PRÁCTICA EMPRESARIAL EN LA EMPRESA DE TELECOMUNICACIONES DE BUCARAMANGA E.S.P. - TELEBUCARAMANGA

MARITZA BENAVIDES CÉSPEDES

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2005

PRÁCTICA EMPRESARIAL EN LA EMPRESA DE TELECOMUNICACIONES DE BUCARAMANGA E.S.P. - TELEBUCARAMANGA

MARITZA BENAVIDES CÉSPEDES

Trabajo de grado para obtener el título de Ingeniera de Sistemas

Tutor Empresarial: Humberto Rueda Rivero
Profesional Master de Sistemas, Subgerencia de Informática y Tecnología
TELEBUCARAMANGA

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2005

DEDICATORIA

Quiero dedicar este trabajo de grado a todas aquellas personas que permitieron que yo estuviera en el lugar adecuado y en el momento adecuado.

A ***DIOS PADRE*** por iluminar todos los días mi camino.

A mi ***mamá*** por escucharme y a mi ***papá*** por su apoyo y ayuda.

A mi hermana ***Jasmilly*** por poder entregar mi hoja de vida en el último minuto (sin eso no hubiese llegado acá); a ***Jackeline*** y a mi ***Santy*** por estar ahí.

Al ***Flaco*** por ser la persona que más problemas me ha dado en la vida y por eso trabajaba más de la cuenta.

Pero en especial quiero dedicarme este trabajo porque por fin terminé un proyecto y esta vez sin aburrirme a la mitad del camino.

AGRADECIMIENTOS

Quiero agradecer por su apoyo y ayuda a:

El Ingeniero ***Humberto Rueda Rivero***, profesional master de sistemas de la Subgerencia de Informática y Tecnología de TELEBUCARAMANGA y tutor empresarial, por el gran apoyo para iniciar y desarrollar de la mejor forma esta práctica; por todas las cosas que me enseñó tanto a nivel profesional como a nivel personal; por su constante preocupación.

A la Señora ***Ángela Arévalo***, quien pertenecía al área de nómina y al Ingeniero ***Sergio Oswaldo Cajias Lizcano***, Subgerente de Informática y Tecnología de TELEBUCARAMANGA, por darme la oportunidad de ocupar este lugar.

A todos los ***integrantes de la Subgerencia de Informática y Tecnología***, ingenieros, secretaría y demás personas, por esos ratos de sano esparcimiento.

Al Director de Escuela de Ingeniería de Sistemas e Informática, ***Luis Ignacio González Ramírez***, por abrir el espacio para desarrollar proyectos de grado modalidad práctica empresarial.

A mi compañero de trabajo, ***Rodrigo Eduardo***, porque, aunque discutimos mucho, algunos de los momentos compartidos fueron divertidos.

A los integrantes del grupo de la página yahoo, ***Desarrollo-Oracle***, por todos sus comentarios y por sacarme de tantos apuros con respecto al código del software.

CONTENIDO

pág.

LISTA DE TABLAS

pág.

LISTA DE FIGURAS

pág.

LISTA DE ANEXOS

pág.

TITULO: PRÁCTICA EMPRESARIAL EN LA EMPRESA DE TELECOMUNICACIONES DE BUCARAMANGA E.S.P. – TELEBUCARAMANGA [*]

AUTORA: MARITZA BENAVIDES CÉSPEDES [**]

Palabras claves: TELEBUCARAMANGA, SQL, Cartridge de PL/SQL, documentación de sistemas, portal web.

RESUMEN

La Empresa de Telecomunicaciones de Bucaramanga E.S.P – TELEBUCARAMANGA conociendo la importancia de contar con la documentación de sus sistemas de información actualizada ha trabajado en varias ocasiones en la búsqueda del sistema idóneo para prestar este servicio a sus usuarios. Actualmente cuenta con una documentación realizada en el año 2002 pero viendo la necesidad de actualizarla comenzó un proyecto con la colaboración de estudiantes en práctica para elaborar un completo sistema que ayude en el almacenamiento de archivos en una base de datos y los muestre de forma dinámica en la intranet empresarial.

El sistema está dividido en dos, por un lado una interfaz para el ingreso de registros a la base de datos y por otro, el comprendido en ésta práctica, un portal web desde el cual los usuarios pueden consultar la información correspondientes a la documentación de los sistemas de la Empresa. Una vez creada la base de datos que almacena archivos multimedia de documentación, se empezó a trabajar en el desarrollo del portal web, para ello se utilizaron las herramientas del Cartridge de PL/SQL en conjunto con sentencias y funciones del SQL.

El portal web está desarrollado para dos grupos de usuarios: los usuarios finales de los sistemas de información y los ingenieros desarrolladores del área de informática de la Empresa quienes adicionalmente pueden ver contenidos más especializados.
Algunas de las funcionalidades del portal web son:

- Posee un buscador avanzado que mediante restricciones de diferente tipo permiten obtener un resultado más específico.
- Cuenta con la posibilidad de ingresar al manual completo de un sistema para el caso de capacitación de nuevo personal.
- Tiene un glosario de términos de uso frecuente dentro de la Empresa.

---

[*] Trabajo de grado, modalidad práctica empresarial.

[**] Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingeniería de Sistemas e Informática. Universidad Industrial de Santander, UIS.
Tutor Empresarial: Humberto Rueda Rivero. Profesional Master de Sistemas. TELEBUCARAMANGA.

TITLE: ENTERPRISE PRACTICE IN THE BUCARAMANGA TELECOMMUNICATIONS' COMPANY - TELEBUCARAMANGA [*]

AUTHOR: MARITZA BENAVIDES CÉSPEDES [**]

Keywords: TELEBUCARAMANGA, SQL, Cartridge of PL/SQL, documentation of systems, vestibule Web.

ABSTRACT

The Bucaramanga Telecommunications' Company - TELEBUCARAMANGA knowing the importance of having the documentation of its updated information systems has worked in several occasions in the search of the suitable system to lend this service to its users. At the moment it counts on a documentation carried out in the year 2002 but seeing the necessity of update it begins a project with the collaboration of students in practice to elaborate a system complete that helps in the storage of files in a database and shows them in a dynamic way in the Enterprise intranet.

The system is divided in two, by a side an interface for the entrance of registers to the database and on the other, adopted in this practice, a web portal from which the users can consult the information corresponding to the documentation of the Enterprise's systems. Once created the database which keeps the multimedia files of documentation, one worked in the development of the web portal, for were used it the tools of the Cartridge of PL/SQL together with sentences and functions of the SQL.

The web portal is developed for two groups of users: the users final of the systems of information and the developer's engineers of the area of computer science of the Enterprise who additionally can see specialized contents.
Some of the functionalities of the web portal are:

- An advanced searcher that allows to obtain a more specific result by means of restrictions of different types.
- It has the possibility of entering to the complete manual of a system for the case of training new personal.
- It has a glossary of terms of frequent use inside the Company.

---

[*] Grade work, managerial practical modality.

[**] Physical-Mechanical Engineering's faculty.
Systems Engineering and Computer science school. UIS.
Managerial tutor: Humberto Rueda Rivero. Systems Master Professional.
TELEBUCARAMANGA.

INTRODUCCIÓN

La documentación es el conjunto de información que dice qué hacen los sistemas, cómo lo hacen y para quién lo hacen. Es el material que explica las características técnicas y de operación de un sistema, proporciona información sobre el funcionamiento de un sistema a los encargados de su mantenimiento; permite la auditoría; enseña a los usuarios finales a interactuar con el sistema y a los desarrolladores cómo modificarlo o actualizarlo.

La Empresa de Telecomunicaciones de Bucaramanga E.S.P – TELEBUCARAMANGA teniendo en cuenta la importancia de contar con la documentación de sus sistemas de información al día, ha trabajado en varias ocasiones en la búsqueda del sistema idóneo de documentación. Actualmente cuenta con un sistema realizado en el año 2002 que se encuentra tanto en archivos impresos como en la intranet empresarial pero viendo la necesidad de actualizarlo empezó un proyecto con la colaboración de estudiantes en práctica para elaborar un completo sistema de documentación que ayude en el almacenamiento de estos archivos en una base de datos y los muestre de forma dinámica a los usuarios de los sistemas.

En el desarrollo de esta práctica empresarial se participó en la creación de la interfaz de un portal web que sirve para realizar consultas sobre una base de datos de documentación (también implementada en esta práctica) que almacena archivos multimedia que documentan los sistemas.

Este libro recopila todas las actividades realizadas en los seis meses de práctica empresarial en la Empresa TELEBUCARAMANGA.

El primer capítulo presenta el trabajo realizado; describe la Empresa donde se trabajó y los objetivos y las actividades propuestas para desarrollar la práctica empresarial.

El segundo capítulo muestra la investigación realizada sobre la importancia de documentar sistemas de información, sus ventajas y problemas a los que se pueden incurrir al no poseerla de forma clara, completa y actualizada.

En el tercer apartado de este libro se describe el análisis realizado al sistema de documentación que posee actualmente la Empresa; se explica el por qué es necesario cambiarlo y cuales son las posibles soluciones que el nuevo sistema puede ofrecer. Finalmente se listan los requerimientos del sistema.

El cuarto capítulo contiene la investigación realizada previamente al desarrollo de la base de datos de documentación; la capacitación en la herramienta case utilizada para el diseño del diagrama Entidad-Relación y el modelo de datos y el análisis a los diferentes tipos de datos que almacenan archivos de gran tamaño. Se explica como quedó implementado el diagrama E-R, el modelo de datos y cuales son las tablas y vistas creadas de la nueva base de datos de documentación.

En el quinto capítulo se describe la herramienta utilizada para el desarrollo de paquetes, procedimientos y funciones que muestran a través del portal web los archivos multimedia de documentación de los sistemas de información de la Empresa. De igual forma se explica cómo quedó implementado el portal y qué funcionalidades contiene.

El último capítulo presenta las diferentes evaluaciones realizadas tanto por parte de la Universidad como por la Empresa, por medio del director encargado y del tutor empresarial, de las actividades desarrolladas.

En los anexos se incluyen los cronogramas comparativos entre las actividades realizadas y las planteadas para cada periodo de la práctica; el formato de la encuesta realizada para conocer la opinión de los ingenieros desarrolladores sobre el actual sistema de documentación; el formato utilizado en la Empresa para levantar los requerimientos iniciales de un sistema y el informe presentado al Auditor Interno. Adicionalmente se presentan los manuales realizados sobre las capacitaciones impartidas por el tutor empresarial en SQL, PL/SQL y el Cartridge de PL/SQL.

1. DESCRIPCIÓN DE LA PRÁCTICA EMPRESARIAL

1.1 DESCRIPCIÓN DE LA EMPRESA

1.1.1 Nombre de la Empresa

Empresa de Telecomunicaciones de Bucaramanga S.A. E.S.P. – TELEBUCARAMANGA.

1.1.2 Características de la Empresa

1.1.2.1 Visión. En el año 2005 TELEBUCARAMANGA es reconocida en Colombia por haber maximizado el valor para los accionistas, fundamentada en desarrollar relaciones de aprendizaje y confianza mutua con sus clientes, para garantizar su satisfacción.

Nuestra gente es ejemplo de servicio al cliente y su desarrollo personal y profesional permiten ofrecer una empresa sencilla, abierta, ágil, cercana y en constante desarrollo para beneficio de la comunidad en la que se desenvuelve.

1.1.2.2 Misión. Generar valor para los accionistas orientados por el cliente y el mercado, fruto de una relación de conocimiento profundo de nuestros clientes y sus necesidades de servicios de telecomunicaciones.

Innovar con soluciones integrales y de la más alta calidad y servicio, ofrecidas por un talento humano excelente y comprometido con enriquecer continuamente la experiencia de ser cliente de TELEBUCARAMANGA.

1.1.2.3 Valores Organizacionales.

- Planeación: basados en el conocimiento de la Organización y su entorno establecemos eficazmente nuestras metas y prioridades, determinando las acciones y los plazos y los recursos requeridos para alcanzarlos.

- Compromiso: entendemos y sentimos como propios los objetivos de la Organización y hacemos lo necesario para cumplirlos.

- Trabajo en equipo: integramos nuestras experiencias, conocimientos y cualidades para lograr la excelencia en los procesos y el cumplimiento de los objetivos propuestos.

- Orientación al cliente: nuestras acciones están dirigidas hacia la acción del cliente y basadas en las relaciones sólidas de conocimiento y aprendizaje permanente.

- Adaptabilidad al cambio: somos capaces de modificar nuestras propias conductas cuando surgen nuevos escenarios en la Empresa y el entorno.

- Orientación al resultado: enfocamos nuestros esfuerzos hacia el cumplimiento de los objetivos de la Organización. Siendo conscientes de la importancia del proceso, ponemos especial énfasis en el logro de los resultados.

1.1.3 Objeto Social

La empresa de Telecomunicaciones de Bucaramanga S.A. E.S.P - TELEBUCARAMANGA, tiene como objeto social:

1. La presentación de servicios públicos domiciliarios de telecomunicaciones, telemáticas y demás actividades complementarias, de valor agregado, derivadas y/o afines de tales servicios.

2. Los servicios de telecomunicaciones contemplados por la ley.

3. La sociedad podrá participar como asociada en otras empresas de servicios públicos o de las que tenga como objeto principal la prestación de un servicio o la provisión de un bien indispensable para cumplir su objeto social.

4. Inversión de capital o de bienes evaluables en dinero en otras entidades prestadoras de servicios públicos. La cobertura de los servicios se encaminaría preferentemente los municipios del territorio Colombiano siempre que pueda ofrecerlos eficientemente.

5. La sociedad podrá desarrollar actividades complementarias de los servicios que presta y brindar asesoría y asistencia técnica, operativa, administrativa e institucional a cualquier entidad oficial o de naturaleza privada, que lo solicite para el diseño, construcción, manejo o administración de los servicios públicos que formen parte de su objeto o razón social.

6. También podrá la sociedad en desarrollo de su objeto:
   a. Formar parte de consorcios, uniones temporales, alianzas, convenios o uniones estratégicas u otras figuras jurídicas semejantes a ellas aceptadas por la ley.
   b. Adquirir enajenar, administrar, construir, conservar mejorar, gravar, dar o tomar en arrendamiento o a cualquier clase de título, toda clase de bienes muebles e inmuebles, innecesarios o convenientes para el cumplimiento de sus fines.
   c. Girar, aceptar, otorgar, endosar, ceder, negociar, cobrar, descontar, y dar en prenda de garantía toda clase de títulos valores.
   d. Comprar y vender acciones, bonos, documentos de deuda pública emitidos por empresas o entidades de cualquier naturaleza.

e. Celebrar con las compañías aseguradoras operaciones relacionadas con la protección de los bienes propios o ajenos a cualquier título para la protección de sus negocios, bienes o personas que laboran a su servicio.
f. Transigir, desistir y apelar las decisiones de árbitros o de amigables componedores, en los asuntos en que tenga interés frente a terceros, asociados, administradores y demás funcionarios o trabajadores de la sociedad.
g. Contratar servicios con personas naturales o jurídicas, nacionales o extranjeras.
h. Tomar dinero a titulo mutuo, con garantías si es el caso.
i. Aceptar negociar, ceder o endosar títulos de obligaciones privadas así como celebrar el contrato de cuenta correspondiente y realizar todas las operaciones propias del giro de los negocios bancarios y financieros.
j. Adquirir u otorgar concesiones para su explotación.
k. En generar ejecutar todos los actos y celebrar todos los contratos que tienden a las realizaciones de los fines que persigue la sociedad o que se relacione con su existencia o funcionamiento.

1.1.4 Servicios prestados por la Empresa

La ruptura de la Empresa durante el año 1998 y la formalización durante el año 2002 de la Empresa de Telecomunicaciones de Bucaramanga S.A. E.S.P - TELEBUCARAMANGA, generó un cambio estructural de la Empresa que elimina la restricción de servicios exclusivos de telefonía, ampliando su espectro a las diferentes alternativas en materia de telecomunicaciones.

- Servicios de Telefonía pública básica conmutada

Línea telefónica (digital y análoga)
Línea telefónica inalámbrica

- Servicios Complementario

Código secreto
Despertador automático
Llamada en espera
Marcación abreviada
Conexión sin marcar a línea directa
Transferencia de llamadas
Transferencias de llamadas en no respuesta / ocupado
Servicio Class: complementación de llamada ocupada
Servicio Class: complementación de llamada en no respuesta
Identificador de llamadas
Línea 900 y 901

- Servicios Telemáticos

Correo de voz / Eco contestador
Línea virtual / Telebox

- Servicio de Internet

Acceso dedicado
Acceso conmutado por demanda / Internet libre
Ilimitado

- Servicio de Valor Agregado

PBX
Enlace E1
RDSI / ISDN

- Otros servicios

Línea 113
Línea 155
Línea 114

Telefonía pública
Video Combo
Red Multiservicio
Otros

1.2 DESCRIPCIÓN DEL PROYECTO

1.2.1 Objetivos

1.2.1.1 General

Aplicar y complementar los conocimientos adquiridos durante los estudios de la carrera de ingeniería de sistemas en la Universidad Industrial de Santander – UIS para apoyar el diseño y desarrollo de una herramienta software que ayude en las consultas sobre la documentación de los sistemas de información de la Empresa TELEBUCARAMANGA.

