA TIENE DISPONIBLE SE
DA ESTA OPCION
```

```
    IF V_CEDULA = '9999999999' THEN
      RETURN TRUE;
    END IF;

    --SI ES QUE ESTO DA ERROR ES PORQUE HAY LETRAS Y PUEDE SER UN PASAPORTE
    --POR ENDE SE PUEDE CAER BAJO ERROR ORA-01722 EL CUAL CONTROLA LA
EXCEPCION Y LA FUNCION RETORNA TRUE
    VALOR := TO_NUMBER(V_CEDULA);

    IF LENGTH(V_CEDULA) <> 10 THEN
      RETURN FALSE;
    END IF;

    FOR CONT IN 1..9 LOOP
       VALOR := TO_NUMBER(Substr(V_CEDULA, CONT, 1));

       --SI EL DIGITO ESTA EN LA POSICION IMPAR
       IF Mod(CONT, 2) = 1 THEN
         VALOR := VALOR * 2;
         IF VALOR > 9 THEN
           VALOR := VALOR - 9;
         END IF;
       END IF;
       SUMA := SUMA + VALOR;
    END LOOP;

    WHILE DECENA < SUMA LOOP
        DECENA := DECENA + 10;
    END LOOP;

    VALOR := TO_NUMBER(SUBSTR(V_CEDULA, 10, 1));

    IF DECENA - SUMA = VALOR THEN
      RESULT := TRUE;
    ELSE
      RESULT := FALSE;
    END IF;

    RETURN RESULT;

    EXCEPTION
      WHEN OTHERS THEN
         IF SQLCODE = -6502 THEN
           RETURN TRUE;
         ELSE
           RETURN FALSE;
         END IF;

 END VALIDARCEDULA;

 --VALIDACION DEL MAIL
 FUNCTION VALIDARMAIL (V_MAIL   IN VARCHAR2)
 RETURN BOOLEAN
 IS
  RESULT    BOOLEAN := TRUE;
  HAY_ARROBA NUMBER  := 0;
  HAY_PUNTO  NUMBER  := 0;
  CONT       NUMBER;
  VAL_ASCII  NUMBER;
```

```
  CAR       CHAR(1);

 BEGIN
  IF NVL(V_MAIL,'0') = '0' THEN
    RETURN FALSE;
  END IF;

  HAY_PUNTO  := Instr(V_MAIL, '.', 1, 1);
  HAY_ARROBA := Instr(V_MAIL, '@', 1, 1);

  IF HAY_PUNTO = 0 OR HAY_ARROBA = 0 THEN
    RETURN FALSE;
  END IF;

  FOR CONT IN 1..LENGTH(V_MAIL) LOOP
     CAR := Substr(V_MAIL, CONT, 1);
     VAL_ASCII := Ascii(CAR);

     IF CONT = 1 THEN
       IF VAL_ASCII >= 48 AND VAL_ASCII <= 57 THEN
         RESULT := FALSE;
       END IF;
     END IF;

     IF VAL_ASCII >= 0 AND VAL_ASCII <= 45 THEN
       RESULT := FALSE;
     END IF;

     IF VAL_ASCII IN(47) OR VAL_ASCII >= 123 THEN
       RESULT := FALSE;
     END IF;

     IF VAL_ASCII >= 58 AND VAL_ASCII <= 63 THEN
       RESULT := FALSE;
     END IF;

     IF VAL_ASCII >= 91 AND VAL_ASCII <= 96 THEN
       RESULT := FALSE;
     END IF;
  END LOOP;

  RETURN RESULT;

 END VALIDARMAIL;

END PK_UTILITARIOS;
```

❖ PK_REPORTES

```
CREATE OR REPLACE Package Body PK_REPORTES IS

    CONTADOR        NUMBER;
    V_CARGO         VARCHAR2(20);

--CARGA LOS DATOS DE LOS REPORTES, USANDO SP'S
```

```
--LLENA LOS DATOS OBTENIDOS POR LOS SP'S EN SUS RESPECTIVAS TABLAS
TEMPORALES

--REPORTES DE CLIENTES

  PROCEDURE SP_REP_CLIENTES_MIXTOS
  (
    --SIRVE PARA LAS CONSULTAS POR CEDULA, CLIENTE O FECHAS
    P_CEDULA        IN VARCHAR2  DEFAULT NULL,
    P_NOMBRES       IN VARCHAR2  DEFAULT NULL,
    P_APELLIDOS     IN VARCHAR2  DEFAULT NULL,
    --PARA FECHAS
    P_FECINI_IN     IN VARCHAR2  DEFAULT NULL,
    P_FECFIN_IN     IN VARCHAR2  DEFAULT NULL,
    --TIPO DE CONSULTA C=CEDULA, A=APELLIDOS, N=NOMBRES, F=FECHAS
    P_OP            IN CHAR      DEFAULT NULL,
    --SI INCLUYE VENDEDOR EL ID DEL MISMO CASO CONTRARIO P_VENDEDOR = -1 O
SEA ADMINISTRADOR QUE PUEDE CONSULTAR A TODOS
    P_VENDEDOR      IN NUMBER    DEFAULT NULL
  )
  IS
   ES_VENDEDOR      BOOLEAN     := FALSE;
   P_FECINI         DATE        := TO_DATE(P_FECINI_IN,'DD/MM/YYYY');
   P_FECFIN         DATE        := TO_DATE(P_FECFIN_IN,'DD/MM/YYYY');

  BEGIN
     SELECT COUNT(CODUSUARIO) INTO CONTADOR
     FROM USUARIOS WHERE CARGO ='Vendedor' AND CODUSUARIO = P_VENDEDOR;

     IF CONTADOR > 0 THEN
        ES_VENDEDOR := TRUE;
     END IF;

     DELETE FROM tmp_rep_clientes;

     IF ES_VENDEDOR = FALSE THEN
     --ADMINISTRADOR
       --TODOS
       IF P_OP = 'T' THEN
         INSERT INTO tmp_rep_clientes
         SELECT * FROM VISCONSULCLIENTES
         WHERE  ACTIVO = 1 ORDER BY APELLIDOS, NOMBRES;
       END IF;
       --CEDULA
       IF P_OP = 'C' THEN
         INSERT INTO tmp_rep_clientes
         SELECT * FROM VISCONSULCLIENTES
         WHERE  ACTIVO = 1 AND CEDULA = P_CEDULA;
       END IF;
       --APELLIDOS
       IF P_OP = 'A' THEN
         INSERT INTO tmp_rep_clientes
         SELECT * FROM VISCONSULCLIENTES
         WHERE  ACTIVO = 1 AND UPPER(APELLIDOS) LIKE '%' || UPPER(P_APELLIDOS) ||
'%'
         ORDER  BY APELLIDOS;
       END IF;
       --NOMBRES
       IF P_OP = 'N' THEN
```