1.2.1.2 Específicos

- Ampliar los conocimientos, mediante capacitación impartida, sobre las herramientas software PL/SQL, Oracle Developer y Oracle Portal que se emplean en la Empresa para los desarrollos.

- Participar en la actualización los archivos existentes sobre documentación de los diferentes sistemas de información.

- Participar en el diseño de una base de datos de documentación para todos los sistemas de información empleados en TELEBUCARAMANGA.

- Apoyar el diseño y desarrollo de un portal web que permita a los usuarios tener un conocimiento general de los sistemas de información y de la documentación que ellos ofrecen.

- Apoyar el diseño y el desarrollo de un esquema flexible de búsquedas que facilite la extracción de información de la base de datos de documentación.

1.2.2 Justificación

La realización de esta práctica se fundamentó, en primer lugar, por el informe presentado por la empresa consultora “KPMG” la cual, a finales del año 2004, realizó un trabajo de auditoría en la Empresa y como resultado expuso para la Subgerencia de Informática y Tecnología, entre otras cosas, las falencias observadas sobre el manejo de la documentación de los sistemas de información: *“... la empresa cuenta con documentación técnica y operativa de los principales sistemas de información, ésta no se encuentra actualizada, en la cual se reflejen los últimos cambios y nuevos requerimientos desarrollados e implementados por el área de informática.”*

En segundo lugar, se buscaba desarrollar una herramienta la cual pudiera ser consultada en todo momento por los usuarios para resolver sus inquietudes sobre el funcionamiento de los diferentes sistemas de información sin tener que recurrir a los ingenieros de la Subgerencia de Informática y Tecnología para resolver sus dudas, es decir, crear una cultura de "autoservicio" en los usuarios finales y de esa forma optimizar el tiempo de trabajo de los miembros de la Subgerencia generando una mayor productividad para la Empresa.

Y en tercera instancia, se quiere evitar el riesgo de no tener una documentación al día. Actualmente no se tiene entre los miembros del área de sistemas una cultura de documentar las aplicaciones una vez se realizan, quedando la información

sobre la misma en el desarrollador el cual se convierte en indispensable al momento de presentarse dudas o al querer hacer modificaciones.

La Empresa motivada por todas éstas razones inició el proyecto de actualización de la documentación de los sistemas de información, con responsables asignados para cada tarea (miembros de la Subgerencia de Informática y Tecnología) pero debido a la carga de trabajo que poseen se vio en la necesidad de un apoyo extra en algunas tareas relacionadas con el diseño y desarrollo de aplicaciones.

En los seis meses de trabajo se participó en el diseño y desarrollo de una herramienta software que permite a los usuarios finales, desde la Intranet de empresarial o desde los mismos sistemas, conocer la documentación de los sistemas de información.

1.2.3 Plan de Trabajo

Fase 1. Reunión con los ingenieros que se encuentran vinculados al proyecto y revisión de la documentación de los sistemas de información existente en la Empresa: conocer el estado de avance que tenía el proyecto al momento de iniciar la práctica.

Fase 2. Estado del arte y planteamiento de posibles soluciones para contar con una documentación que presten los servicios necesarios para los usuarios: determinar cómo otras empresas manejan el tema de la documentación y las ayudas en línea de los sistemas de información. Levantamiento de requerimientos.

Fase 3. Desarrollo de la base de datos: para el desarrollo de la base de datos se impartió por parte del ingeniero tutor una capacitación sobre el manejo de la herramienta case que posee Oracle y junto con los demás ingenieros se diseñó y desarrolló la base de datos de documentación.

Fase 4. Desarrollo de la aplicación software: construcción de un portal web desde el cual los usuarios y los mismos ingenieros de desarrollo de la Empresa pudieran acceder a la documentación que se posea de los sistemas de información.

1.2.4 Cronograma de Actividades

La práctica comenzó el 26 de enero de 2005 y se terminó el 26 de julio del mismo año.

Figura 1. Cronograma de actividades

ACTIVIDADES	enero		febrero				marzo				abril				mayo					junio				julio			
	24	31	7	14	21	28	7	14	21	28	4	11	18	25	2	9	16	23	30	6	13	20	27	4	11	18	25
Revisión de la documentación existente	X	X																									
Diagnóstico	X	X																									
Estado del arte y posibles soluciones		X	X	X																							
Problemas con la falta de documentación		X	X																								
Búsqueda de documentación de ayuda en línea		X	X																								
Definir posibles soluciones			X	X																							
Desarrollo de la Base de Datos					X	X	X	X	X	X	X	X	X	X													
Capacitación en la herramienta CASE					X																						
Diseño de la Base de Datos					X	X																					
Poblar la Base de Datos con datos de prueba							X	X	X	X	X	X	X	X													
Desarrollo del Software								X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Capacitación y Autocapacitación								X	X	X	X	X	X	X	X	X	X										
Desarrollo de la aplicación										X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Entrega de informes de avance					24					31					3					7					12		

En el Anexo A se presenta la comparación entre las actividades planeadas y las actividades realizadas en cada uno de los periodos de tiempo comprendidos entre las entregas de los informes de avance.

2. MARCO TEÓRICO

La principal investigación realizada en esta práctica fue sobre la importancia de contar con la documentación, completa y actualizada, de los sistemas de información y sobre los problemas en que se pueden incurrir al no poseerla o no tenerla en forma adecuada.

Se consultaron varias fuentes y entre ellas una empresa española encargada de hacer este tipo de sistemas la cual tiene una metodología ya implementada para su realización, más adelante se presenta dicho procedimiento.

2.1 IMPORTANCIA DE DOCUMENTAR LOS SISTEMAS DE INFORMACIÓN

La documentación es el conjunto de información que dice qué hacen los sistemas, cómo lo hacen y para quién lo hacen. Es el material que explica las características técnicas y de operación de un sistema. Proporciona información sobre el funcionamiento de un sistema a los encargados de su mantenimiento; permite la auditoría; enseña a los usuarios finales a interactuar con el sistema y a los desarrolladores cómo modificarlo o actualizarlo.

Un desarrollo software a lo largo de su ciclo de vida debe estar acompañado de su respectiva documentación: empezando por la documentación del proceso de desarrollo; pasando por la información y manuales del sistema; y finalizando con la documentación de servicio técnico para futuras actualizaciones.

Un sistema pobremente documentado carece de valor aunque este funcionando correctamente. En el caso de programas pequeños y poco importantes que sólo se utilizan durante un corto periodo de tiempo, unos cuantos comentarios en el código podrían ser suficientes pero la mayoría de los programas cuya única documentación es el código, quedan obsoletos rápidamente y es casi imposible hacerles mantenimiento. La dedicación de un poco de esfuerzo a la documentación se ve recompensado cuando el software puede seguir funcionando incluso si cambian algunos requerimientos.

La importancia de la documentación podría compararse con la importancia de la existencia de una póliza de seguro, mientras todo va bien no existe la precaución de confirmar si la póliza está o no vigente.

La documentación adecuada y completa de un sistema que se desea implantar, mantener y actualizar en forma satisfactoria es de gran importancia, sin embargo, frecuentemente es la parte a la cual se dedica el menor tiempo y se le presta menos atención.

2.1.1 Ventajas de tener una documentación actualizada

Las ventajas de contar con la documentación de los sistemas completa y actualizada se dividieron según el tipo de usuario a quien va dirigida: los usuarios finales de los sistemas o los ingenieros desarrolladores.

2.1.1.1 Ventajas para los usuarios finales

- Ayuda en el entrenamiento de nuevo personal y sirve como referencia para todos aquellos que ya pertenecen a la Empresa.

- Facilita el uso eficiente de todos los recursos que disponen los sistemas.

- El uso de una documentación estandarizada proporciona la base de una comunicación clara y rápida entre los usuarios y los desarrolladores.

- La documentación es de gran ayuda para resolver dudas; conocer la operación básica de los sistemas; determinar errores que se cometen en los sistemas y la forma adecuada de corregirlos.

- Toda la información sobre el funcionamiento de un sistema se concentra en un lugar accesible por todo el personal de la Empresa.

- La documentación en línea además de ofrecer las mismas ventajas de la documentación impresa, permite la navegación dinámica entre temas relacionados y la interacción entre la aplicación que se está ejecutando con su información de ayuda.

2.1.1.2 Ventajas para los desarrolladores

- Contar con una documentación bien diseñada y estructurada en una base de datos, por ejemplo, permite controlar y administrar grandes volúmenes de información facilitando su actualización o modificación.

- Una buena documentación es útil para el personal que tiene la responsabilidad del mantenimiento de los sistemas.

- Ayuda a los analistas y diseñadores de sistemas en el trabajo de integración de aplicaciones con otros sistemas.

- Asegura que el sistema opere correctamente.

- La estandarización de procedimientos pactados para el desarrollo de los sistemas facilita una mejor interpretación del código por parte de todos los ingenieros del área de sistemas.

- Si la documentación está completa, actualizada y tiene la divulgación adecuada no se crearán retrasos en el tiempo de los desarrolladores o involucrados con los sistemas al tener que realizar nuevas implementaciones, desconociendo si dicha opción ya se encuentra disponible.

- Se amplían los servicios que prestan los sistemas sin necesidad de más personal y mejorando los tiempos de realización de tareas.

Es importante enfatizar en las ventajas de la documentación para tener desarrolladores con código más legibles y conscientes de documentar sus desarrollos en todo momento.

2.2 POR QUÉ NO SE DOCUMENTAN LOS SISTEMAS?

Algunas de las razones o disculpas que los desarrolladores dan para posponer la elaboración de la documentación de los sistemas son:

- *"Más tarde lo hago"*

Aunque un desarrollador realmente tenga la intención de hacer la documentación de los sistemas, siempre colocará otras tareas como más importantes, dejando a un lado el desarrollo de esta labor. Mientras más tiempo se espera para hacerla más se olvida, terminando con un documento menos detallado y menos útil.

- *"Para qué escribirlo? Yo me acuerdo."*

Con el paso del tiempo se recuerda menos sobre el desarrollo de un sistema y sin darse cuenta puede hasta olvidarse completamente qué se hizo o cómo se realizó un sistema. Esto conduce a una pérdida de tiempo tratando de reaprender lo olvidado o teniendo que realizar de nuevo una aplicación ya existente porque debe adaptarse a ciertas modificaciones.

- *"Si lo mantengo en mi memoria, no me despedirán, así tendré seguridad laboral!"*

Aunque esto puede funcionar por un tiempo, frecuentemente conduce a menos, y no más, seguridad laboral. Si se presenta una emergencia y el desarrollador no se encuentra disponible para solucionarla, la documentación puede ayudar a que alguien más la resuelva. Si el desarrollador cuenta con una buena documentación de sus sistemas quedará bien ante las directivas de la empresa ya que en su ausencia los problemas se pueden solucionar.

2.3 PROBLEMAS CON LA DOCUMENTACIÓN

Si no se cuenta con una adecuada documentación de los sistemas, no está al alcance de todos los usuarios o si está incompleta o desactualizada se puede incurrir en los siguientes problemas:

- Uso inadecuado o desuso de las aplicaciones u opciones que contienen los sistema de información.

- Si la documentación de un sistema esta incompleta el desarrollador continuamente estará involucrado con él y no podrá moverse a otra asignación o avanzar en su carrera profesional.

- Si un desarrollador, a cargo de uno de los sistemas de información, se retira de la empresa todo el conocimiento que aportaba desaparecerá con él, lo que puede provocar graves prejuicios para la empresa y los usuarios de sus sistemas.

- Impide la reparación, optimización o mejoras a los sistemas por parte de nuevos desarrolladores al no poseer suficiente información sobre su funcionamiento.

- La falta de documentación no sólo genera trabajo adicional, sino que también tiende a dañar la calidad del código de un software. Si no se documentan las decisiones tomadas en el desarrollo se cometerán graves errores al tratar de comprender lo que pudo haberse descrito fácilmente en una ocasión.

- Los principiantes en el tema de la documentación tienden a centrar sus esfuerzos en temas sencillos ya que éstos les resultan más fáciles de documentar. Esto es una pérdida de tiempo; no se aprende nada de este esfuerzo y se termina escribiendo una documentación poco útil.

- Documentar de forma breve es un error habitual entre los desarrolladores, pero el otro extremo es igual de nocivo; si se escriben documentaciones extensas, se agobiará al lector y serán una carga a la hora de conservarlas. Es esencial documentar sólo lo necesario. La documentación no sirve de ayuda para nadie si su extensión desanima a la hora de leerla.

- Cuando la documentación se convierte en un requisito, el desarrollador se ve en la necesidad de completar ésta tarea en poco tiempo y con herramientas que no siempre son las adecuadas para un trabajo tan complejo.

Para solucionar estos inconvenientes es fundamental que no se piense en la documentación como un asunto rutinario y aburrido, si se hace, la documentación no servirá para nada y será penosa a la hora de se utilizada por los usuarios finales. Se debe documentar de forma consciente, preguntándose a medida que se realiza, por qué se hace y si se está empleando el tiempo de la forma más eficaz en algo que realmente será útil.

2.4 QUÉ SE DEBE DOCUMENTAR?

A la hora de decidir qué se debe documentar, tienen prioridad:

- *Políticas:* las políticas se dan para formalizar y aclarar la relación existente entre los desarrolladores y los usuarios finales de los sistemas; establecen la forma en que se manejan las solicitudes de recursos o de asistencia. La naturaleza, estilo y método de difusión de las políticas varían según la empresa.

- *Procedimientos:* los procedimientos son secuencias de pasos sobre acciones que deben realizarse para alcanzar una tarea determinada. Si un procedimiento es ejecutado más de una vez, es una buena idea tenerlo documentado.

- *Cambios:* una gran parte del trabajo de un desarrollador de sistemas gira alrededor de ejecutar cambios. Todos los cambios deben estar documentados, de lo contrario, se pueden presentar confusiones con los cambios realizados meses atrás.

2.5 CUÁNDO SE DEBE DOCUMENTAR?

Otro asunto importante es decidir cuándo documentar. Aunque algunas veces es conveniente posponer esta tarea mientras se realizan pruebas, los programadores con experiencia suelen documentar el código provisional, el análisis a un problema inicial y los borradores de un diseño, esto hace que la experimentación sea más productiva. Además, dado que se toma la documentación como hábito, resulta normal documentar a medida que se va avanzando.

2.6 PROCEDIMIENTO PARA ELABORAR UNA DOCUMENTACIÓN

Según la empresa española Reinisch[1], dedicada al desarrollo de documentación técnica de compañías de diversas actividades económicas, el procedimiento para elaborar una correcta documentación debe contener los siguientes pasos:

2.6.1 Análisis del grupo receptor: ¿a quién va dirigida la documentación?.

La forma y el contenido de la documentación dependen siempre del usuario. En consecuencia, la primera y principal pregunta que se debe plantear para elaborar correctamente cualquier documentación debe ser: ¿quién utilizará esa

[1] www.reinisch.es. documentación técnica.pdf.

documentación?. Una definición precisa e inequívoca del grupo receptor evitará errores en la documentación elaborada.

2.6.2 Investigación del producto: "el redactor también se ensucia las manos". Sólo una persona con conocimientos especializados, y al mismo tiempo capaz de ponerse en el lugar del usuario, podrá redactar una documentación que cumpla con las necesidades del mismo. Para obtener un resultado satisfactorio es fundamental conocer a la perfección el funcionamiento práctico de un sistema. Sólo quien está en contacto directo con un sistema será capaz de elaborar una documentación clara y estructurada.

2.6.3 Diseño de la documentación: una documentación didáctica. Todos los usuarios tienen un punto en común: todos demandan una documentación clara y comprensible a primera vista. La adecuada combinación entre explicaciones textuales e ilustraciones, permite un óptimo aprovechamiento y hace de la documentación un apoyo eficaz en la utilización de un sistema.

2.6.4 Medios de distribución: papel, CD-ROM, web, etc. Un factor esencial para determinar la forma y estructura de la documentación es el medio de distribución de la misma. A menudo, será necesario publicar la documentación a través de distintos medios para que todos los usuarios puedan tener acceso a ella.

3. ESTUDIO PRELIMINAR

El punto de partida de esta práctica fue el estudio realizado al actual sistema de documentación que posee la Empresa Telebucaramanga, la recopilación de la opinión de los diferentes usuarios y el planteamiento de los requerimientos del nuevo sistema de documentación. Este capítulo plantea los resultados de los análisis y estudios realizados al respecto.

3.1 ANÁLISIS DE LA DOCUMENTACIÓN EXISTENTE

La Empresa TELEBUCARAMANGA cuenta con la documentación de los sistemas de información que maneja pero se encuentra desactualizada, tanto en los archivos impresos como en la intranet de la Empresa; su estructura, presentación y contenido es poco amigable y didáctica. Ésta es la opinión general de los ingenieros desarrolladores de la Subgerencia de Informática y Tecnología quienes al responder una encuesta realizada, ampliaron la definición inicial planteada por la estudiante en práctica.

Inicialmente en forma individual se analizó la documentación del principal sistema de información de la Empresa llamado SINTEL y de los seis (6) módulos en que está dividido, denominados CENTEL, DANTEL, FACTEL, MULTIRED, PUBLITEL Y SAPTEL; se observaron en forma general y de ellos se sacaron conclusiones sobre la estructura y el contenido de la documentación de todos los sistemas ya que fue realizada para todos los sistemas siguiendo un mismo patrón.