```
          INSERT INTO tmp_rep_clientes
          SELECT * FROM VISCONSULCLIENTES
          WHERE  ACTIVO = 1 AND UPPER(NOMBRES) LIKE '%' || UPPER(P_NOMBRES) ||
'%'
          ORDER  BY NOMBRES;
        END IF;
        --FECHAS
        IF P_OP = 'F' THEN
          INSERT INTO tmp_rep_clientes
          SELECT * FROM VISCONSULCLIENTES
          WHERE  ACTIVO = 1 AND FECHAATENCION BETWEEN P_FECINI AND
P_FECFIN
          ORDER  BY FECHAATENCION, APELLIDOS;
        END IF;
      ELSE
      --VENDEDOR
        --TODOS
        IF P_OP = 'T' THEN
          INSERT INTO tmp_rep_clientes
          SELECT * FROM VISCONSULCLIENTES
          WHERE  ACTIVO = 1 AND CODVENDEDOR = P_VENDEDOR ORDER BY
APELLIDOS, NOMBRES;
        END IF;
        --CEDULA
        IF P_OP = 'C' THEN
          INSERT INTO tmp_rep_clientes
          SELECT * FROM VISCONSULCLIENTES
          WHERE  ACTIVO = 1 AND CEDULA = P_CEDULA AND CODVENDEDOR =
P_VENDEDOR;
        END IF;
        --APELLIDOS
        IF P_OP = 'A' THEN
          INSERT INTO tmp_rep_clientes
          SELECT * FROM VISCONSULCLIENTES
          WHERE  ACTIVO = 1 AND UPPER(APELLIDOS) LIKE '%' || UPPER(P_APELLIDOS) ||
'%' AND CODVENDEDOR = P_VENDEDOR
          ORDER  BY APELLIDOS;
        END IF;
        --NOMBRES
        IF P_OP = 'N' THEN
          INSERT INTO tmp_rep_clientes
          SELECT * FROM VISCONSULCLIENTES
          WHERE  ACTIVO = 1 AND UPPER(NOMBRES) LIKE '%' || UPPER(P_NOMBRES) ||
'%' AND CODVENDEDOR = P_VENDEDOR
          ORDER  BY NOMBRES;
        END IF;
        --FECHAS
        IF P_OP = 'F' THEN
          INSERT INTO tmp_rep_clientes
          SELECT * FROM VISCONSULCLIENTES
          WHERE  ACTIVO = 1 AND FECHAATENCION BETWEEN P_FECINI AND
P_FECFIN AND CODVENDEDOR = P_VENDEDOR
          ORDER  BY FECHAATENCION, APELLIDOS;
        END IF;
      END IF;

      COMMIT;

      EXCEPTION
```

```
        WHEN OTHERS THEN
            RAISE_APPLICATION_ERROR(-20000, 'ERROR AL GENERAR REPORTE DE
CLIENTES, OPCION: '
            || P_OP || ' ' || SQLERRM);
            ROLLBACK;

  END SP_REP_CLIENTES_MIXTOS;

  PROCEDURE SP_OCULTAR_TELEFONOS
  (
    P_VENDEDOR      IN NUMBER    DEFAULT NULL
  )
  IS
  BEGIN
     UPDATE TMP_REP_CLIENTES SET
          TELCASA = 'NO DISP.',
          TELTRABAJO = 'NO DISP.',
          CELULAR = 'NO DISP.'
     WHERE  CODVENDEDOR != P_VENDEDOR;

     COMMIT;

  END SP_OCULTAR_TELEFONOS;

  PROCEDURE SP_REP_CLIENTES_GENERAL
  (
    --SIRVE PARA LAS CONSULTAS POR CEDULA, CLIENTE O FECHAS
    P_CEDULA        IN VARCHAR2  DEFAULT NULL,
    P_NOMBRES       IN VARCHAR2  DEFAULT NULL,
    P_APELLIDOS     IN VARCHAR2  DEFAULT NULL,
    P_MODELO        IN VARCHAR2  DEFAULT NULL,
    --PARA FECHAS
    P_FECINI_IN     IN VARCHAR2  DEFAULT NULL,
    P_FECFIN_IN     IN VARCHAR2  DEFAULT NULL,
    --TIPO DE CONSULTA C=CEDULA, A=APELLIDOS, N=NOMBRES, F=FECHAS,
MCF=MODELO CON FECHAS, MSF= MODELO SIN FECHAS
    P_OP            IN VARCHAR2  DEFAULT NULL,
    --SI INCLUYE VENDEDOR EL ID DEL MISMO CASO CONTRARIO P_VENDEDOR = -1 O
SEA ADMINISTRADOR QUE PUEDE CONSULTAR A TODOS
    P_VENDEDOR      IN NUMBER    DEFAULT NULL
  )
  IS
   ES_VENDEDOR      BOOLEAN     := FALSE;
   P_FECINI         DATE        := TO_DATE(P_FECINI_IN,'DD/MM/YYYY');
   P_FECFIN         DATE        := TO_DATE(P_FECFIN_IN,'DD/MM/YYYY');

  BEGIN
    SELECT COUNT(CODUSUARIO) INTO CONTADOR
    FROM USUARIOS WHERE CARGO ='Vendedor' AND CODUSUARIO = P_VENDEDOR;

    IF CONTADOR > 0 THEN
       ES_VENDEDOR := TRUE;
    END IF;

    DELETE FROM TMP_REP_CLIENTES;

    --ADMINISTRADOR
    --TODOS
    IF P_OP = 'T' THEN
```

```
      INSERT INTO TMP_REP_CLIENTES
      SELECT *
      FROM   VISCONSULCLIENTES
      WHERE  ACTIVO = 1 ORDER BY APELLIDOS, NOMBRES;

      COMMIT;

      IF ES_VENDEDOR = TRUE THEN
        SP_OCULTAR_TELEFONOS(P_VENDEDOR);
      END IF;
    END IF;
    --CEDULA
    IF P_OP = 'C' THEN
      INSERT INTO TMP_REP_CLIENTES
      SELECT *
      FROM   VISCONSULCLIENTES
      WHERE  ACTIVO = 1 AND CEDULA = P_CEDULA;

      COMMIT;

      IF ES_VENDEDOR = TRUE THEN
        SP_OCULTAR_TELEFONOS(P_VENDEDOR);
      END IF;
    END IF;
    --APELLIDOS
    IF P_OP = 'A' THEN
      INSERT INTO TMP_REP_CLIENTES
      SELECT *
      FROM   VISCONSULCLIENTES
      WHERE  ACTIVO = 1 AND UPPER(APELLIDOS) LIKE '%' || UPPER(P_APELLIDOS) ||
'%'
      ORDER  BY APELLIDOS;

      COMMIT;

      IF ES_VENDEDOR = TRUE THEN
        SP_OCULTAR_TELEFONOS(P_VENDEDOR);
      END IF;
    END IF;
    --NOMBRES
    IF P_OP = 'N' THEN
      INSERT INTO TMP_REP_CLIENTES
      SELECT *
      FROM   VISCONSULCLIENTES
      WHERE  ACTIVO = 1 AND UPPER(NOMBRES) LIKE '%' || UPPER(P_NOMBRES) || '%'
      ORDER  BY NOMBRES;

      COMMIT;

      IF ES_VENDEDOR = TRUE THEN
        SP_OCULTAR_TELEFONOS(P_VENDEDOR);
      END IF;
    END IF;
    --FECHAS
    IF P_OP = 'F' THEN
      INSERT INTO TMP_REP_CLIENTES
      SELECT *
      FROM   VISCONSULCLIENTES
      WHERE  ACTIVO = 1 AND FECHAATENCION BETWEEN P_FECINI AND P_FECFIN
```

```
      ORDER  BY FECHAATENCION, APELLIDOS;

      COMMIT;

      IF ES_VENDEDOR = TRUE THEN
        SP_OCULTAR_TELEFONOS(P_VENDEDOR);
      END IF;
    END IF;

    --MODELO DE VEHICULO
    IF P_OP = 'MCF' THEN
      INSERT INTO TMP_REP_CLIENTES
      SELECT *
      FROM   VISCONSULCLIENTES
      WHERE  ACTIVO = 1 AND FECHAATENCION BETWEEN P_FECINI AND P_FECFIN
      AND    MODELO = P_MODELO
      ORDER  BY FECHAATENCION, APELLIDOS;

      COMMIT;

      IF ES_VENDEDOR = TRUE THEN
        SP_OCULTAR_TELEFONOS(P_VENDEDOR);
      END IF;
    END IF;

    IF P_OP = 'MSF' THEN
      INSERT INTO TMP_REP_CLIENTES
      SELECT *
      FROM   VISCONSULCLIENTES
      WHERE  ACTIVO = 1
      AND    MODELO = P_MODELO
      ORDER  BY FECHAATENCION, APELLIDOS;

      COMMIT;

      IF ES_VENDEDOR = TRUE THEN
        SP_OCULTAR_TELEFONOS(P_VENDEDOR);
      END IF;
    END IF;

    EXCEPTION
       WHEN OTHERS THEN
          RAISE_APPLICATION_ERROR(-20000, 'ERROR AL GENERAR REPORTE DE
CLIENTES, OPCION: '
          || P_OP || ' ' || SQLERRM);
          ROLLBACK;

  END SP_REP_CLIENTES_GENERAL;

--REPORTES DE CREDITOS

  PROCEDURE SP_REP_CREDITOS_TODOS
  (
    --PARA FILTRAR POR VENDEDOR
    P_CODVENDEDOR    IN NUMBER     DEFAULT NULL,
    P_FECINI         IN DATE       DEFAULT NULL,
    P_FECFIN         IN DATE       DEFAULT NULL,
    --INDICA SI SE CONSULTA POR FECHAS
    P_OPFEC          IN CHAR       DEFAULT NULL
```

```
)

IS

BEGIN

    SELECT CARGO INTO V_CARGO FROM USUARIOS
    WHERE CODUSUARIO = P_CODVENDEDOR;

    DELETE FROM TMP_REP_CREDITOS;

    IF V_CARGO = 'Administrador' THEN
      IF P_OPFEC = 'S' THEN
        --SIN VENDEDOR CON FECHAS
        INSERT INTO TMP_REP_CREDITOS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.FECHAING BETWEEN P_FECINI AND P_FECFIN
        ORDER  BY A.Cliente;

      ELSE
        --SIN VENDEDOR SIN FECHAS
        INSERT INTO TMP_REP_CREDITOS
        SELECT  *
        FROM    VISCONSULCREDITOS_CLIENTES A
        ORDER BY A.Cliente;

      END IF;
    ELSE
      IF P_OPFEC = 'S' THEN
        --CON VENDEDOR CON FECHAS
        INSERT INTO TMP_REP_CREDITOS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        AND    A.FECHAING BETWEEN P_FECINI AND P_FECFIN
        ORDER BY A.Cliente;

      ELSE
        --CON VENDEDOR SIN FECHAS
        INSERT INTO TMP_REP_CREDITOS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        ORDER BY A.Cliente;

      END IF;
    END IF;

    COMMIT;

    EXCEPTION
      WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20000, 'ERROR AL GENERAR REPORTE DE
CREDITOS, OPCION: '
        || P_OPFEC || ' ' || SQLERRM);
        ROLLBACK;

END SP_REP_CREDITOS_TODOS;
```

```
PROCEDURE SP_REP_CREDITOS_VENDEDOR
(
   --PARA FILTRAR POR VENDEDOR
   P_CODVENDEDOR    IN NUMBER     DEFAULT NULL,
   P_FECINI         IN DATE       DEFAULT NULL,
   P_FECFIN         IN DATE       DEFAULT NULL,
   --INDICA SI SE CONSULTA POR FECHAS
   P_OPFEC          IN CHAR       DEFAULT NULL
)
IS

BEGIN
   DELETE FROM TMP_REP_CREDITOS;

   IF P_OPFEC = 'S' THEN
     --CON VENDEDOR CON FECHAS
     INSERT INTO TMP_REP_CREDITOS
     SELECT *
     FROM   VISCONSULCREDITOS_CLIENTES A
     WHERE  A.CODUSUARIO = P_CODVENDEDOR
     AND    A.FECHAING BETWEEN P_FECINI AND P_FECFIN
     ORDER BY A.Cliente;

   ELSE
     --CON VENDEDOR SIN FECHAS
     INSERT INTO TMP_REP_CREDITOS
     SELECT *
     FROM   VISCONSULCREDITOS_CLIENTES A
     WHERE  A.CODUSUARIO = P_CODVENDEDOR
     ORDER BY A.Cliente;
   END IF;

   COMMIT;

   EXCEPTION
      WHEN OTHERS THEN
         RAISE_APPLICATION_ERROR(-20000, 'ERROR AL GENERAR REPORTE DE
CREDITOS POR VENDEDOR, OPCION: '
         || P_OPFEC || ' ' || SQLERRM);
         ROLLBACK;

END SP_REP_CREDITOS_VENDEDOR;

PROCEDURE SP_REP_CREDITOS_BANCOS
(
   P_CODBANCO       IN NUMBER     DEFAULT NULL,
   --PARA FILTRAR POR VENDEDOR
   P_CODVENDEDOR    IN NUMBER     DEFAULT NULL,
   P_FECINI         IN DATE       DEFAULT NULL,
   P_FECFIN         IN DATE       DEFAULT NULL,
   --INDICA SI SE CONSULTA POR FECHAS
   P_OPFEC          IN CHAR       DEFAULT NULL
)

IS

BEGIN
```

```
      SELECT CARGO INTO V_CARGO FROM USUARIOS
      WHERE CODUSUARIO = P_CODVENDEDOR;

      DELETE FROM TMP_REP_CREDITOS;

      IF V_CARGO = 'Administrador' THEN
        IF P_OPFEC = 'S' THEN
          --SIN VENDEDOR CON FECHAS
          INSERT INTO TMP_REP_CREDITOS
          SELECT *
          FROM   VISCONSULCREDITOS_CLIENTES A
          WHERE  A.FECHAING BETWEEN P_FECINI AND P_FECFIN
          AND    A.CODBANCO = P_CODBANCO
          ORDER  BY A.Cliente;
        ELSE
          --SIN VENDEDOR SIN FECHAS
          INSERT INTO TMP_REP_CREDITOS
          SELECT  *
          FROM    VISCONSULCREDITOS_CLIENTES A
          WHERE   A.CODBANCO = P_CODBANCO
          ORDER BY A.Cliente;
        END IF;
      ELSE
        IF P_OPFEC = 'S' THEN
          --CON VENDEDOR CON FECHAS
          INSERT INTO TMP_REP_CREDITOS
          SELECT *
          FROM   VISCONSULCREDITOS_CLIENTES A
          WHERE  A.CODUSUARIO = P_CODVENDEDOR
          AND    A.FECHAING BETWEEN P_FECINI AND P_FECFIN
          AND    A.CODBANCO = P_CODBANCO
          ORDER BY A.Cliente;
        ELSE
          --CON VENDEDOR SIN FECHAS
          INSERT INTO TMP_REP_CREDITOS
          SELECT *
          FROM   VISCONSULCREDITOS_CLIENTES A
          WHERE  A.CODUSUARIO = P_CODVENDEDOR
          AND    A.CODBANCO = P_CODBANCO
          ORDER BY A.Cliente;
        END IF;
      END IF;