3.1.1 Observaciones preliminares sobre la actual documentación

El primer inconveniente encontrado a la documentación es su ubicación, los archivos impresos se encuentran en la Subgerencia de Informática y Tecnología y no en cada área donde se necesitan y la ruta para encontrarla en la intranet de la Empresa no es conocida por la mayoría de usuarios.

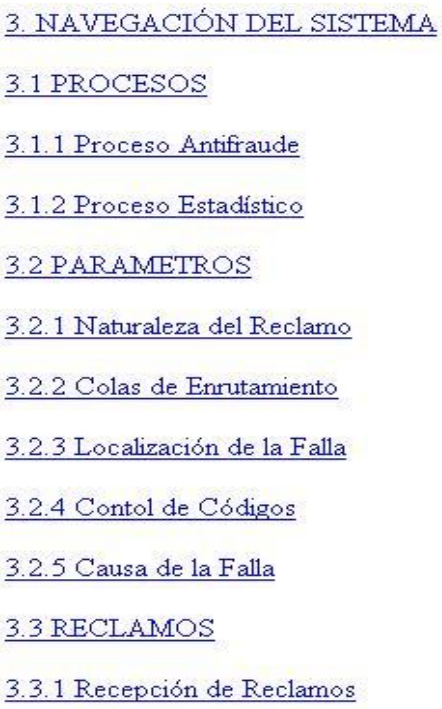
Para acceder a los manuales de los sistemas de información de la Empresa es necesario ingresar a la Intranet (http://telsun/intranet) e ir al link *procesos* que se encuentra en la barra principal ubicada después del logo; dar clic en el link *manuales sistemas de información* y por último en el link del sistema o módulo del cual se quiere conocer su documentación.

Figura 2. Ruta de acceso a los manuales desde la intranet empresarial

La documentación de los sistemas no es más que los archivos que se realizaron en formato de texto convertidos en formato HTML, por eso, su estructura y presentación es estática.

La pantalla inicial de la documentación de cualquier sistema está formada por links que guían el acceso a cualquier opción del sistema.

Figura 3. Presentación inicial del sistema de documentación



Al ingresar a la documentación de un sistema se da una pequeña explicación de sus funciones y se presenta una imagen de la pantalla principal. De igual forma cuando se accede a la documentación de una opción en particular se presenta una explicación seguida de una imagen.

Figura 4. Estructura interna de la documentación

3.1 CLIENTES

3.1.1 Cuentas Corrientes

Esta opción tiene como finalidad permitir el registro de las cuentas corrientes empleadas aquellos usuarios que así lo soliciten. Las líneas pertenecientes a una cuenta corriente Cédula o NIT del responsable, Barrio, Dirección, Teléfono, Zona Postal, Código Postal y sucursal. Ver Figura No.3

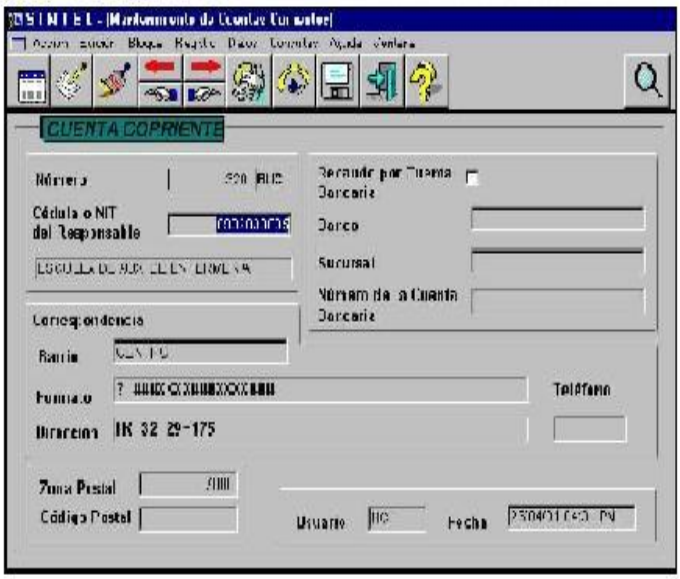


Figura No 3 Cuenta Corriente

Los aspectos más sobresalientes del análisis que se realizó de los diferentes módulos del sistema de información SINTEL fueron:

- Es deficiente en la redacción de textos que explican el funcionamiento de las pantallas de los sistema, en la mayoría de casos no se muestra cuáles son los datos que deben ingresarse ni mucho menos cómo debe hacerse el ingreso (por ejemplo el caso en que se debe llenar un campo con siglas). La información que se presenta antes o después de una imagen no es muy explícita y en muchas ocasiones no se presenta una descripción del formulario y de lo que debe hacerse en él.

- No se presentan ejemplos que tengan una secuencia lógica que guíen el ingreso de datos a una pantalla o que enseñen cómo se realiza una actividad específica.

- La calidad de algunas imágenes es mala; algunos pantallas no poseen imágenes y si las tienen no explican claramente de que campos se compone.

- La documentación es poco didáctica en su estructura, lo cual dificulta el proceso de entrenamiento y capacitación de nuevos usuarios en el uso de un sistemas de información.

- La documentación en su estado general se encuentra incompleta, no contempla una explicación de todos las opciones de los módulos, muchas pantallas e íconos están sin su respectiva documentación y la falta de actualización impide conocer las nuevas opciones del sistema por parte de los usuarios que las maneja.

3.1.2 Elaboración de la encuesta

Junto al tutor empresarial se decidió utilizar una de las técnicas para recolectar información y de esa forma conocer la opinión que se tiene en la Empresa de la actual documentación de los sistemas. Se realizó una encuesta a los siete (7) ingenieros desarrolladores integrantes de la Subgerencia de Informática y

Tecnología de la Empresa los cuales con sus respuestas reforzaron la definición dada inicialmente sobre la documentación.

La recolección de información se refiere al uso de una gran diversidad de técnicas y herramientas que pueden ser utilizadas por el analista para desarrollar los sistemas de información con mejor calidad y siempre pensando en las necesidades de los usuarios. Estas técnicas pueden ser la entrevista, la encuesta, el cuestionario, la observación, el diagrama de flujo y el diccionario de datos.

Después de estudiar los métodos más utilizados se decidió realizar una encuesta el cual es un método para obtener información acerca de un tema específico; consiste en una serie de preguntas que se han elaborado previamente y son aplicadas a una persona en particular. La diferencia con la entrevista es que no requiere que la persona que la aplique sea un especialista en el tema, es más simple y no requiere que el encuestado entre en una gran comunicación con el encuestador. Una encuesta recoge información de una "muestra" que es sólo una porción de la población bajo estudio.

En el procedimiento seguido para la elaboración y la ejecución de la encuesta fue el siguiente[2]:

1. Definición de los objetivos de la encuesta.
2. Elección de la población a encuestar.
3. Determinación de la muestra.
4. Elaboración del cuestionario.
   - ü Titulo de la encuesta.
   - ü Un párrafo para informar al encuestado del motivo de la encuesta.
   - ü Párrafo que motive al encuestado a responder sinceramente.
   - ü Tres preguntas que permitan conocer al publico encuestado.

[2] ¿Cómo determinar los requerimientos de usuario?. http://www.uc.cl/related/atees/chile/domeyko/html/plangral/1ros/como.pdf.

- ü Tres preguntas de cuestiones relacionadas con el objeto del proyecto.
- ü Cuatro preguntas sobre características del objetivo del proyecto.
- ü Pregunta respecto al "precio" que el posible usuario está dispuesto a pagar por el objeto del proyecto.

5. Elección y formación de los encuestados y encuestadores.
6. Realización de la encuesta entre la población objetivo.
7. Análisis de los resultados.
8. Interpretación de los resultados de la encuesta.

3.1.3 Resultado de la encuesta

En la encuesta planteada a los ingenieros desarrolladores de la Subgerencia (en el Anexo B se muestra el formato de la encuesta utilizado) se trataron dos aspectos, con el primero se quería conocer la opinión que tienen de la actual documentación y con el segundo saber cómo les gustaría que fuera la presentación de la misma para que sea realmente utilizada por los usuarios finales.

A las preguntas sobre la opinión que se tiene de la actual documentación de los sistemas, los ingenieros contestaron:

- ü La gran mayoría de ingenieros (más del 85%) opinan que la actual estructura de la documentación no ayuda a solucionar inquietudes propias o de los usuarios.

- ü Casi el 43% de los ingenieros ha tenido que implementar alguna opción ya disponible, lo cual indica que la inexistencia de una buena documentación hace que se pierdan tiempo con el desarrollo aplicaciones ya creadas.

- Los ingenieros son consultados para explicar la funcionalidad de los sistemas con una frecuencia del 28% para cada una de las opciones una vez a la semana y varias veces a la semana.

- La intranet es el mejor sitio para colocar la documentación de los sistemas de información pero es conveniente colocarla en un lugar de fácil acceso y hacer la publicidad correspondiente para que todos los usuarios puedan conocerla y manejarla.

3.1.4 Diagnóstico general de la actual documentación

A parte de la encuesta realizada a los ingenieros desarrolladores de la Subgerencia, el otro estudiante en práctica en la Empresa realizó una encuesta para los usuarios finales de los sistemas de información obteniendo como resultado, entre otras cosas:

- La mayoría de los usuarios (más del 73%) cuando necesitan acceder a opciones diferentes a las que habitualmente maneja de los sistemas buscan ayuda en otras personas antes de consultar los manuales disponibles.

- Casi la mitad de los encuestados desconocen que la Empresa cuenta con manuales de los sistemas en archivos impresos y en la intranet.

- De los usuarios que han utilizado los manuales que se encuentran en la intranet (menos del 37% de los encuestados) casi el 30% respondieron que no encontraron fácilmente lo que buscaban; que no fue satisfactorio para resolver su inquietud lo encontrado y que un cambio en la redacción de textos y la distribución de contenidos facilitaría la compresión y la haría más agradable de utilizar.

Se concluye, con estos tres puntos de vista, el de la estudiante en práctica, el de los ingenieros desarrolladores y de los usuarios finales de los sistemas, que:

- La documentación existente no presta los servicios adecuados para los usuarios de los sistemas ni para los ingenieros desarrolladores.

- Un cambio en la estructura y presentación de los contenidos de los manuales facilitaría su uso y compresión por parte de los usuarios y evitaría que los ingenieros de la Subgerencia inviertan más tiempo del necesario explicando el funcionamiento de los sistemas.

- El uso de la intranet de la Empresa para colocar los manuales de los sistemas crea un acercamiento entre el sitio donde los usuarios realizan su trabajo y donde pueden resolver sus dudas.

3.2 ESTUDIO DE POSIBLES SOLUCIONES

Se trabajó en la búsqueda de información sobre las diferentes formas cómo documentan los sistemas e implementan ayudas en línea las empresas que prestan el servicio de cursos virtuales, empresas que comercializan con software, entre otras empresas afines.

Después de eliminar todas aquellas ayudas que se encuentran en formatos texto - imagen o solo texto se realizó un documento con aquellas ayudas cuya estructura y presentación pueden guiar el proceso de desarrollo de la interfaz del portal web. Estas son algunas de las opciones que se presentaron para obtener un primer modelo de cómo sería el diseño de la interfaz del portal web.

3.2.1 Acceso a la documentación

- *Promocionar el acceso*: para que el manual sea realmente utilizado se ve la necesidad de promocionarlo ya sea mediante el uso del correo interno de la Empresa o por pequeños anuncio en la intranet.

- *Creación de un acceso más sencillo*: teniendo en cuenta los resultados de las encuestas se ve la necesidad de crear en la página de la intranet una ruta más accesible a los manuales, diferente a la actual (procesos => manuales de sistemas de información => SINTEL => ...) ya que ésta no se encuentra a simple vista por el usuario.

3.2.2 Pantalla principal

En las páginas como *http://www.dpi.inpe.br/spring/usuario_spa/indice.htm* (sistema de ayuda *on-line* de SPRING Sistema de Procesamiento de Información Georeferenciada) o como *http://www.prado.com.mx/Orange_Internet/ORANGE_Internet.htm* (sistema para llevar a través de internet el proceso contable de una pequeña o mediana empresa) se hace uso de un menú inicial, el cual permite al usuario tener acceso a varias opciones del sistema desde la pantalla inicial.

Las posibles opciones con que puede contar la pantalla inicial del portal web son:

- *Ayuda*: la idea de esta opción es que el usuario echen un vistazo a la página con la que se va a encontrar al acceder al sistema de documentación. Esta opción es pensada en aquellos usuario que por primera vez tienen acceso al sistema de documentación.

- *Ingreso a manuales*: mediante esta opción el usuario experto podrá acceder a los manuales de los sistemas de información y el usuario que esta en capacitación podrá conocer su funcionamiento.

- *¿Qué hay de nuevo?*: allí el usuario encontrará noticias de nuevas opciones que han sido desarrolladas o han sido modificadas de los sistemas de información.

3.2.3 Glosario de términos

- *A nivel interno*: en la página *emagister.com*, una página de cursos en línea, se encontró un javascript de ventana emergente usado para promocionar obsequios por inscripción en algunos cursos. Se cree que es posible utilizarlo para definiciones de términos, abreviaciones o palabras cuyo desconocimiento afecten el aprendizaje de los conceptos manejados en la descripción de contenidos de los sistemas del información

- *A nivel externo*: sería una de las opciones del menú principal, dado el caso en que el usuario desee consultar algún término que desconozca o una sigla de la cual haya olvidado su significado y que aparece en alguno de los formularios que maneja.

3.2.4 Buscadores

La opción de un buscador sencillo es muy utilizado en casi todas las páginas consultadas, se ubica en un frame que permanezca constante para realizar la consulta desde cualquier parte del sistema sin tener que devolverse a la página principal. También puede tener un link para acceder a una búsqueda avanzada, una opción muy utilizada por usuarios que desean consultar temas más específicos.

Las páginas de bibliotecas virtuales o periódicos (la página *www.elpais.es* cuenta con un buscador sencillo en el frame superior y un buscador avanzado bastante bueno) son un buen ejemplo esta clase de buscadores.

3.2.5 Observaciones generales sobre el portal web

- Para la estructura de la interfaz de la página de inicio se observaron diferentes páginas como la de Oracle (*www.oracle.com*), que posee un menú de árbol el cual permite explorar el contenido de las opciones.

- Es importante que cada página cuente con una ruta que indique al usuario en donde se encuentra.

- Otra forma de orientar y ubicar al usuario en el contenido de cada sistema es colocar una pequeña descripción de lo que realiza la opción, resaltándola frente a las otras opciones (por ejemplo la página de documentación en línea de grandiyasociados.com).

- También se puede hacer una ampliación de la opción a explicar o simplemente resaltándola de las demás opciones, acompañada del texto donde, de forma clara y sencilla, se explica lo que realiza dicha pantalla (la página deremate.com.co utiliza esta técnica en su ayuda).

- La Empresa de software Oracle implementa las ayudas de sus programas mediante el uso de diferentes tipos de archivos, documentos con texto e imágenes, videos explicativos e incluso con voz. La forma de mostrar estos archivos será el punto de partida y el modelo a seguir para el diseño de la forma de cómo se mostrarán los archivos de ayuda en el portal web de documentación.

3.3 REQUERIMIENTOS DEL SISTEMA

Para levantar los requerimientos del sistema que se esta apoyando en su desarrollo se siguió el modelo utilizado en la Empresa. El Anexo C muestra el formato utilizado para lograr este fin.

3.3.1 Nombre del Sistema

DOCUMENTACIÓN TÉCNICA Y DE USUARIO FINAL (DTU)

Descripción General

El sistema de Documentación Técnica y de Usuario Final nace con la necesidad cambiar el actual sistema utilizado para documentar los sistemas de información de la Empresa de Telecomunicaciones de Bucaramanga E.S.P TELEBUCARAMANGA. El sistema estará dividido en dos módulos, por un lado la interfaz para el ingreso de datos y por otro el portal web desde el cual los usuarios podrán consultar la información sobre documentación.

3.3.2 Nombre del Módulo

PORTAL WEB

Descripción General

El Portal Web será la nueva herramienta utilizada para consultar los archivos multimedia que documentan los sistemas de información manejados en la Empresa. Tendrá dos clases de usuarios: los usuarios finales y los ingenieros desarrolladores de la Subgerencia de Informática y Tecnología de la Empresa quienes adicionalmente tendrán acceso a documentación del código del software desarrollado.

3.3.3 Requerimientos de Entorno

El entorno es todo lo que rodea al sistema; el sistema usa el entorno y lo necesita como una fuente de los servicios necesarios para que funcione.

Tabla 1. Requerimientos de Entorno

Sistema	***Descripción***
Interfaz para el ingreso de registros a la base de datos de documentación	Este es el otro módulo del sistema de documentación (DTU). Con ésta interfaz se hace el ingreso de registros a la base de datos de documentación los cuales son mostrados desde el portal. La herramienta está desarrollada en Forms Builder 10g.
Sistemas de información de la Empresa	Los sistemas de información de la Empresa están desarrollados en la herramienta Oracle Forms Builder 6. Por medio del botón de ayuda de cada pantalla se hace el llamado al portal web y específicamente al componente del cual se quiere consultar su ayuda.