      COMMIT;

      EXCEPTION
        WHEN OTHERS THEN
           RAISE_APPLICATION_ERROR(-20000, 'ERROR AL GENERAR REPORTE DE
CREDITOS POR BANCO, OPCION: '
           || P_OPFEC || ' ' || SQLERRM);
           ROLLBACK;

  END SP_REP_CREDITOS_BANCOS;

  PROCEDURE SP_REP_CREDITOS_ESTADOS
  (
     P_ESTADO        IN VARCHAR2    DEFAULT NULL,
     --PARA FILTRAR POR VENDEDOR
     P_CODVENDEDOR    IN NUMBER      DEFAULT NULL,
```

```
    P_FECINI        IN DATE       DEFAULT NULL,
    P_FECFIN        IN DATE       DEFAULT NULL,
    --INDICA SI SE CONSULTA POR FECHAS
    P_OPFEC         IN CHAR       DEFAULT NULL
)

IS
BEGIN

    SELECT CARGO INTO V_CARGO FROM USUARIOS
    WHERE CODUSUARIO = P_CODVENDEDOR;

    DELETE FROM TMP_REP_CREDITOS;

    IF V_CARGO = 'Administrador' THEN
      IF P_OPFEC = 'S' THEN
        --SIN VENDEDOR CON FECHAS
        INSERT INTO TMP_REP_CREDITOS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.FECHAING BETWEEN P_FECINI AND P_FECFIN
        AND    A.Estado = P_ESTADO
        ORDER  BY A.Cliente;
      ELSE
        --SIN VENDEDOR SIN FECHAS
        INSERT INTO TMP_REP_CREDITOS
        SELECT  *
        FROM    VISCONSULCREDITOS_CLIENTES A
        WHERE   A.Estado = P_ESTADO
        ORDER BY A.Cliente;
      END IF;
    ELSE
      IF P_OPFEC = 'S' THEN
        --CON VENDEDOR CON FECHAS
        INSERT INTO TMP_REP_CREDITOS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        AND    A.FECHAING BETWEEN P_FECINI AND P_FECFIN
        AND    A.Estado = P_ESTADO
        ORDER BY A.Cliente;
      ELSE
        --CON VENDEDOR SIN FECHAS
        INSERT INTO TMP_REP_CREDITOS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        AND    A.Estado = P_ESTADO
        ORDER BY A.Cliente;
      END IF;
    END IF;

    COMMIT;

    EXCEPTION
      WHEN OTHERS THEN
         RAISE_APPLICATION_ERROR(-20000, 'ERROR AL GENERAR REPORTE DE
CREDITOS POR ESTADO, OPCION: '
         || P_OPFEC || ' ' || SQLERRM);
```

```
        ROLLBACK;

END SP_REP_CREDITOS_ESTADOS;

PROCEDURE SP_REP_CREDITOS_MODELO
(
   P_CODVEHICULO    IN NUMBER     DEFAULT NULL,
   --PARA FILTRAR POR VENDEDOR
   P_CODVENDEDOR    IN NUMBER     DEFAULT NULL,
   P_FECINI         IN DATE       DEFAULT NULL,
   P_FECFIN         IN DATE       DEFAULT NULL,
   --INDICA SI SE CONSULTA POR FECHAS
   P_OPFEC          IN CHAR       DEFAULT NULL
)

IS
BEGIN

    SELECT CARGO INTO V_CARGO FROM USUARIOS
    WHERE CODUSUARIO = P_CODVENDEDOR;

    DELETE FROM TMP_REP_CREDITOS;

    IF V_CARGO = 'Administrador' THEN
      IF P_OPFEC = 'S' THEN
        --SIN VENDEDOR CON FECHAS
        INSERT INTO TMP_REP_CREDITOS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.FECHAING BETWEEN P_FECINI AND P_FECFIN
        AND    A.CodVehiculo = P_CODVEHICULO
        ORDER  BY A.Cliente;
      ELSE
        --SIN VENDEDOR SIN FECHAS
        INSERT INTO TMP_REP_CREDITOS
        SELECT  *
        FROM    VISCONSULCREDITOS_CLIENTES A
        WHERE   A.CodVehiculo = P_CODVEHICULO
        ORDER BY A.Cliente;
      END IF;
    ELSE
      IF P_OPFEC = 'S' THEN
        --CON VENDEDOR CON FECHAS
        INSERT INTO TMP_REP_CREDITOS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        AND    A.FECHAING BETWEEN P_FECINI AND P_FECFIN
        AND    A.CodVehiculo = P_CODVEHICULO
        ORDER BY A.Cliente;
      ELSE
        --CON VENDEDOR SIN FECHAS
        INSERT INTO TMP_REP_CREDITOS
        SELECT *
        FROM   VISCONSULCREDITOS_CLIENTES A
        WHERE  A.CODUSUARIO = P_CODVENDEDOR
        AND    A.CodVehiculo = P_CODVEHICULO
        ORDER BY A.Cliente;
      END IF;
```

```
      END IF;

      COMMIT;

      EXCEPTION
        WHEN OTHERS THEN
          RAISE_APPLICATION_ERROR(-20000, 'ERROR AL GENERAR REPORTE DE
CREDITOS POR MODELO, OPCION: '
          || P_OPFEC || ' ' || SQLERRM);
          ROLLBACK;

  END SP_REP_CREDITOS_MODELO;

--REPORTES DE VENTAS

  PROCEDURE SP_REP_VENTAS_TODOS
  (
    --PARA FILTRAR POR VENDEDOR
    P_CODVENDEDOR    IN NUMBER     DEFAULT NULL,
    P_FECINI         IN DATE       DEFAULT NULL,
    P_FECFIN         IN DATE       DEFAULT NULL,
    --INDICA SI SE CONSULTA POR FECHAS
    P_OPFEC          IN CHAR       DEFAULT NULL
  )

  IS

  BEGIN

      SELECT CARGO INTO V_CARGO FROM USUARIOS
      WHERE CODUSUARIO = P_CODVENDEDOR;

      DELETE FROM TMP_REP_VENTAS;

      IF V_CARGO = 'Administrador' THEN
        IF P_OPFEC = 'S' THEN
          --SIN VENDEDOR CON FECHAS
          INSERT INTO TMP_REP_VENTAS
          SELECT *
          FROM   VISCONSULVENTAS_CLIENTES A
          WHERE  A.FECHAFAC BETWEEN P_FECINI AND P_FECFIN
          ORDER  BY A.Cliente;

        ELSE
          --SIN VENDEDOR SIN FECHAS
          INSERT INTO TMP_REP_VENTAS
          SELECT  *
          FROM    VISCONSULVENTAS_CLIENTES A
          ORDER BY A.Cliente;

        END IF;
      ELSE
        IF P_OPFEC = 'S' THEN
          --CON VENDEDOR CON FECHAS
          INSERT INTO TMP_REP_VENTAS
          SELECT *
          FROM   VISCONSULVENTAS_CLIENTES A
          WHERE  A.CODUSUARIO = P_CODVENDEDOR
          AND    A.FECHAFAC BETWEEN P_FECINI AND P_FECFIN
```

```
          ORDER BY A.Cliente;

        ELSE
          --CON VENDEDOR SIN FECHAS
          INSERT INTO TMP_REP_VENTAS
          SELECT *
          FROM   VISCONSULVENTAS_CLIENTES A
          WHERE  A.CODUSUARIO = P_CODVENDEDOR
          ORDER BY A.Cliente;

        END IF;
      END IF;

      COMMIT;

      EXCEPTION
        WHEN OTHERS THEN
          RAISE_APPLICATION_ERROR(-20000, 'ERROR AL GENERAR REPORTE DE
VENTAS, OPCION: '
          || P_OPFEC || ' ' || SQLERRM);
          ROLLBACK;

  END SP_REP_VENTAS_TODOS;

  PROCEDURE SP_REP_VENTAS_VENDEDOR
  (
    --PARA FILTRAR POR VENDEDOR
    P_CODVENDEDOR    IN NUMBER     DEFAULT NULL,
    P_FECINI         IN DATE       DEFAULT NULL,
    P_FECFIN         IN DATE       DEFAULT NULL,
    --INDICA SI SE CONSULTA POR FECHAS
    P_OPFEC          IN CHAR       DEFAULT NULL
  )
  IS

  BEGIN
    DELETE FROM TMP_REP_VENTAS;

    IF P_OPFEC = 'S' THEN
      --CON VENDEDOR CON FECHAS
      INSERT INTO TMP_REP_VENTAS
      SELECT *
      FROM   VISCONSULVENTAS_CLIENTES A
      WHERE  A.CODUSUARIO = P_CODVENDEDOR
      AND    A.FECHAFAC BETWEEN P_FECINI AND P_FECFIN
      ORDER BY A.Cliente;

    ELSE
      --CON VENDEDOR SIN FECHAS
      INSERT INTO TMP_REP_VENTAS
      SELECT *
      FROM   VISCONSULVENTAS_CLIENTES A
      WHERE  A.CODUSUARIO = P_CODVENDEDOR
      ORDER BY A.Cliente;
    END IF;

    COMMIT;

    EXCEPTION
```

```
        WHEN OTHERS THEN
            RAISE_APPLICATION_ERROR(-20000, 'ERROR AL GENERAR REPORTE DE
VENTAS POR VENDEDOR, OPCION: '
            || P_OPFEC || ' ' || SQLERRM);
            ROLLBACK;

  END SP_REP_VENTAS_VENDEDOR;

  PROCEDURE SP_REP_VENTAS_BANCOS
  (
     P_CODBANCO       IN NUMBER     DEFAULT NULL,
     --PARA FILTRAR POR VENDEDOR
     P_CODVENDEDOR    IN NUMBER     DEFAULT NULL,
     P_FECINI         IN DATE       DEFAULT NULL,
     P_FECFIN         IN DATE       DEFAULT NULL,
     --INDICA SI SE CONSULTA POR FECHAS
     P_OPFEC          IN CHAR       DEFAULT NULL
  )

  IS

  BEGIN
      SELECT CARGO INTO V_CARGO FROM USUARIOS
      WHERE CODUSUARIO = P_CODVENDEDOR;

      DELETE FROM TMP_REP_VENTAS;

      IF V_CARGO = 'Administrador' THEN
        IF P_OPFEC = 'S' THEN
          --SIN VENDEDOR CON FECHAS
          INSERT INTO TMP_REP_VENTAS
          SELECT *
          FROM   VISCONSULVENTAS_CLIENTES A
          WHERE  A.FECHAFAC BETWEEN P_FECINI AND P_FECFIN
          AND    A.CODBANCO = P_CODBANCO
          ORDER  BY A.Cliente;
        ELSE
          --SIN VENDEDOR SIN FECHAS
          INSERT INTO TMP_REP_VENTAS
          SELECT  *
          FROM    VISCONSULVENTAS_CLIENTES A
          WHERE   A.CODBANCO = P_CODBANCO
          ORDER BY A.Cliente;
        END IF;
      ELSE
        IF P_OPFEC = 'S' THEN
          --CON VENDEDOR CON FECHAS
          INSERT INTO TMP_REP_VENTAS
          SELECT *
          FROM   VISCONSULVENTAS_CLIENTES A
          WHERE  A.CODUSUARIO = P_CODVENDEDOR
          AND    A.FECHAFAC BETWEEN P_FECINI AND P_FECFIN
          AND    A.CODBANCO = P_CODBANCO
          ORDER BY A.Cliente;
        ELSE
          --CON VENDEDOR SIN FECHAS
          INSERT INTO TMP_REP_VENTAS
          SELECT *
          FROM   VISCONSULVENTAS_CLIENTES A
```

```
            WHERE  A.CODUSUARIO = P_CODVENDEDOR
            AND    A.CODBANCO = P_CODBANCO
            ORDER BY A.Cliente;
          END IF;
        END IF;

        COMMIT;

        EXCEPTION
          WHEN OTHERS THEN
            RAISE_APPLICATION_ERROR(-20000, 'ERROR AL GENERAR REPORTE DE
VENTAS POR BANCO, OPCION: '
            || P_OPFEC || ' ' || SQLERRM);
            ROLLBACK;

  END SP_REP_VENTAS_BANCOS;

  PROCEDURE SP_REP_VENTAS_MODELO
  (
     P_CODVEHICULO    IN NUMBER     DEFAULT NULL,
     --PARA FILTRAR POR VENDEDOR
     P_CODVENDEDOR    IN NUMBER     DEFAULT NULL,
     P_FECINI         IN DATE       DEFAULT NULL,
     P_FECFIN         IN DATE       DEFAULT NULL,
     --INDICA SI SE CONSULTA POR FECHAS
     P_OPFEC          IN CHAR       DEFAULT NULL
  )

  IS
  BEGIN
      SELECT CARGO INTO V_CARGO FROM USUARIOS
      WHERE CODUSUARIO = P_CODVENDEDOR;

      DELETE FROM TMP_REP_VENTAS;