3.3.4 Requerimientos Funcionales

Describen lo que el sistema debe hacer más no cómo lo hace. Estos requerimientos se convierten en los algoritmos, la lógica y gran parte del código del sistema.

Tabla 2. Requerimientos Funcionales

Requerimiento	***Descripción***
Ingreso	El sistema debe permitir el ingreso a dos clases de usuarios. Los usuarios de la Subgerencia de Informática y Tecnología serán los únicos que para ingresar introducirán un login y password que los identifique y de esa forma puedan tener acceso a la información que los usuarios finales no pueden acceder.
Buscador Sencillo	Un buscador sencillo en donde los usuarios ubicados en cualquier página del sistema puedan hacer una búsqueda sin necesidad de ir a una página en concreto.
Buscador Avanzado	Un buscador avanzado que permita hacer restricciones en la búsqueda y de esa forma obtener resultados más específicos. Las restricciones pueden ser de sistema donde se busca, de tipo de objeto (si es una pantalla, un reporte o un programa lo que se busca), de tipo de componente (si lo buscado es una forma, una ventana, un bloque de datos, un icono, etc.) y de un rango de fechas entre las cuales se encuentre la fecha de creación o modificación del componente buscado.
Manuales	Debe permitir la rápida y completa capacitación de personal nuevo en el uso de un sistema de información o de una parte específica de él.
Glosario de Términos	Glosario de términos de telecomunicaciones o siglas que se manejen con frecuencia en la Empresa o que se utilicen en el llenado de una pantalla o reporte.

Requerimiento	*Descripción*
Información completa de un componente	Debe mostrar, según el tipo de usuario que ingresa al sistema, la información completa de un componente y sus ayudas multimedia. Si el usuario es un ingeniero desarrollador debe tener acceso a los archivos de texto que amplían su descripción, a los componentes que se ven afectados por el componente que se esta revisando y las variables que interactúan en él (para el caso de tratarse de un componente de tipo objeto programa).
Ayudas Multimedia	De las ayudas multimedia se debe mostrar la información básica (nombre, descripción y fecha de creación o modificación), el tipo de archivo (imagen, video o sonido) y el respectivo link para acceder a él.
Estadísticas de Ingreso	Se debe guardar el registro de las visitas que se realicen a los componentes y los archivos multimedia. Es necesario conocer la dirección ip y el nombre del equipo desde el cual se ingresa al sistema.

3.3.5 Requerimientos de Interfaz

La interfaz es como interactúa el sistema con el ser humano o con otros sistemas.

Tabla 3. Requerimientos de Interfaz

Requerimiento	*Descripción*
Presentación	La pantalla principal o de presentación del sistema debe contener las opciones de ingreso a todas las funciones del sistema.

Requerimiento	*Descripción*
Noticias	Se debe tener un espacio especial para colocar noticias de interés para los usuarios. Debe ser de fácil actualización.
Ayuda del sistema	La ayuda del sistema tiene que ser de fácil acceso.
Información sobre la ubicación	Las páginas deben contener una ruta sobre la ubicación actual del usuario.

3.3.6 Requerimientos Generales

Se agrupan los demás requerimientos necesarios para el correcto funcionamiento del sistema.

Tabla 4. Requerimientos Generales

Requerimiento	*Descripción*
Estándar de programación	Se deben manejar estándares en la programación siempre pensando en las mejores prácticas.
Ubicación	El portal web de documentación debe estar disponible para todos los usuarios en un lugar al cual todos tengan acceso (intranet de la Empresa, por ejemplo).
Dirección de acceso	La dirección de acceso o su ubicación dentro de la página de la intranet debe ser fácil de recordar o de encontrar.
Almacenamiento de archivos	Se debe tener cuidado con el almacenamiento de archivos multimedia en el servidor, los archivos desactualizados o poco consultados deben ser retirados.

Requerimiento	***Descripción***
Documentación del sistema DTU	El sistema igual que todos los sistemas de información de la Empresa debe tener sus archivos de documentación actualizados.
Capacitación a los Ingeniero Desarrolladores de la Subgerencia	Se debe impartir a todos los ingenieros desarrolladores de la Subgerencia de Informática y Tecnología una capacitación sobre el software desarrollado para posteriores mejoras.

4. BASE DE DATOS DE DOCUMENTACIÓN

En el primer diseño que se participó fue en la creación del diagrama Entidad – Relación para ello inicialmente se capacitó en el uso de la herramienta case de Oracle y en especial en los módulos que se utilizarían de ella y posteriormente se analizó la estructura de la base de datos, se elaboraron los respectivos diagramas y se crearon las tablas que la conformarían.

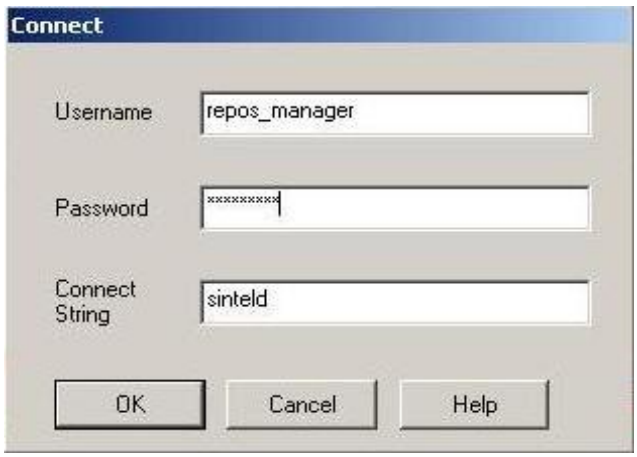
4.1 HERRAMIENTA CASE ORACLE DESIGNER

La case que posee Oracle9i AS llamada Designer es una herramienta software utilizada para analizar los requerimientos de las organización y diseñar y generar sistemas cliente/servidor. Incorpora las ayuda necesarias para modelar organizaciones, analizar sistemas, diseñar software y generar procesos. Oracle Designer proporciona un repositorio multiusos y se integra fácilmente con Oracle Developer.

Oracle Designer (anteriormente llamado Designer/2000) es una de las mejores herramientas case que existen actualmente en el mercado, especialmente para el modelamiento de sistemas de información y desarrollo rápido de aplicaciones.

Para empezar a trabajar en la herramienta case es necesario tener un username, un password y saber el nombre de la base de datos a la cual se debe conectar para hacer los desarrollos.

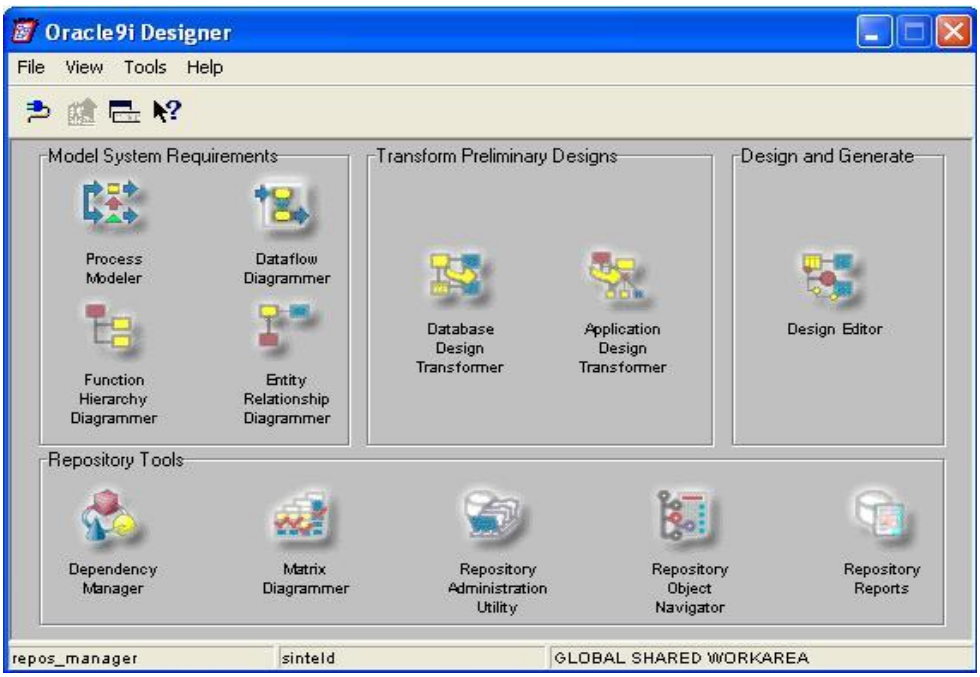
Figura 5. Acceso a la Herramienta Case Oracle Designer



4.1.1 Componentes

La herramienta case Designer tiene cuatro componentes principales incluyendo utilidades y diagramadores:

Figura 6. Presentación de las diferentes utilidades de Oracle Designer



4.1.1.1 Modelador de requerimientos del sistema. Este módulo modela los procesos del negocio; describe los requisitos del negocio; reexamina los métodos usados para alcanzar las metas de la organización y crea las representaciones diagramáticas de los procesos, los flujos de datos, las funciones jerárquicas y de las entidades y relaciones de los sistemas de la organización.

Utiliza cuatro herramientas para realizar los diferentes diagramas:

- Modelador de Procesos (Process Modeller): Esta herramienta muestra procesos y flujos, así como las unidades de organización afectadas por ellos. El Modelador de procesos se utiliza para mostrar conceptualmente, y en muchos casos redefinir, lo que sucede en el sistema actual o lo que sucederá en un sistema nuevo.

- Diagramador de jerarquías de funciones (Function Hierarchy Diagrammer): Este diagramador permite mostrar los diferentes niveles de procesos o funciones (términos que Designer considera sinónimos) del sistema en un único diagrama. Esta herramienta evita tener que definir flujos, lo cual tendría que hacerse en los casos del Modelador de Procesos o el Diagramador de flujos de datos.

- Diagramador Entidad - Relación (Entity Relationship Diagrammer): Este diagramador muestra las entidades con sus atributos y relaciones, representando los modelos lógicos de los datos. Las entidades que se creen aquí serán la entrada de las asociaciones de uso de datos de las funciones del Diagramador de jerarquía de funciones (FHD), por lo que normalmente se utilizará el Diagramador E-R al mismo tiempo que el FHD, o por lo menos antes de asociar las funciones del FHD con entidades.

- Diagramador de flujos de datos (Data Flow Diagrammer): Este diagramador permite, como el Modelador de procesos, mostrar funciones y flujos de datos. Muestra la misma información que el Modelador de procesos, pero con un tipo

diferente de diagrama que resulta más familiar para analistas y usuarios. Una de las características del Diagramador de flujos de datos que no tiene el Modelador de procesos es la asociación de los elementos de los datos (entidades y atributos) con funciones y flujos.

4.1.1.2 Generación del diseño preliminar. Este módulo incorpora dos herramientas: Transformador del diseño de la base de datos y Transformador del diseño de la aplicación.

- El transformador del diseño de la base de datos realiza la transformación del diseño conceptual de la base de datos al diseño lógico. Puede ejecutarse más de una vez para poder realizar cambios.

- El transformador del diseño de la aplicación busca los procesos diseñados en la jerarquía de procesos mirando cual es el uso que hace de las entidades del sistema.

4.1.1.3 Diseño y generación del sistema. En este módulo se combina tanto el diseño del sistema como su posible generación. En esta fase se utilizan los objetos diseñados en módulos anteriores o los crearlos directamente en él. Esta segunda aproximación es útil para crear prototipos rápidos, tanto formularios como informes.

La incorporación en este módulo de ambas fases es consecuente con la política de Designer para generar código: se diseña el módulo, se genera, se evalúa, se determinan mejoras, se vuelve a rediseñar y se vuelve a generar, hasta alcanzar la solución óptima.

4.1.1.4 Utilidades del repositorio. Consiste en tablas que almacenan información sobre el sistema que se está analizando, diseñando y produciendo.

Como el repositorio es una base estándar de Oracle, tiene todos los beneficios y consideraciones de un sistema multiusuario: seguridad, conectividad, concurrencia, etc. Estas utilidades asisten a lo largo de todo el ciclo de vida del desarrollo de sistemas para comprobar, cambiar o complementar el trabajo de repositorio que se efectúa con los diagramadores, generadores y otras utilidades de Designer.

4.2 ANÁLISIS DE LA ESTRUCTURA DE LA BASE DE DATOS

Entendiendo el funcionamiento de la herramienta case, Oracle Designer, se procedió a realizar el análisis de la estructura de la base de datos que almacenará los archivos multimedia correspondientes a la documentación de los sistemas de información de la Empresa.

Se comenzó con el estudio de cómo sería la mejor forma para almacenar dichos archivos y para ello se consultó sobre el manejo de las bases de datos multimedia las cuales almacenan directamente los archivos en las tablas como flujos de bytes en campos de tipo binarios. Este fue el principal inconveniente encontrado y para solucionarlo se estudió el uso de los tipos de datos Lob y BFile de Oracle que manejan de otra forma el almacenamiento de esta clase de archivos.

4.2.1 Almacenamiento multimedia en Base de Datos

En ocasiones se presenta la necesidad de almacenar cierta información como imágenes, documentos hechos en procesadores de texto, hojas de cálculo, diapositivas e incluso archivos de video o de sonido en una base de datos, ya sea porque es indispensable cumplir con los requerimientos del negocio o de las personas que en últimas son los que están en contacto directo con los sistemas, los usuarios finales.

Se puede decir que existen dos tipos de información que se almacena en una base de datos, la información de carácter “estructurado” o información simple y la información “no estructurada” o información compleja. En términos de base de datos y almacenamiento se llama información simple a un campo en una tabla que guarda, por ejemplo, el nombre de un empleado y para la cual existe un tipo de dato básico como una cadena de caracteres y el tipo de información compleja sería la correspondiente a la almacenada en un campo que guarda la fotografía del empleado, por ejemplo.

Se pueden enumerar ventajas y desventajas de este tipo de bases de datos. Una ventaja es que tanto la información "estructurada" como la "no estructurada" relacionadas lógicamente, están almacenadas en un mismo repositorio y formando parte de un registro en una base de datos, así permanecen siempre unidas aún si la base de datos cambia de lugar o de servidor. Un ejemplo es la posibilidad de tener en una tabla llamada Empleado información como el nombre, apellidos, dirección, etc. y la fotografía del mismo en la misma tabla, incluso su hoja de vida en un archivo de texto.

Otro beneficio es la seguridad de la información ya que los documentos se almacenaran en su versión definitiva para asegurar la unicidad y para que no estén expuestos a cambios que dañen su integridad. Para estos casos las bases de datos corporativas manejan mecanismos de protección que permiten autorizar o denegar el cambio de ciertos campos a determinados usuarios de base de datos.

Pero la principal desventaja encontrada es el consumo de espacio en la base de datos por lo que es recomendable no abusar en almacenar tanta información "no estructurada" ya que esta se almacena como flujos de bytes en campos de tipo binarios para los cuales se mantiene en una proporción mayor a la usada para los tipos de datos simples.

4.2.2 Tipos de datos LOB y BFILE de Oracle

Los tipos de datos que permiten almacenar sonidos, imágenes, videos y textos de gran tamaño corresponden a los tipos LOB (Large OBject) de ORACLE (BLOB, CLOB, NCLOB y BFILE). Estos tipos de datos permiten almacenar datos hasta de 4 gigabytes de tamaño y por lo tanto requieren un manejo diferente.

El tipo de datos BLOB (Binary Large Object) es un tipo de dato binario que puede utilizarse para almacenar cualquier tipo de dato; los tipos CLOB (Character Large Object) puede utilizarse para almacenar grandes bloques de información textual y el NCLOB es un LOB de caracteres multibyte.

El tipo de datos BFILE de Oracle no es un LOB propiamente, se diferencia de ellos porque sus datos se almacenan en un archivo físico del servidor y no en la base de datos, en ella solo se almacena el puntero a la dirección física. Los tipo de datos BFILE sólo proporciona acceso de lectura a los datos.

Este tipo de datos tiene las siguientes características:

- Contiene o almacena datos “no estructurados”.
- Admite fragmentación en el servidor donde se guardan los archivos.
- Utiliza semántica de copia de referencia. Por ejemplo, si se lleva a cabo una operación de copia de un BFile, sólo se copia el localizador o puntero y no el archivo.

Se utiliza el tipo de datos BFile en el desarrollo de la base de datos de documentación para almacenar en el servidor, y no en la base de datos, los archivo multimedia ya que por su tamaño pueden causar demora en el equipo cliente al cargarse y porque no se necesita tanta seguridad como con otros archivos.

4.2.2.1 Forma de manejo de los BFile. Una vez decidido el tipo de datos a utilizar para almacenar los archivos multimedia en la base de datos de documentación se procedió a hacer la configuración correspondiente para su manejo.

Inicialmente se crea el objeto Oracle de tipo *directorio*.

```
CREATE DIRECTORY NombreAlias AS path de almacenamiento;
```

donde
NombreAlias es el alias dado al sistema, para el caso de éste sistema *DTU*;
path de almacenamiento es la ruta donde se almacenarán los archivos, *Ultratel.ultratel.com* para este caso, un servidor de la Empresa.