      IF V_CARGO = 'Administrador' THEN
        IF P_OPFEC = 'S' THEN
          --SIN VENDEDOR CON FECHAS
          INSERT INTO TMP_REP_VENTAS
          SELECT *
          FROM   VISCONSULVENTAS_CLIENTES A
          WHERE  A.FECHAFAC BETWEEN P_FECINI AND P_FECFIN
          AND    A.CodVehiculo = P_CODVEHICULO
          ORDER  BY A.Cliente;
        ELSE
          --SIN VENDEDOR SIN FECHAS
          INSERT INTO TMP_REP_VENTAS
          SELECT  *
          FROM    VISCONSULVENTAS_CLIENTES A
          WHERE   A.CodVehiculo = P_CODVEHICULO
          ORDER BY A.Cliente;
        END IF;
      ELSE
        IF P_OPFEC = 'S' THEN
          --CON VENDEDOR CON FECHAS
          INSERT INTO TMP_REP_VENTAS
          SELECT *
          FROM   VISCONSULVENTAS_CLIENTES A
          WHERE  A.CODUSUARIO = P_CODVENDEDOR
```

```
            AND    A.FECHAFAC BETWEEN P_FECINI AND P_FECFIN
            AND    A.CodVehiculo = P_CODVEHICULO
            ORDER BY A.Cliente;
          ELSE
            --CON VENDEDOR SIN FECHAS
            INSERT INTO TMP_REP_VENTAS
            SELECT *
            FROM   VISCONSULVENTAS_CLIENTES A
            WHERE  A.CODUSUARIO = P_CODVENDEDOR
            AND    A.CodVehiculo = P_CODVEHICULO
            ORDER BY A.Cliente;
          END IF;
        END IF;

        COMMIT;

        EXCEPTION
          WHEN OTHERS THEN
            RAISE_APPLICATION_ERROR(-20000, 'ERROR AL GENERAR REPORTE DE
VENTAS POR MODELO, OPCION: '
            || P_OPFEC || ' ' || SQLERRM);
            ROLLBACK;

  END SP_REP_VENTAS_MODELO;

  PROCEDURE SP_REP_AGENDA_FECHAS
  (
        P_OPCION          IN CHAR       DEFAULT NULL,
        --S=SI INCLUYE FECHAS, N=NO INCLUYE FECHAS
        P_FECINI          IN DATE       DEFAULT NULL,
        P_FECFIN          IN DATE       DEFAULT NULL,
        P_CODUSUARIO      IN NUMBER     DEFAULT NULL
  )

  IS

  BEGIN
     DELETE FROM TMP_REP_AGENDA;

     IF P_OPCION = 'S' THEN
       INSERT INTO TMP_REP_AGENDA
       SELECT * FROM VisConsulAgenda
       WHERE  codusuario = P_CODUSUARIO
       AND    Fecha BETWEEN P_FECINI AND P_FECFIN;
     ELSE
       INSERT INTO TMP_REP_AGENDA
       SELECT * FROM VisConsulAgenda
       WHERE  codusuario = P_CODUSUARIO;
     END IF;

     COMMIT;

     EXCEPTION
       WHEN OTHERS THEN
         RAISE_APPLICATION_ERROR(-20000, 'ERROR AL GENERAR REPORTE DE
AGENDA, OPCION: '
         || P_OPCION || ' ' || SQLERRM);
         ROLLBACK;
```

```
END SP_REP_AGENDA_FECHAS;

END PK_REPORTES;
```

PRESUPUESTO

RECURSOS HUMANOS

Desarrolladores

- Bogdan Gabriel Babici
- Doris Tierra
- María del Carmen Achig C

Director del Seminario

- Ing. Raúl Tingo

Coordinador del Seminario

- Ing. Miguel Quiroz M

Docentes y Tutores de Tesis

- Ing. Lili Santos
- Ing. Ricardo Naranjo
- Ing. Raúl Álvarez
- Ing. Jaime Lozada
- Ing. Alice Naranjo

Gráfico # 23

Detalle del Presupuesto del Proyecto Terrasoft

1.PERSONAL			
Investigadores del Proyecto	Cantidad	Salario (USD)	CostoTotal (USD)
Desarrolladores	3	$ 3.000	$ 9.000
Subtotal 1:		$ 9.000	

2. EQUIPOS COMPUTACIONALES			
Equipos	Cantidad	Costo Unitario	Costo Total
Computador portátil	3	$ 1.200	$ 3.600
Impresora	1	$ 150	$ 150
Subtotal 2:		$ 3.750	
3.UTILES DE OFICINA			
Materiales	Cantidad	Costo Unitario	Costo Total
Resma de papel	5	$ 5	$ 25
Cartucho de tinta negra	2	$ 8	$ 16
Cartucho de tinta color	3	$ 10	$ 30
Subtotal 3:		$ 71	
4.Varios			
Items	Días	Costo Unitario(USD)	Costo Total (USD)
Transporte	25	$5	$ 125
Alimentación	10	$2,50	$ 25
Internet	200	$0,80	$ 160
Subtotal 4:		$ 310	
Costo total previo a imprevistos		**$ 13.131**	
Imprevistos considerados en un 10%		**$ 1.313,1**	
Costo total final (USD)		**$ 14.444,1**	

Elaborado: Achig María, Babici Bogdan, Tierra Doris
Fuente: Proformas de Servicios

El acumulado total resultó $13.131 pero considerando un 10% de imprevistos se define $14.441 como el valor total de este proyecto.

MANUAL DE INSTALACION DEL SISTEMA

REQUISTOS DE SOFTWARE

SERVIDOR

Para que el Sistema Terrasoft pueda ser implementado debemos tener un Servidor con los Siguientes Programas Instalados:

- Base de Datos Oracle 9i, 10G o superior.
- El Application Server – Form Services para Oracle 10G que incluya el Servicio OC4J el mismo lo podemos instalar desde el paquete de instalación de Oracle Developer Suite 10G.
- El Report Server de Oracle Reports 10G el cual también lo podemos encontrar en Oracle Developer Suite 10G.
- .Net Framework 2.0 para ejecutar un Servicio de Traspaso de Solicitudes en Excel.
- Contar con un Servidor de Correo para el envío por Mails de Solicitudes.

USUARIOS

Para que el Sistema funcione en los Terminales de los Usuarios estos deberán tener instalados lo siguiente:

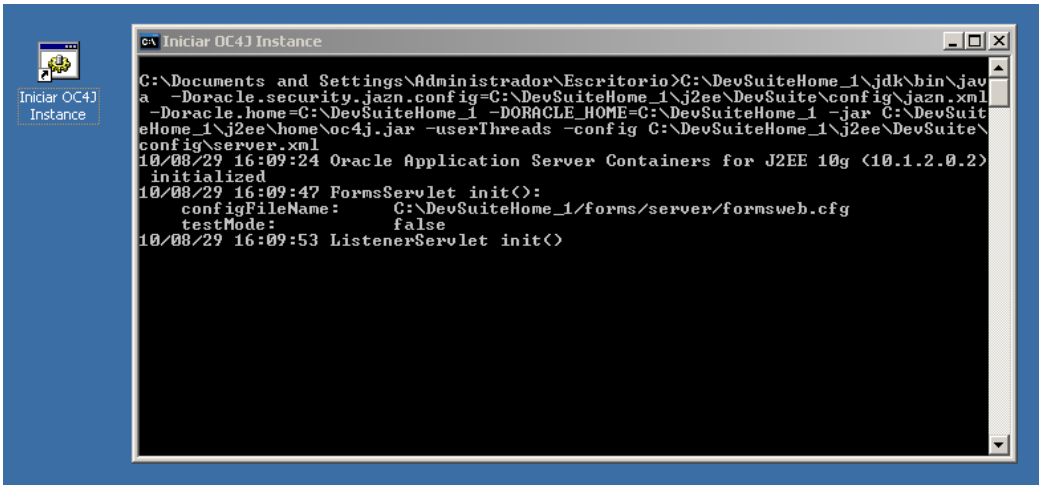
- Un explorador Web de preferencia **Mozilla Firefox 2.0** o superior.
- El **JInitiator** el cual se puede bajar al instante de conectarse por primera vez al Sistema, esto lo hace automáticamente el Application Server.
- Microsoft Excel 2000 o superior para Visualizar las Solicitudes en Excel generadas por el Sistema.

COMO INICIAR EL SISTEMA

Para que el Sistema pueda ser comenzado a usar debemos realizar lo siguiente dentro de nuestro servidor:

Iniciar el Servicio de la Base de Datos

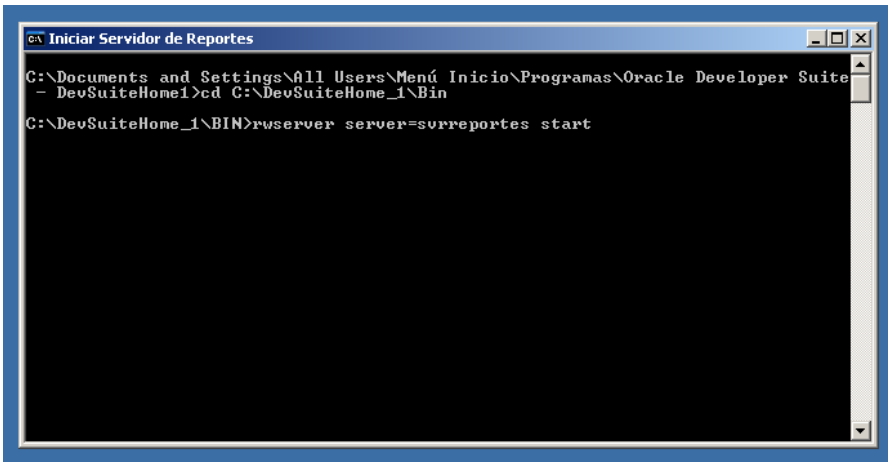
Iniciar el Servicio OC4J.- Esto lo podemos hacer mediante un Acceso directo a que nos brinda el mismo Instalador del Oracle Developer Suite 10G.



Iniciar el Servidor de Reportes.- Para esto SOLO la primera vez, debemos crear un Servidor de Reportes con el Nombre “svrreportes” esto lo hacemos desde una Línea de Comandos en el CMD en donde digitamos la siguiente instrucción: ***rwserver –install svrreportes autostart=yes***

A partir de las Próximas ocasiones ya no es necesario instalarlo sino solamente abrir el servicio con estas Líneas de Comando: ***rwserver server=svrreportes start***

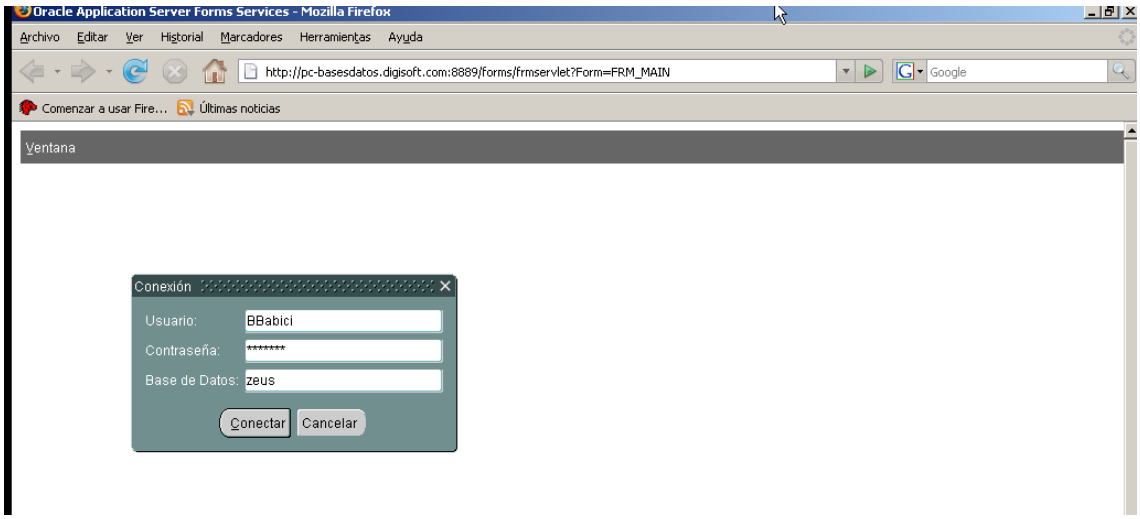
En nuestro caso esta última línea la hemos colocado en un Archivo BAT para no escribirla siempre.



COMO INGRESAR AL SISTEMA

Para ingresar al Sistema damos click en la Página de Acceso Directo de la Aplicación Web

Al hacer esto nos llevará la Página de Inicio de Sesión.