Luego se concede el privilegio de lectura al objeto *directorio* creado:

```
GRANT READ ON DIRECTORY NombreAlias TO user;
```

donde
NombreAlias es el alias dado al sistema, para el caso de éste sistema *DTU*;
user nombre del usuario o usuarios que tiene acceso al sistema. Para este caso el usuario tiene el mismo nombre, *DTU*.

Posteriormente, cuando ya se tienen definidas las tablas y los campos de contenido BFile, se deben utilizar ciertos paquetes especiales creados para su acceso y manipulación. El paquete Oracle DBMS_LOB proporciona un conjunto de procedimientos que hacen esto posible.

Para cargar en la base de datos los archivos de tipo BFile se requiere emplear un lenguaje que permita el manejo del paquete DBMS_LOB, por ejemplo el PL/SQL, haciendo uso de procedimientos creados para ello. Es necesario para la carga inicial de los BFILE un BLOB inicializado en vacío.

4.3 DIAGRAMA ENTIDAD – RELACIÓN

4.31 Descripción general de los Diagrama Entidad-Relación

El modelo entidad-relación es el modelo conceptual más utilizado para el diseño conceptual de bases de datos. Fue introducido por Peter Chen en 1976 y está formado por un conjunto de conceptos que permiten describir la realidad mediante un conjunto de representaciones gráficas y lingüísticas.

Sus características son:

- Reflejan tan sólo la existencia de los datos sin expresar lo que se hace con ellos.
- Es independiente de las bases de datos y de los sistemas operativos.
- Incluye todos los datos que se estudian sin tener en cuenta las aplicaciones que se van a tratar.

Sus elementos fundamentales son las entidades y las relaciones.

Una **entidad** caracteriza a un tipo de objeto, real o abstracto, del problema a modelar. Toda entidad tiene existencia propia, se distingue del resto de las entidades, tiene nombre y posee atributos definidos en un dominio determinado. Se declara una entidad como todo aquello de lo que se desea almacenar información. En el diagrama E-R las entidades se representan mediante rectángulos.

Hay dos tipos de entidades: fuertes y débiles. Una entidad fuerte es la que posee llame primaria y una entidad débil es una entidad cuya existencia depende de la existencia de otra entidad y no tiene llave primaria.

Una **relación** es una asociación o relación matemática entre varias entidades. Las relaciones también se nombran. Se representan en el diagrama E-R mediante flechas y rombos.

Las entidades que están involucradas en una determinada relación se denominan *entidades participantes*. El número de participantes en una relación es lo que se denomina *grado de la relación*. Por lo tanto, una relación en la que participan dos entidades es una *relación binaria*; si son tres las entidades participantes, la *relación es ternaria*; etc.

Una *relación recursiva* es una relación donde la misma entidad participa más de una vez en la relación con distintos papeles. El nombre de estos papeles es importante para determinar la función de cada participación.

La *cardinalidad* con la que una entidad participa en una relación especifica el número mínimo y el número máximo de correspondencias en las que puede tomar parte cada ocurrencia de dicha entidad. La participación de una entidad en una relación es *obligatoria (total)* si la existencia de cada una de sus ocurrencias requiere la existencia de, al menos, una ocurrencia de la otra entidad participante. Si no, la participación es *opcional (parcial)*. Las reglas que definen la cardinalidad de las relaciones son las reglas del negocio.

Los **atributos** representan las propiedades básicas o características de interés de las entidades y de las relaciones. Toda la información extensiva es portada por los atributos. Gráficamente, se representan mediante bolitas que cuelgan de las entidades o relaciones a las que pertenecen.

Cada atributo tiene un conjunto de valores asociados denominado *dominio*. El dominio define todos los valores posibles que puede tomar un atributo. Puede haber varios atributos definidos sobre un mismo dominio.

Los atributos pueden ser *simples o compuestos*. Un *atributo simple* es un atributo que tiene un solo componente, que no se puede dividir en partes más pequeñas que tengan un significado propio. Un *atributo compuesto* es un atributo con varios componentes, cada uno con un significado por sí mismo.

Un **identificador** de una entidad es un atributo o conjunto de atributos que determina de modo único cada ocurrencia de esa entidad. Un identificador de una entidad debe cumplir dos condiciones:

- No pueden existir dos ocurrencias de la entidad con el mismo valor del identificador.
- Si se omite cualquier atributo del identificador, la condición anterior deja de cumplirse.

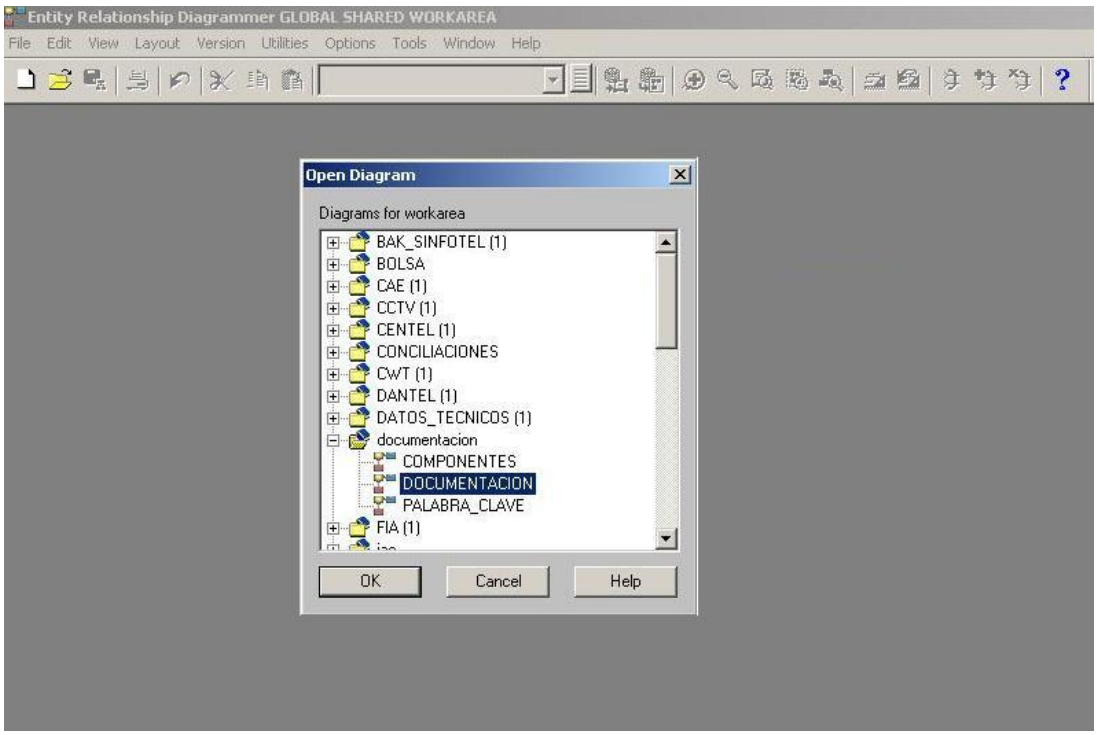
Toda entidad tiene al menos un identificador y puede tener varios identificadores alternativos. Las relaciones no tienen identificadores.

4.3.2 Diagrama Entidad-Relación desarrollado

En este desarrollo participaron los dos estudiantes en práctica que se encuentran trabajando en la Empresa y los ingenieros que se encuentran a cargo del proyecto ya que es una misma base de datos para los dos proyectos y su diseño tiene que satisfacer las necesidades de ambos sistemas.

Para el desarrollo del diagrama E-R se utilizó el *diagramador Entidad - Relación* (*Entity Relationship Diagrammer*) del módulo *modelador de requerimientos del sistema* de la herramienta case Oracle Designer (ver figura 6).

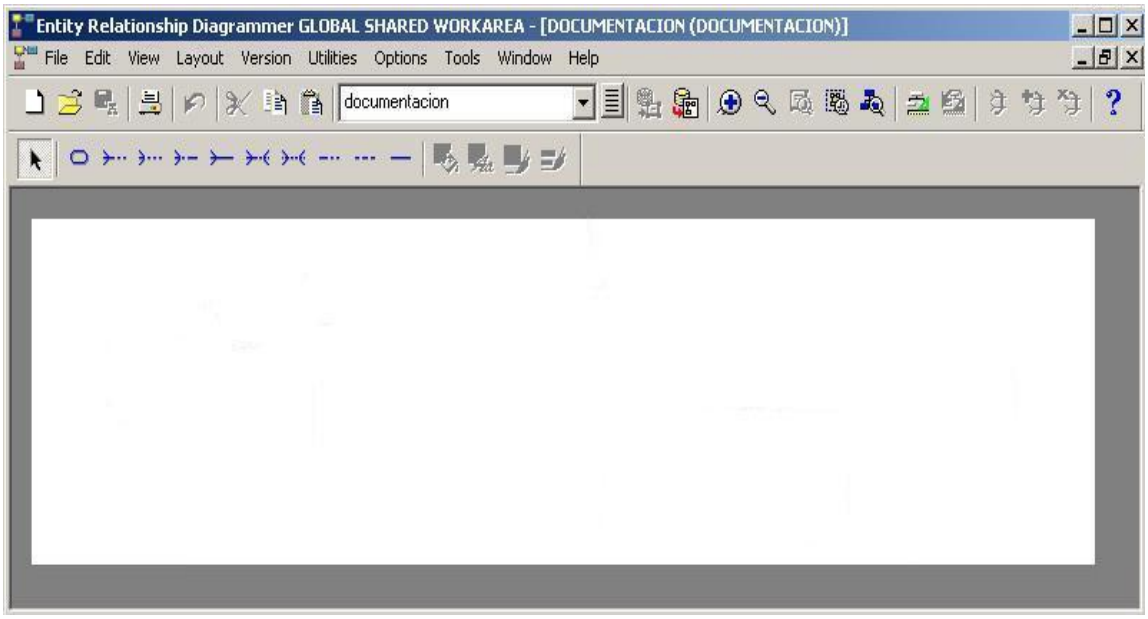
Figura 7. Diagramador Entidad – Relación



Una vez se ingresa a este diagramador se da la opción de *nuevo* para crear un diagrama o *abrir* cuando se desea modificar o seguir avanzando en un diagrama ya empezado a desarrollar

La siguiente imagen muestra como es la interfaz de trabajo en este diagramador.

Figura 8. Área de trabajo, Diagramador Entidad - Relación

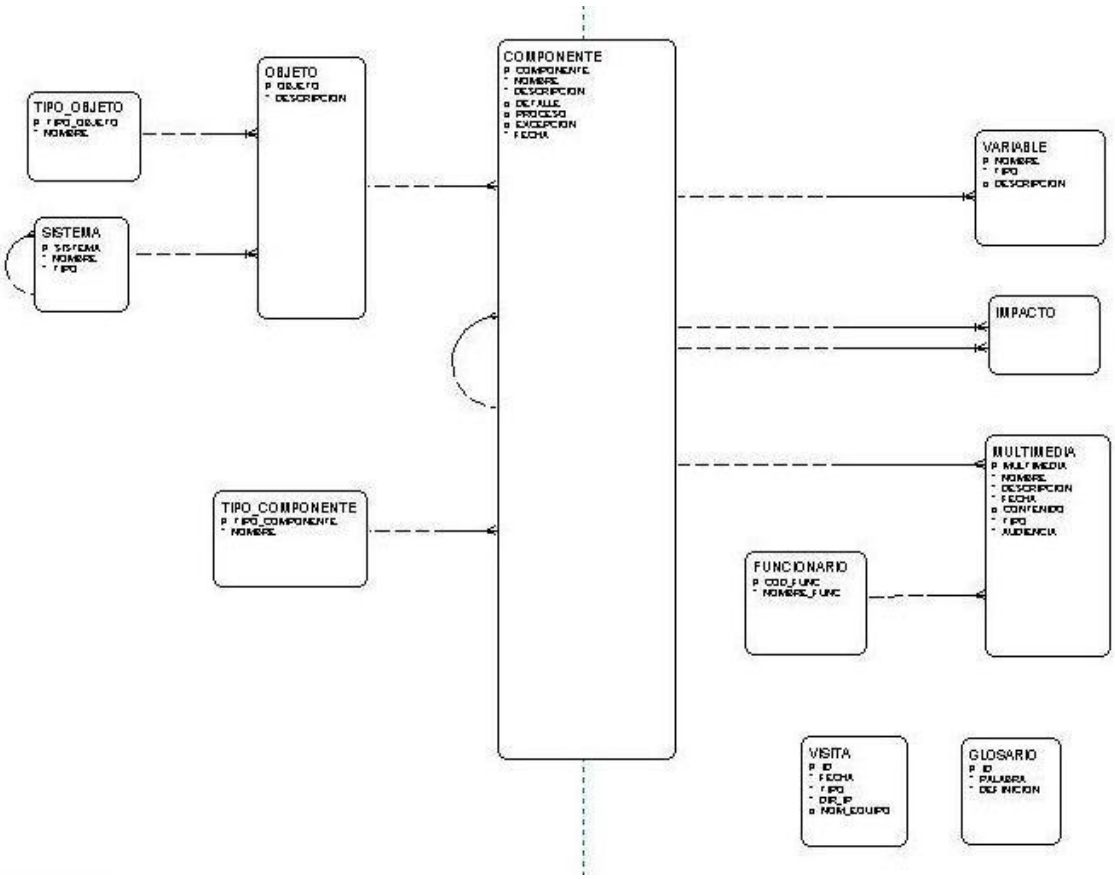


Se manejaron varios prototipos del diagrama Entidad-Relación antes de obtener el que más se ajustaba al sistema en desarrollo, validándose constantemente si tanto las entidades como sus atributos deben formar parte del diagrama.

El diagrama E-R final esta constituido por 10 entidades propias del esquema y una heredada de la base de datos general de la Empresa, aunque para el diagrama se muestra como propia.

A continuación se presenta el diagrama Entidad - Relación final.

Figura 9. Diagrama Entidad - Relación



Como se observa en el diagrama las relaciones entre las entidades *tipo_objeto* y *objeto* y las entidades *sistema* y *objeto* tienen una raya en la terminación hijo (entidad *objeto*) que indica que las llaves primarias de estas dos tablas también se convertirán en la llave primaria de esta, quedando con una sola llave primaria compuesta por tres columnas (las columnas *tipo_objeto*, *sistema* y *objeto*). Lo mismo se aprecia en las dos relaciones que llegan a la entidad débil *impacto* procedente de la entidad *componente* y de ella hasta la entidad *variable*.

4.3.2.1 Entidades

- Entidades fuertes o subordinantes

Sistema: encierra todos los sistemas de información que se manejan en la Empresa. La Empresa cuenta con seis grandes sistemas pero algunos de ellos se encuentran divididos en módulos. Esta distinción (sistema – módulo) se representa con la relación recursiva que posee ésta tabla.

Tipo_Objeto: se llamaron objetos a aquellos elementos que conforman un sistema, ya sea a nivel de usuario o a nivel técnico. Los objetos contenidos en un sistema, pueden ser pantallas, pantallas restringidas, reportes, programas, etc.

Objeto: es donde se almacenará el nombre y la descripción de los diferentes tipos de objeto que conforman los sistemas de información.

Tipo_Componente: se definieron componentes de una pantalla o reporte como íconos, menús, barras de herramientas, bloque de datos, etc., y procedimientos y funciones como componentes de un objeto de tipo programa.

Componente: la entidad componente almacena los datos básicos de un componente que se encuentran en una pantalla o reporte y adicionalmente el detalle, el proceso y las excepciones de un programa.

Multimedia: se almacenan todos los recursos con que cuenta un componente para su documentación. En esta entidad se definió un atributo llamado *contenido* con el tipo de dato BFile que acepta un tamaño de archivo hasta de 4 GB. Los archivos pueden ser de tipo imagen, sonido o video.

Funcionario: es la persona del área de sistema que realiza ingreso del archivo multimedia a la base de datos de documentación.

Variable: entidad que almacena las variables que interactúan en un programa. Pueden ser de entrada, salida o de entrada y salida.

Glosario: en esta entidad se encuentran todos los términos, con sus respectiva definición, usados frecuentemente en la Empresa.

Visita: guarda el registro del componente o archivo multimedia visitado, junto con la fecha de ingreso, dirección ip y nombre del equipo desde el cual se ingreso al sistema.

- Entidad débil o subordinada

Impacto: en esta entidad se muestra en donde y por quien es utilizado cierto procedimiento o función; cuales sistemas se ven afectados al momento de realizar una modificación al código de una función o procedimiento.

4.3.2.2 Relaciones. Todas las relaciones que se manejaron en el diagrama son de cardinalidad uno a muchos y de modalidad parcial u opcional.
SIS_SIS_FK: relación entre la entidad *sistema* y ella misma
OBJ_SIS_FK: relación entre las entidades *sistema* y *objeto*
OBJ_ TIP_OBJ_FK: relación entre las entidades *tipo_objeto* y *objeto*
COM_OBJ_FK: relación entre las entidades *objeto* y *componente*
COM_TIPCOM_FK: relación entre las entidades *tipo_componente* y *componente*

COM_COM_FK: relación entre la entidad *componente* y ella misma
VAR_COM_FK: relación entre las entidades *componente* y *variable*
IMP_COM_fk y IMP_COM_Q_FK: relaciones entre la entidad fuerte *componente* y la débil impacto
MUL_COM_FK: relación entre las entidades *componente* y *multimedia*
MUL_FUN_FK: relación entre las entidades *funcionario* y *multimedia*

4.4 MODELO DE DATOS

Al inicio de esta etapa se trabajó con el *transformador del diseño de la base de datos* (*Database Design Transformer*) que se encuentra en el *generador de diseño preliminar* (ver figura 6) para convertir el diagrama E-R al modelo de datos.