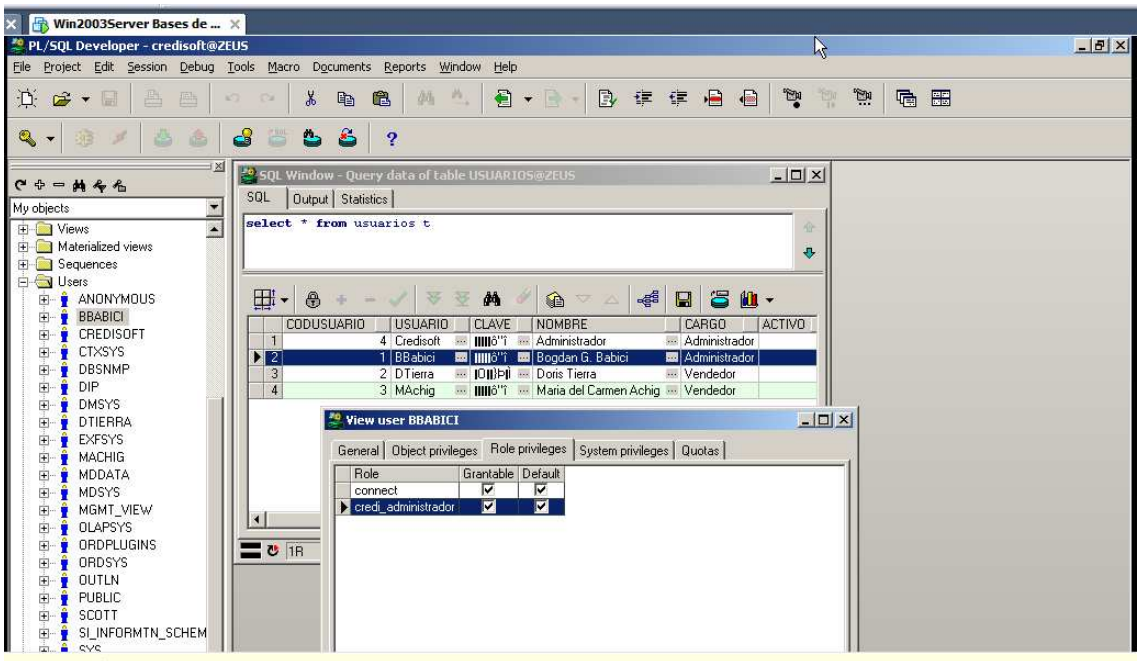
Damos Click en el Botón Conectar y se abrirá la Pantalla Principal del Sistema.

CONSIDERACIONES

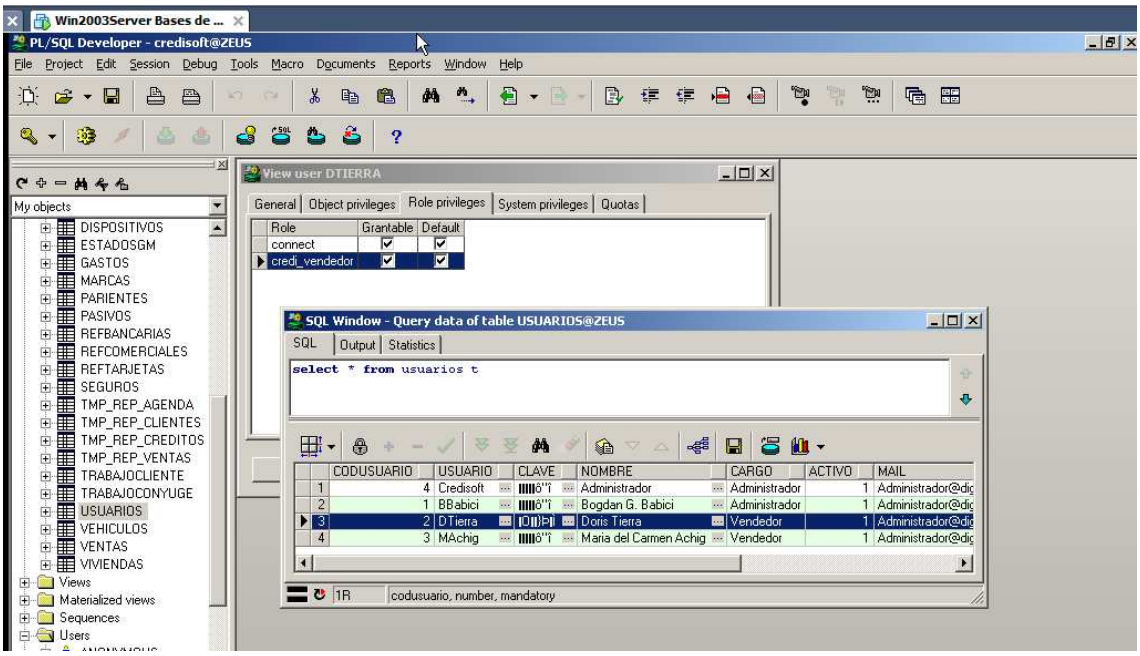
No hay que olvidar que el Usuario con el que nos conectamos a la Aplicación debe ser un usuario ORACLE válido, y que el mismo debe estar registrado en la Tabla "Usuarios" de nuestro Sistema, con el mismo usuario y clave con el que está registrado en Oracle.

Así mismo si el Usuario dentro de la Tabla "Usuarios" está registrado con Cargo de Vendedor, entonces en la Base de Datos el Usuario Oracle deberá tener el Rol **credi_vendedor**, y si está registrado como Administrador deberá tener el Rol **credi_administrador**, tal como vemos en la siguiente imagen.

Aparte todos los usuarios deben de tener el Rol propio de Oracle "**connect**"



Usuario Administrador



Usuario Vendedor

Bibliografía

AMAT, JOAN Ma. **El Control de Gestión: Una perspectiva de Dirección**. Ediciones Gestión 2000 S.A. Barcelona. España. 1992.

FRED R., DAVID (1994**). La Gerencia Estratégica**. Bogotá. Editorial. Fondo Editorial Legis.

JOFREE (2000), **Computación Global los resultados obtenidos de la aplicación del Cuadro de Mando Integral**, Venezuela.

KAPLAN, R y NORTON, D. (2000), **Cuadro de Mando Integral**. Ediciones Gestión 2000. S.A. Barcelona. España.

MARTINEZ, R. (2001), **Cuadro de Mando Integral: "Nuevo Modelo para el Diseño de indicadores y Control de Gestión"**. VII Congreso Nacional de Control Interno en las Entidades del Estado. Santa Fe. Colombia.

MESA (2000), **"Desarrollo formal de la Estrategia del Departamento de la Calidad de General Motors Venezolana (GMV) utilizando el Cuadro de Mando Integral y su impacto en la estructura organizacional de ese Departamento"**, Trabajo presentado para la obtención del grado de Especialista en Operaciones. IESA. Caracas.

MORA ANTONIO (2000). **Nuevas Herramientas de Gestión Pública: El Cuadro de Mando Integral.** 1era Edición. Editora Gestión 2000. España.

QUIROGA (2000), **"Cuadro de Mando Integral aplicado al Individuo".** Universidad del CEMA.

ORTIZ R. (2004), **"Gestión por Indicadores para el Departamento de Sistemas dentro de la Perspectiva de Cuadro de Mando Integral"**, Trabajo de Grado presentado ante el Decanato de Investigación y Postgrado de la Universidad Fermin para optar al grado de Magíster Scientiarum en Gerencia Empresarial. Barquisimeto (Venezuela).

TAMAYO, M. (2000), **El Proceso de Investigación Científica**. Editores Noriega. México.

UNIVERSIDAD POLITÉCNICA SALESIANA, **Instructivo para la Elaboración y Presentación Trabajos de Grado de Tesis.**

ALLAN FREEDMAN, **Diccionario de Computación.**

REFERENCIAS WEB

http://www.alegsa.com.ar/Dic/vmware%20workstation.php

http://web.jet.es/amozarrain/Gestion_procesos.htm

http://www.usmp.edu.pe/publicaciones/boletin/fia/info7/bpwin.htm

http://xue.unalmed.edu.co/~mfcabrera/db/arqoracle.pdf

http://www.infor.uva.es/~jvegas/cursos/bd/oraseg/oraseg.html

http://www.materiabiz.com/mbz/estrategiaymarketing/nota.vsp?nid=32013

http://www.sirac.info/Curtiembres/html/indicadores.asp

http://docs.kde.org/stable/es/kdesdk/umbrello/uml-elements.html

http://es.wikipedia.org/wiki/M%C3%A1quina_virtual

http://www.ingenierosoftware.com/analisisydiseno/uml.php

http://www.devjoker.com/contenidos/catss/22/Introducci%c3%b3n-a-PLSQL.aspx

http://www.monografias.com/trabajos25/oracle/oracle.shtml

http://www.oracle.com/technology/products/forms/index.html

http://es.wikipedia.org/wiki/PL/SQL

http://es.wikipedia.org/wiki/Procedimiento_almacenado

http://www.devjoker.com/contenidos/Tutorial-PLSQL/33/Estructuras-de-control-en-PLSQL.aspx