Los pasos a seguir son los siguientes: primero se selecciona el contenedor del diagrama que se desea convertir y se inicia el proceso:

Figura 10. Selección del diagrama a convertir

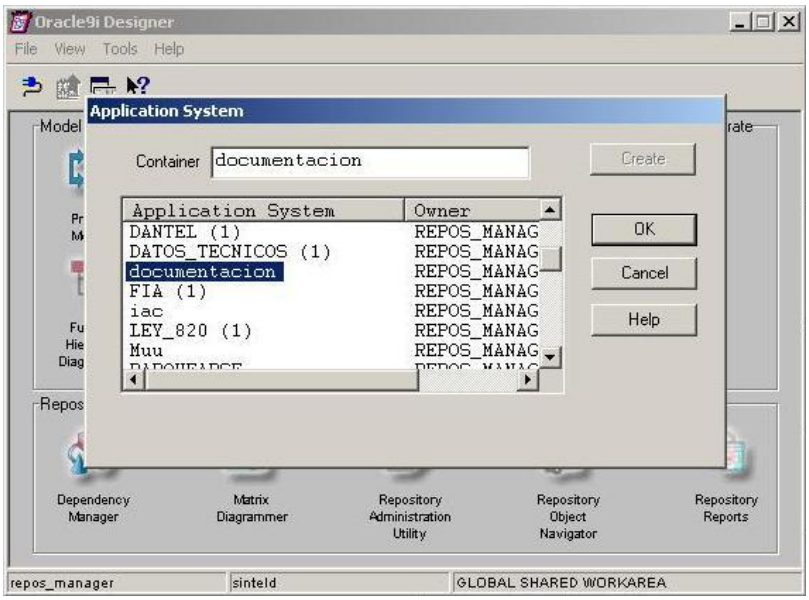
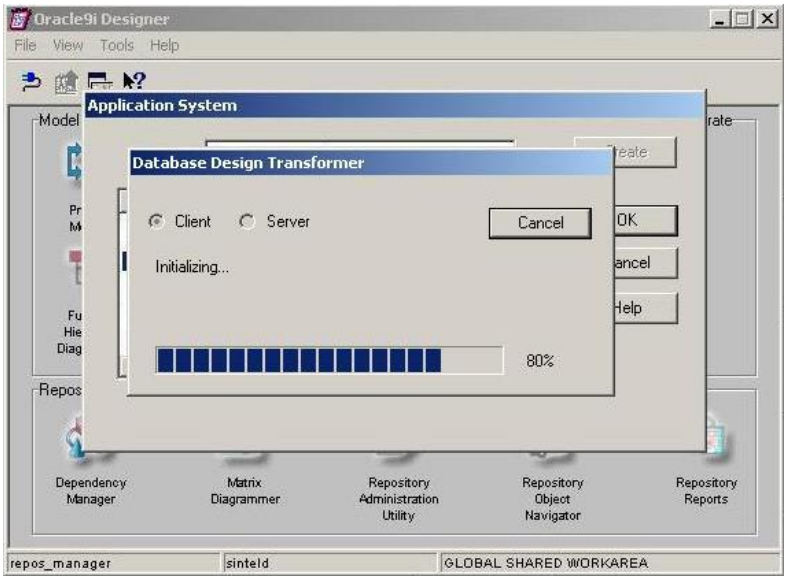
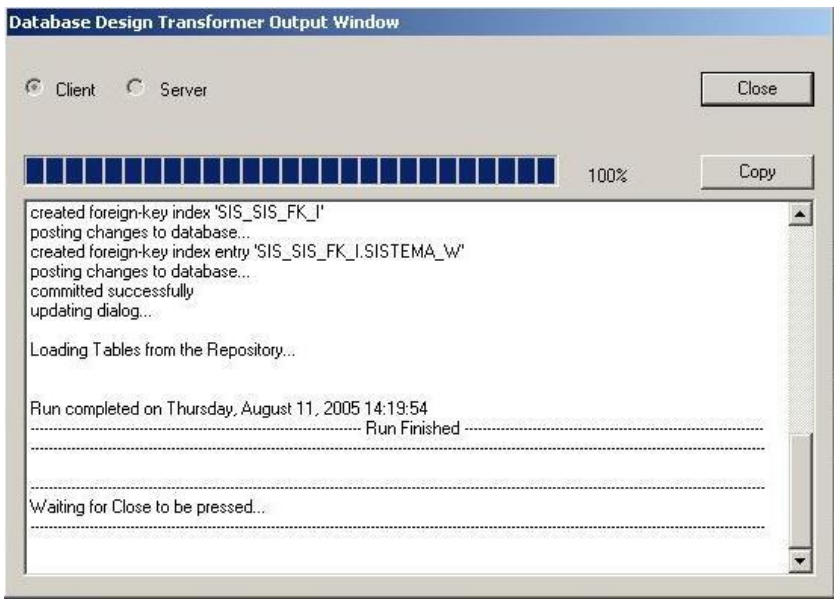


Figura 11. Proceso de transformación



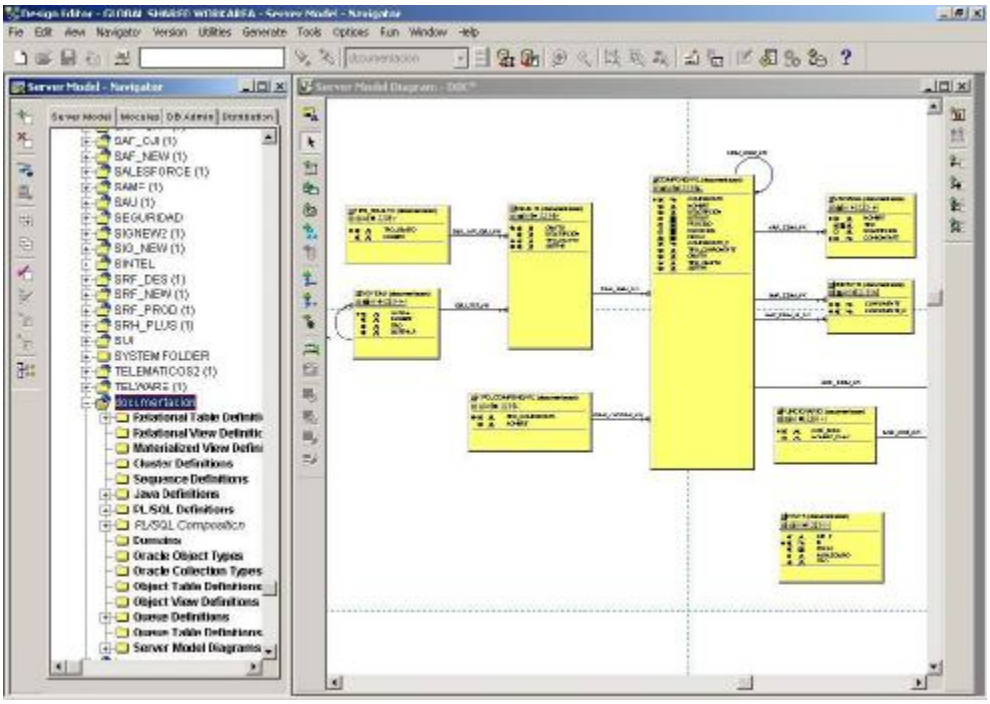
En la pantalla que aparece posteriormente (paso intermedio) solo es necesario dar clic en el botón *RUN* para continuar con el proceso de la transformación.

Figura 12. Fin del proceso de transformación



Una vez terminado el proceso de transformación del diagrama E-R se utilizó el *editor de diseño* (*Design Editor*) que se encuentra en la utilidad de *diseño y generación* (*Design and Generate*) para concluir con el proceso de diseño de las tablas que conformarán la base de datos de documentación.

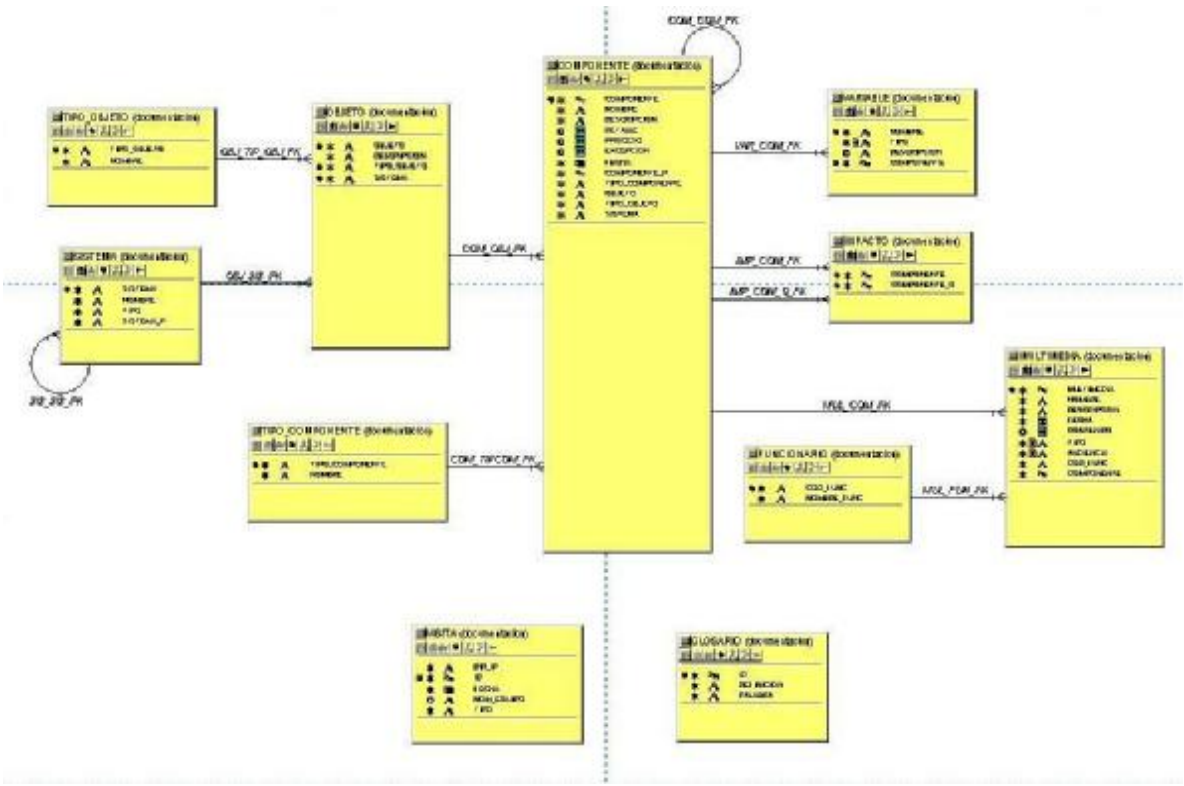
Figura 13. Editor de diseño



En el modelo de datos se modificaron las entidades con atributos de tipo BFile ya que en el proceso de conversión se crearon como tipo varchar2. De igual forma se verificaron todos los atributos de las demás tablas comprobando que no hayan sufrido cambios en su estructura.

El siguiente es el modelo de datos final.

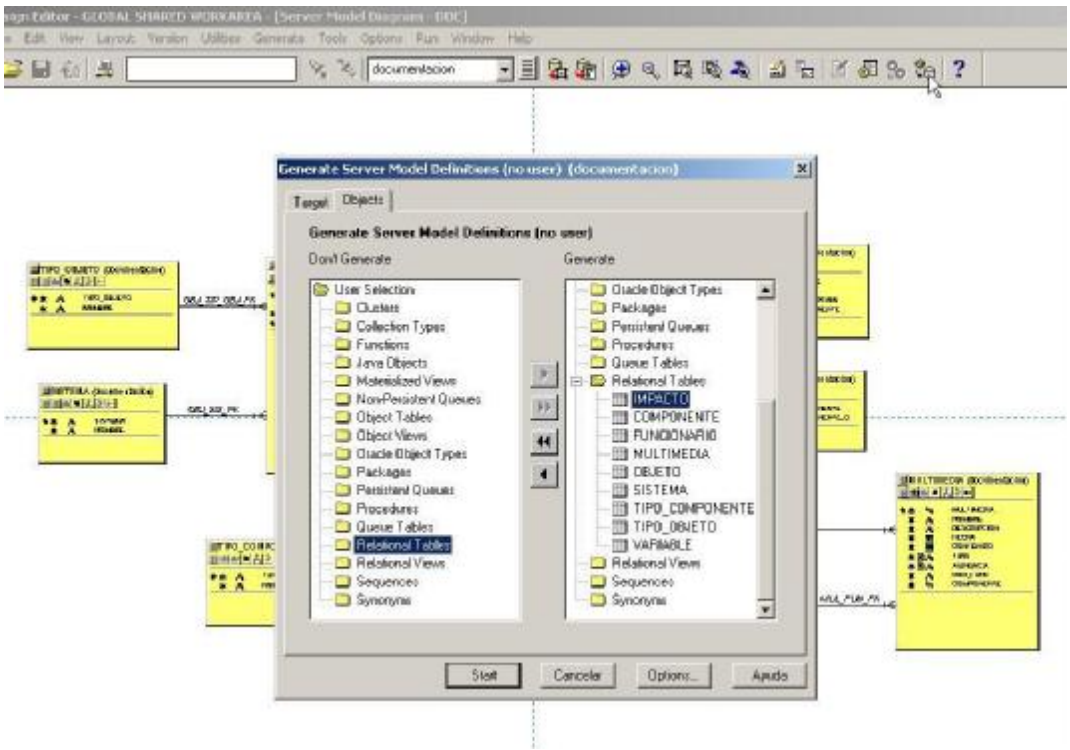
Figura 14. Modelo de Datos



4.5 GENERACIÓN DE TABLAS

Las tablas de la base de datos se crean a partir de la ejecución de archivos *DLL* generados al seleccionar la opción *Generate DDL* que se encuentra en la barra de herramientas del *editor de diseño*.

Figura 15. Generación de archivos ddl



Se generan cuatro archivos con el mismo nombre, *cdsddl* pero con extensiones diferentes: *tab*, *ind*, *con* y *sql* el cual contiene el siguiente código:

```
SPOOL cdsddl.lst
@@cdsddl.tab
@@cdsddl.ind
@@cdsddl.con
SPOOL OFF
```

y una vez ejecutado en la herramienta apropiada genera las tablas, los índices, las llaves primarias, los constrains y las llaves foráneas de la nueva base de datos.

4.6 TABLAS DE LA BASE DE DATOS

Los archivos *DLL* que generan físicamente las tablas de la base de datos se ejecutaron en la herramienta Sql Navigator y desde allí, provisionalmente, las tablas serán manipuladas mediante sentencias SQL.

Las tablas generadas son: sistema, tipo_objeto, objeto, tipo_componente, componente, impacto, variable, multimedia, glosario y visita. Para cada una se crearon las columnas, los índices y los constraints respectivos. La tabla funcionario fue tomada del esquema general de la Empresa y no fue creada. A continuación se muestra en detalle los campos de cada una de las tablas.

4.6.1 SISTEMA

La tabla sistema tiene tres campos originales y uno formado por la relación recursiva que tiene: *sistema* que son las tres letras que lo identifican (llave primaria), *nombre* del sistema, *tipo* de sistema, “S” para sistema y “M” para módulo y *sistema_p* que es la llave del sistema al cual pertenece el módulo.

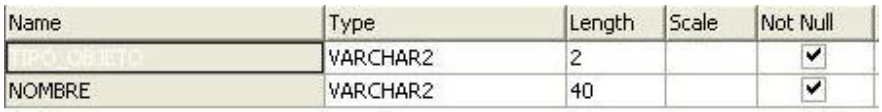
Figura 16. Tabla Sistema

Name	Type	Length	Scale	Not Null
SISTEMA	VARCHAR2	3		☑
NOMBRE	VARCHAR2	60		☑
TIPO	VARCHAR2	1		☑
SISTEMA_P	VARCHAR2	3		☐

4.6.2 TIPO_OBJETO

La tabla esta compuesta por: *tipo_objeto* que es un número consecutivo de identificación y *nombre* que es el nombre dado a dicho objeto (por ejemplo pantalla, pantalla privada, reporte, programa, etc).

Figura 17. Tabla Tipo_Objeto

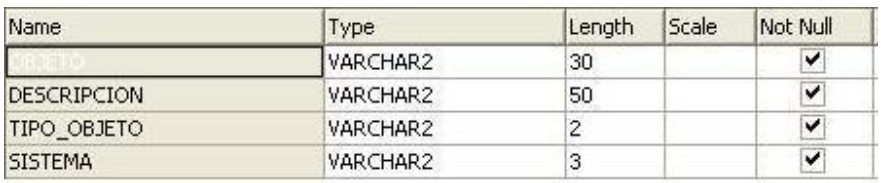


Name	Type	Length	Scale	Not Null
TIPO_OBJETO	VARCHAR2	2		✔
NOMBRE	VARCHAR2	40		✔

4.6.3 OBJETO

La tabla objeto esta formada por *objeto* que es el nombre como esta guardada la pantalla, el reporte o el programa (llave primaria), *descripción* que corresponde al nombre con el que se le conoce o como esta en el sistema y los campos *tipo_objeto* y *sistema* que vienen de las relaciones con estas tablas y que también son llaves primarias.

Figura 18. Tabla Objeto

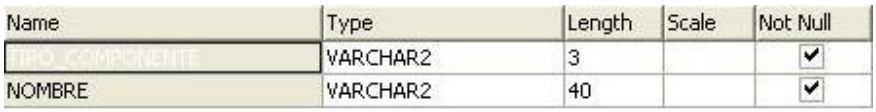


Name	Type	Length	Scale	Not Null
OBJETO	VARCHAR2	30		✔
DESCRIPCION	VARCHAR2	50		✔
TIPO_OBJETO	VARCHAR2	2		✔
SISTEMA	VARCHAR2	3		✔

4.6.4 TIPO_COMPONENTE

Esta tabla tiene los siguientes campos: *tipo_componente* que es el id y *nombre* que declara o describe el tipo de componente (por ejemplo icono, barra de herramientas, bloque de texto, etc.).

Figura 19. Tabla Tipo_Componente



Name	Type	Length	Scale	Not Null
TIPO_COMPONENTE	VARCHAR2	3		☑
NOMBRE	VARCHAR2	40		☑

4.6.5 COMPONENTE

La tabla componente guarda la información básica de todos los componentes que pertenecen a un objeto: *componente* o id, *nombre*, *descripción* y *fecha* de ingreso. Los campos *detalle*, *proceso* y *excepción* que son exclusivos de los objeto de tipo *programa*. El resto de campos, *componente_p*, *tipo_componente*, *objeto*, *tipo_objeto* y *sistema* son las llaves foráneas que vienen de las tablas relacionadas.

Figura 20. Tabla Componente

Name	Type	Length	Scale	Not Null
COMPONENTE	NUMBER	10	0	☑
NOMBRE	VARCHAR2	60		☑
DESCRIPCION	VARCHAR2	400		☑
DETALLE	BFILE	530		☐
PROCESO	BFILE	530		☐
EXCEPCION	BFILE	530		☐
FECHA	DATE	7		☐
COMPONENTE_P	NUMBER	10	0	☐
TIPO_COMPONENTE	VARCHAR2	3		☑
OBJETO	VARCHAR2	30		☑
TIPO_OBJETO	VARCHAR2	2		☑
SISTEMA	VARCHAR2	3		☑

4.6.6 IMPACTO

Esta tabla se forma por las dos relaciones que tiene con la tabla *componente*. En ella se guarda el id del componente (*componente*) que se está analizando junto con el id de los componentes (*componente_q*) que se ven afectados por él.

Figura 21. Tabla Impacto

Name	Type	Length	Scale	Not Null
COMPONENTE	NUMBER	10	0	✔
COMPONENTE_Q	NUMBER	10	0	✔

4.6.7 VARIABLE

La tabla variable tiene una llave primaria formada por dos columnas, *nombre* y *componente* (dato que viene de la relación). Los otros campos son: *tipo* que dice si la variable es de entrada, salida o de entrada y salida y la *descripción* que explica brevemente su funcionamiento.

Figura 22. Tabla Variable

Name	Type	Length	Scale	Not Null
NOMBRE	VARCHAR2	30		✔
TIPO	VARCHAR2	1		✔
DESCRIPCION	VARCHAR2	200		✔
COMPONENTE	NUMBER	10	0	✔

4.6.8 MULTIMEDIA

La principal tabla del sistema se compone por los datos básicos del archivo: id (*multimedia*), *nombre*, *descripción*, *fecha* de creación o modificación y *tipo*, si es imagen “I”, video “V” o sonido “S”. El campo *contenido* es donde se guarda el archivo multimedia, *audiencia* determina la clase de usuario que puede ver el archivo, “U” usuario final o “D” para desarrolladores. Los campos *cod_func* y *componente* vienen de las relaciones con las tablas funcionario y componente respectivamente.

Figura 23. Tabla Multimedia

Name	Type	Length	Scale	Not Null
MULTIMEDIA	NUMBER	10	0	☑
NOMBRE	VARCHAR2	40		☑
DESCRIPCION	VARCHAR2	400		☑
FECHA	DATE	7		☑
CONTENIDO	BFILE	530		☐
TIPO	VARCHAR2	1		☑
AUDIENCIA	VARCHAR2	1		☑
COD_FUNC	VARCHAR2	3		☑
COMPONENTE	NUMBER	10	0	☑

4.6.9 FUNCIONARIO

Esta tabla es tomada de la base de datos general de la Empresa. Los dos únicos campos con que interactúan el sistema son: *cod_func* que determina el código del funcionario y corresponde a tres letras iniciales del nombre y apellidos y *nom_func* que guarda el nombre, los demás datos son ignorados por este sistema.

Figura 24. Tabla Funcionario

Name	Type	Length	Scale	Not Null
COD_FUNC	VARCHAR2	3		☑
NOMBRE_FUNC	VARCHAR2	25		☑
COD_DEP	NUMBER	3	0	☑
COD_CARGO	NUMBER	4	0	☑
COD_FUNC_GRUPO	VARCHAR2	3		☐
USR_ULT_MODIF	VARCHAR2	3		☐
FECHA_ULT_MODIF	DATE	7		☐
CEDULA	VARCHAR2	9		☐
FECHA_VENCIMIENTO	DATE	7		☐
CORREO_ELECTRONICO	VARCHAR2	60		☐

4.6.10 GLOSARIO

Esta tabla no se encuentra relacionada con ningún de las demás tablas de la base de datos. Se compone por un *id* o identificador, *palabra* para nombrar el término a describir y *definición* que guarda el significado.

Figura 25. Tabla Glosario

Name	Type	Length	Scale	Not Null
	NUMBER	10	0	☑
PALABRA	VARCHAR2	60		☑
DEFINICION	VARCHAR2	400		☑

4.6.11 VISITA

Esta tabla tampoco esta relacionada con ninguna de las demás tablas de la base de datos. Registra el ingreso o visita a un componentes y a un archivo multimedia. Guarda el *id* del componente o del multimedia visitado, la *fecha* de ingreso, el *tipo*, “C” para componente y “M” para multimedia, la dirección ip del equipo (*dir_ip*) y el nombre del equipo desde el cual se hizo el ingreso (*nom_equipo*).

Figura 26. Tabla Visita

Name	Type	Length	Scale	Not Null
	NUMBER	10	0	☑
FECHA	DATE	7		☑
TIPO	VARCHAR2	1		☑
DIR_IP	VARCHAR2	20		☑
NOM_EQUIPO	VARCHAR2	20		☐

4.7 VISTAS CREADAS

Las vistas tienen la misma estructura que una tabla, filas y columnas, la única diferencia es que sólo se almacena de ellas la definición, no los datos. Los datos que se recuperan mediante una consulta a una vista se presentarán igual que los de una tabla, de hecho, si no se sabe que se está trabajando con una vista, nada hace suponer que es así.

Para este sistema se crearon dos vista, *búsqueda* e *impacto_r* que se utilizan en los procedimientos *resultado* y *mostrar_imp* respectivamente.

4.7.1 Vista Busqueda

Esta vista toma de las tablas *sistema, tipo_objeto, tipo_componente* y *componente* el nombre del sistema, el nombre del tipo de objeto, el nombre del tipo de componente y el id y el nombre del componente respectivamente de cada una de las tablas. Se utiliza en el procedimiento *resultado* del paquete *buscar* y con estos datos compara las restricciones de la búsqueda.

La sentencia SQL utilizada para crear la vista fue:

```
CREATE OR REPLACE VIEW BUSQUEDA
(NOM_SIS, NOM_TPO, NOM_TPC, NOM_COM, ID_COMP) AS
select s.nombre, tpo.nombre, tpc.nombre, c.nombre, c.componente
from sistema s, tipo_objeto tpo, tipo_componente tpc, componente c
where c.sistema = s.sistema
and   c.tipo_objeto = tpo.tipo_objeto
and   c.tipo_componente = tpc.tipo_componente
```

4.7.2 Vista Impacto_r

Esta vista fue creada para tomar de las relaciones entre las tablas *componente* e *impacto* el nombre del componente que es impactado por otro ya que la relación es solo por el identificador del componente.

La vista se creó de la siguiente forma:

```
CREATE OR REPLACE VIEW IMPACTO_R
(NOMBRE_I, COMPONENTE_Q_I, COMPONENTE_I) AS
select c.nombre, i.componente_q,c.componente
from componente c, impacto i
where i.componente =c.componente
```

y se utiliza en el procedimiento *ver_afectado* del paquete *mostrar_imp*.

5. PORTAL WEB DE DOCUMENTACIÓN

Se tomó la opción de construir un portal web para mostrar a los usuarios la documentación de los sistemas y no un medio impreso por el acercamiento existente entre el sitio donde realizan su trabajo y donde puede resolver sus dudas, es decir, los usuarios la mayor parte del tiempo se encuentran frente a un computador desarrollando sus actividades, el mejor sitio para buscar soluciones a sus inquietudes es ahí mismo, ingresando al portal web alojado en la Intranet empresarial o desde los mismos sistemas sin tener que movilizarse a otro lugar perdiendo tiempo buscando en grandes libros lo que necesitan.

5.1 CONSTRUCCIÓN DE PORTALES WEB

Un portal web es una aplicación que gestiona de forma uniforme y centralizada, contenidos provenientes de diversas fuentes, implementa mecanismos de navegación sobre los contenidos, integra aplicaciones e incluye mecanismos de colaboración para el conjunto de usuarios a los que sirve de marco de trabajo, todo en un entorno web.

Los portales web deben diseñarse para usuarios determinados y sus objetivos más probables. Diseñar intentando abarcar a todos los usuarios y todos sus posibles objetivos significa crear diseños poco usables que no satisfacen a nadie.

A continuación se listan unas pautas seguidas en el diseño del portal web de documentación.

5.1.1 Cómo diseñar un portal web

Existen varias recomendaciones sobre lo que no se debe hacer a la hora de diseñar un portal web:

- *Fondos recargados:* es mejor que el fondo de las páginas sea blanco o de un color claro y neutro que facilite la lectura. No es recomendable utilizar una imagen de fondo, pues solo va a distraer a los visitantes y quizás imposibilitar la lectura del texto.

- *Sobrecarga de gráficos:* sólo hay que utilizar elementos gráficos cuando aporten algo a la página y es preferible que sea sencillo y ocupe poco espacio., esto hará que la página se cargue más rápido.

- *Parcelar con marcos:* los marcos (frames) pueden ser útiles para mantener información siempre visible en la página, pero inducen a errores y dificultan la navegación. Es mejor utilizar tablas en su lugar, y si no es posible, que no sean más de 3 marcos.

- *Modelo papiro:* crear páginas lineales produce un pérdida de espacio. Es mejor utilizar un modelo de página en dos columnas. Esto se puede conseguir con la utilización de tablas, que aunque no es la mejor solución, es la única sencilla.

- *Contadores de visitas:* un contador de visitas no aporta información importante sobre los visitantes. Es mejor que se utilicen herramientas más sofisticadas y menos agresivas para analizar el diario de visitas de las páginas.

- *En Construcción:* es mejor que un enlace que conduce a una página en blanco con el letrero “en construcción” no exista ya que es muy desagradable para los visitantes frenarse con este tipo de páginas.

5.1.2 Principios básicos de navegación

No importa lo bien diseñado que haya quedado un portal web o lo útil que sea la información que en él se ofrece, si no tiene un esquema de navegación prudente, confundirá a los visitantes y los echará del mismo. La buena navegación es un problema de sentido común y existen unos principios básicos por aplicar.

Página principal: la primera página es lo primero que el visitante ve y debe ser suficientemente atractiva. Una pagina principal debe tener las siguientes características:

- Un resumen de lo que está disponible en el portal, cada sección a la que se puede acceder desde la página principal ya sea directamente o a dos o tres clic de distancia.
- Debe resultar atractiva y proyectar una imagen correcta de la Empresa.
- Debe reforzar la imagen de la marca o del producto, de modo que ésta sea fácilmente identificable.
- Debe compartir ciertos elementos con el resto de las páginas del portal, de forma que se de una imagen corporativa del sitio.
- Una página principal se asemeja a la tabla de contenido de un libro o de una revista.

Esquema Piramidal: la mayoría de portales siguen una organización jerárquica con forma de triángulo, en donde el vértice está ocupado por la página principal y las páginas en la base son las que tienen un nivel de detalle menor. Las jerarquías deben ser de tal forma que los lados del triángulo sean similares. Normalmente los portales pequeños tienen de dos a tres niveles.

Longitud de las páginas: no existe un único criterio a la hora de establecer la longitud idónea de las páginas. Las que son demasiado largas para caber en la ventana del navegador obligan a utilizar el scroll para acceder a toda su información. En cambio, la división de un documento en varias páginas facilita su

lectura pero puede ralentizar el acceso a la información pues hay que esperar a que se carguen varias páginas en lugar de una sola.

En cualquier caso, si se está construyendo una página larga, hay que colocar enlaces desde el principio a las distintas secciones y viceversa para facilitar la navegación dentro de ella.

Barras de Navegación: una barra de navegación es un elemento que indica claramente la ubicación dentro del portal. Debe tener la misma apariencia en todas las páginas y debe estar siempre disponible. Por ejemplo: *el sitio* > *docencia* > *asignaturas* > *la asignatura*. Cada uno de los elementos serán un link, de forma que además de indicar claramente dónde se está en cada momento se facilite la navegación.

Mapa del Sitio: especialmente si se trata de un portal complejo, un mapa del mismo puede resultar una excelente herramienta para localizar una página concreta. Un mapa del sitio es una enumeración estrictamente jerárquica de todas las páginas del mismo. Todas los elementos del mapa deben ser un link para facilitar la navegación.

5.2 SQL NAVIGATOR

Para el desarrollo del portal web inicialmente se había pensado en la herramienta Oracle Portal la cual presta todas las funcionalidades necesarias para la creación de portales con base de datos Oracle. Después de estudiarla más a fondo y de tratarla de configurar se vio que no era tan fácil de utilizar, sumado a que los ingenieros de la Empresa desconocen el funcionamiento de la misma se optó por buscar otra alternativa.

Teniendo en cuenta que en desarrollos similares (aunque apenas se han realizado dos aplicaciones web) se ha utilizado el Cartridge de PL/SQL se tomó esta opción como la más viable para realizar el portal web de documentación.

La herramienta elegida para realizar este desarrollo fue Sql Navigator ya que proporciona las herramientas necesarias para reducir el desarrollo de código en el lenguaje PL/SQL y lo logra utilizando una interfaz gráfica 'plug-and-play' que permite acelerar el desarrollo de aplicaciones basadas en Oracle, combinando la codificación, depuración, desarrollo web y control de versiones, proporcionando de esta manera aplicaciones de gran calidad.

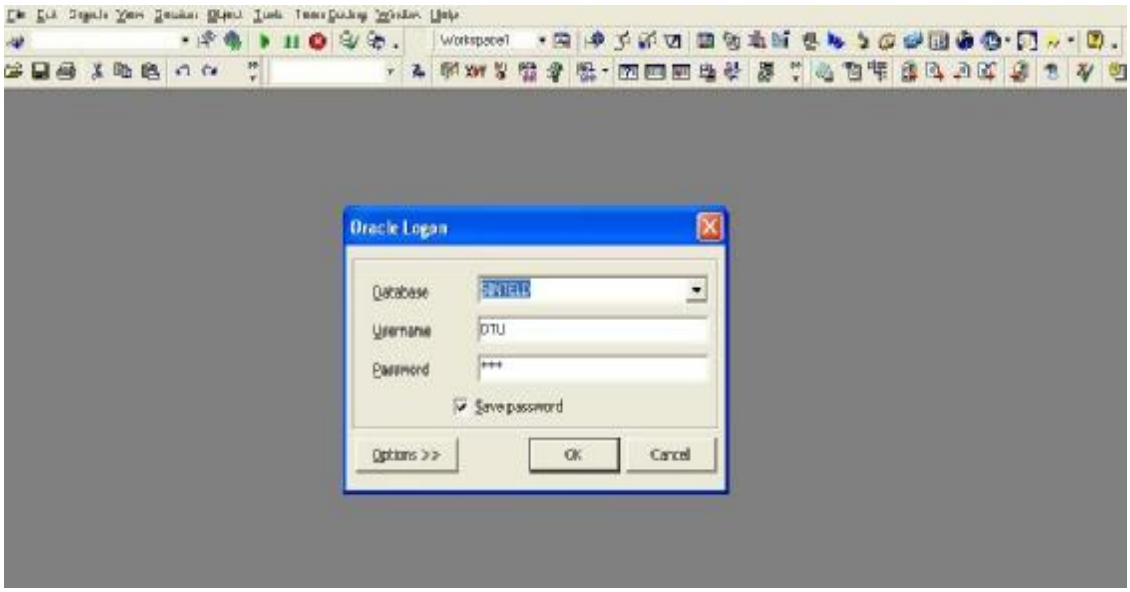
Los beneficios de esta herramienta son:

- Asegura óptima productividad y rendimiento de las aplicaciones.
- Da soporte para la integración con el sistemas de control de versiones.
- Perfecciona el desarrollo de bases de datos con la habilidad para 'cortar y pegar' código Sql.
- Proporciona soporte para bases de datos desarrolladas en Oracle 8i, 9i y JAVA.

5.2.1 Explicación básica de la herramienta

Para empezar a trabajar en la herramienta Sql Navigator es necesario conocer y colocar en el cuadro de diálogo inicial el nombre de la base de datos donde se esta trabajando, el nombre de usuario y la clave de acceso al esquema del sistema en desarrollo.

Figura 27. Inicio de la herramienta Sql Navigator

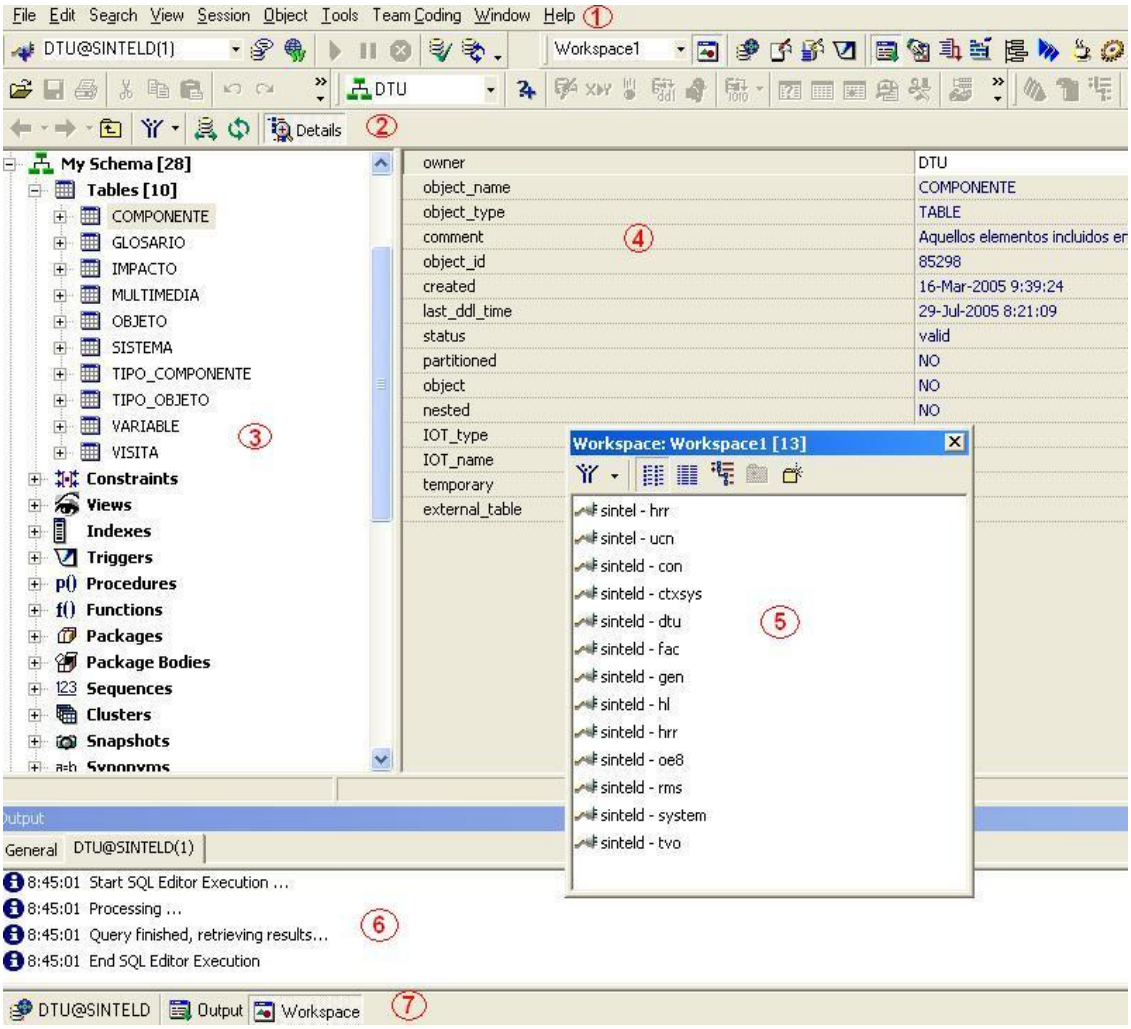


La interfaz de la herramienta esta formada principalmente por:

- Barra de menú principal.
- Barras de herramientas que pueden flotar, colocarse fijas o esconderse. Los botones de barra de herramientas se hacen activos según el elemento seleccionado.
- La estructura de la base de datos se muestra como un árbol con nodos extensibles.
- Un panel de detalles que muestra la información esencial del elemento en el árbol seleccionado.
- Una ventana de zona de trabajo que da acceso rápido a objetos usados con frecuencia.
- Una ventana de salida que muestra mensajes de la herramienta y del servidor e incluso muestra errores de Oracle.
- Una barra de tareas que al hacer clic en cualquier elemento hace que se haga visible en un nivel superior.

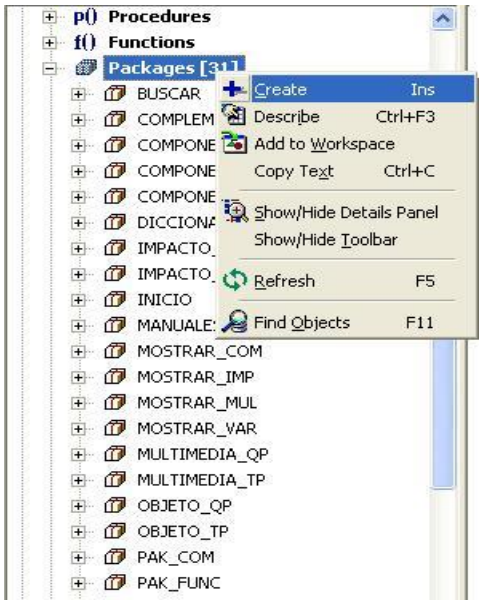
La siguiente figura muestra cada una de las partes anteriormente mencionadas.

Figura 28. Interfaz de la herramienta Sql Navigator



Cuando se tienen las tablas de la base de datos creadas físicamente se desarrollan procedimientos y funciones agrupadas en paquetes para crear aplicaciones basadas en ellas. Un paquete se crea (utilizando Sql Navigator) de la siguiente forma: clic derecho, la opción *create* o con la tecla *insert*.

Figura 29. Creación de paquetes con Sql Navigator



Como se habla en el Anexo F un paquete puede contener procedimientos, funciones, la declaración de tipos de datos, cursores, etc. Para compilar un paquete y conocer si tiene o no errores se puede pulsar las teclas *ctrl + s* o en el respectivo icono de la barra de herramientas:

Figura 30. Compilación de un paquete con Sql Navigator

En el caso de existir errores se sombrea la línea donde se encuentra y se presenta el código del error junto con algunas posibles soluciones.

5.3 PAQUETES DEL PORTAL WEB

Para la creación de la interfaz de consulta de datos (portal web) fue necesario crear paquetes, procedimientos y funciones utilizando las herramientas para el desarrollo de aplicaciones web de PL/SQL (Cartridge de PL/SQL). Los paquetes se dividieron de tal forma que su organización fuera ordenada y fácil de entender y que todos los procedimientos o funciones declarados en cada uno de ellos guardaran cierta relación.

Se desarrollaron 31 paquetes en total pero algunos fueron creados por un Generador de Paquetes que crea, por cada tabla de la base de datos, tres tipos de paquetes con funciones reutilizables en el otros procedimientos o funciones.

5.3.1 Generador de paquetes

El generador de paquetes es un código que en su versión 1.0 se encuentra en prueba por la Empresa. Su uso ya ha sido aprobado por los ingenieros de ésta área por tal motivo su aplicación en este proyecto constituye una prueba más del mismo.

Con el generador se crean tres tipos de paquetes los cuales tienen el mismo nombre de la tabla para la cual se crean pero seguido de dos letras específicas que los diferencia:

- NomTabla_TP (type package): este paquete crea los tipos de datos necesarios para la ejecución de procedimientos y funciones.

- NomTabla_QP (query package): en este paquete se crean funciones que permiten hacer consultas sobre la tabla para el cual fue creado. Pueden utilizarse en cualquier otro paquete haciendo más fácil la escritura de código

Sql y de esta forma disminuyendo la complejidad y la cantidad de líneas de código.

- NomTabla_XP (extra package): este paquete es reservado para aquellas funciones propias de las políticas del negocio. Estas funciones son creadas por el ingeniero desarrollador dependiendo sus necesidades y las del negocio.

Algunas de las funciones creadas en el paquete QP son:

- ALLROWS: muestra todos los registros de la tabla.
- CNT: cuenta los registros de la tabla.
- EXISTE: retorna la existencia de un registro que cumpla con la condición dada.
- ROWEXISTS: devuelve un booleano que determina la existencia de un registro.
- ONEROW: devuelve los registros de una tabla uno por uno.

También se crean funciones por la relación existente entre las tablas. Se crean cuatro funciones que tiene por nombre: el nombre la tabla hijo seguido del nombre de la tabla padre y la expresión propia de la función:

- _FK: devuelve los registros que cumplan con la condición de join entre las tablas.
- _FK_CNT: cuenta los registro que cumplen con la condición de join entre las tablas.
- _FK_CV: devuelve un cursor con los registros de la tabla que cumplen la condición de join.
- _FK_EX: retorna los valores que además de cumplir con la condición de join cumplen también una cláusula WHERE.

Haciendo uso del Generador de Paquetes se crearon los paquetes TP y QP para algunas de las tablas de la base de datos y para la tabla *componente* se creó el

paquete XP. Las funciones más utilizadas son *ONEROW*, *CNT* y la función creada para contar los registros que cumplen con la condición de join entre dos tablas relacionadas, *_FK_CNT*.

5.3.2 Paquetes desarrollados

Como ya se dijo fue necesario el desarrollo de paquetes que permiten extraer la información de la base de datos y la muestra por pantalla de forma práctica y agradable para los usuarios de los sistemas de información.

Los paquetes desarrollados para la creación de la interfaz del portal web se dividen según las funcionalidades del sistema o la tabla que interactúa en él.

5.3.2.1 Paquete Inicio. Paquete que contiene los procedimientos que permiten visualizar la pantalla principal del portal web.

5.3.2.1.1 Procedimiento división. Este procedimiento divide la pantalla en tres frames. Se utiliza la opción del frame para hacer la página más didáctica y funcional ya que, por ejemplo, no se tiene que repetir las líneas de código del logo del sistema sino que se coloca en un frame permanente.

5.3.2.1.2 Procedimiento hora. Este procedimiento es un javascript que muestra la hora dentro de una caja de texto. El cartridge de PL/SQL permite tener código java embebido y obtener funcionalidades que con el lenguaje PL/SQL no se pueden lograr.

5.3.2.1.3 Procedimiento frame1. Este frame incluye las opciones de ingreso y cierta información que puede ser de interés de los usuarios del sistema. Cuenta con un icono para regresar a la pantalla principal, información sobre el día, la

fecha y la hora actual, un buscador sencillo, las opciones de ingreso a los manuales y al glosario y un javascript de scroll de noticias.

5.3.2.1.4 Procedimiento frame2. Con este procedimiento se coloca el logo del sistema de información desarrollado el cual encabeza todas las páginas consultadas.

5.3.2.1.5 Procedimiento frame3. En el procedimiento del frame principal inicialmente aparece el título del sistema, un texto de bienvenida y cierta información importante la cual puede ser reemplazada cuando sea necesario. Después de acceder a cualquier opción cambia según lo seleccionado.

5.3.2.1.6 Procedimiento texto_js. Como ya se dijo el cartridge de PL/SQL permite tener código java embebido. Este scroll de noticias fue creado para incluir la información que los ingenieros desarrolladores consideren importante para los usuarios finales del sistema como por ejemplo el ingreso de nuevas ayudas multimedia o algún cambio en el funcionamiento de un sistema. Se tomó esta opción de varias páginas web que lo utilizan para diferentes fines (por ejemplo la página de la Biblioteca Nacional, http://www.bibliotecanacional.gov.co)

Se utilizó javascript que es un lenguaje script orientado a documentos para agregarle características diferentes y con movimiento al portal web. Javascript es una de las múltiples aplicaciones que han surgido para extender las capacidades del Lenguaje HTML y de paso el cartridge de PL/SQL.

5.3.2.2 Paquete Buscar. Paquete donde se encuentran los procedimientos que realizan la búsqueda sencilla o la búsqueda avanzada de componentes al igual que el procedimiento para mostrar los resultados de dicha búsqueda.

En este paquete y específicamente para el procedimiento *resultado* fue necesario declarar un cursor y un tipo de datos que permiten mostrar el contenido de la tabla *componente* y el link desde el cual se puede acceder.

Para el desarrollo de los buscadores se intentó configurar la herramienta Oracle Text que ofrece una completa solución para la realización de búsquedas textuales.

Oracle Text puede filtrar y extraer contenido de formatos tan ampliamente utilizados como Adobe pdf, Microsoft Office, HTML y XML, entre otros. También permite incorporar características de búsquedas específicas (por ejemplo la configuración de tesauros) en lenguajes como español, inglés, alemán, francés, etc. Puede realizar búsqueda sobre partes específicas de los documentos o columnas de la base de datos, diferenciando mayúsculas / minúsculas, acentos, palabras compuestas, indexar o no números, etc.

Adicionalmente al hacer una búsqueda con Oracle Text se da un porcentaje de similitud entre los resultados encontrados y lo que se busca, una opción de gran ayuda para los usuarios del portal web

Debido a los inconvenientes presentados al momento de configurar la herramienta se dejó a un lado para seguir avanzando en el desarrollo del portal pero se espera que se puedan superar e intentar de nuevo utilizarla. El ingeniero tutor posee la información que se recopiló sobre la configuración y el uso de dicha herramienta.

5.3.2.2.1 Procedimiento buscar_1. El procedimiento buscar_1 despliega el buscador sencillo en donde, por medio del ingreso de una o más palabras, el usuario puede encontrar lo que necesita buscando en todos los sistemas, en todos los objetos y en todos los componentes sin ninguna restricción.
Este buscador se encuentra en el frame izquierdo. Este procedimiento llama al procedimiento *resultado* que se encuentra en este mismo paquete.

5.3.2.2.2 Procedimiento buscar_2. Este procedimiento permite que se envíen los datos correspondientes del buscador avanzado al procedimiento *resultado*. Como los datos del rango de fechas necesitan de una pequeña conversión fue necesario crear el procedimiento *presultado* el cual hace dicha función.

Desde las diferentes restricciones se hace el ingreso de los parámetros de entrada del procedimiento *presultado* los cuales son los mismos que el procedimiento *resultado*. Se tiene como restricciones la posibilidad de seleccionar un sistema específico o todos a la vez y de igual forma se puede seleccionar un objeto o un componente o todos al mismo tiempo.

Los rangos de fecha se toman utilizando el procedimiento *choose_date* del paquete *owa_util*. Este procedimiento genera los elementos de entrada para el día, el mes y el año para utilizarse como parámetros de entrada de otro procedimiento. Puesto que cada elemento tiene el mismo nombre, la fecha se pasa como array.

Los parámetros de este procedimiento son:
p_name in varchar2: nombre del elemento.
p_date in date Default sysdate: valor de la fecha.

Los datos del array se guardan como un tipo de datos *datetype* y así son recibidos por el procedimiento *presultado.*

```
TYPE datetype IS TABLE OF VARCHAR2(10)
INDEX BY BINARY_INTEGER;
```

5.3.2.2.3 Procedimiento presultado. Como ya se dijo fue necesario hacer cierta conversión en el rango de fechas que el usuario selecciona como restricción de su búsqueda.

Se envían los mismos datos de entrada al procedimiento *resultado* solo que los correspondientes a las fechas sufren una conversión. La función *todate* del paquete *owa_util* toma un *datetype* y lo convierte en una variable normal del tipo *date*.

5.3.2.2.4 Procedimiento resultado. Este procedimiento es el encargado de tomar las restricciones de la búsqueda (si las tiene) y hacer la selección correspondiente de los componentes que cumplen con ellas.

Para cada tipo de buscador se declara una variable *tipo*, siendo igual a 1 para el buscador sencillo e igual a 2 para el avanzado lo que permite desarrollar de una mejor forma los *select* de la búsqueda.

Se declaró un cursor dinámico para realizar la consulta el cual funciona de la siguiente forma: se declara un variable como un varchar2 de un tamaño considerable (2000 por ejemplo) la cual guarde toda la sentencia *select*. Se abre el cursor utilizando el contenido de la variable y los parámetros específicos dados en el *where*.

Paralelamente se hace un conteo de los registros que cumplen con las restricciones.

5.3.2.3 Paquete Manuales. Este paquete fue creado para mostrar directamente y de forma completa los manuales de los diferentes sistemas de información que posee la Empresa, para ello por medio de dos procedimientos se listan todos los sistemas con sus respectivos módulos (si los tiene) y al acceder a ellos se muestran todos los objetos y componentes que contiene.

5.3.2.3.1 Procedimiento sistemas_tb. Este procedimiento lista todos los sistemas de la Empresa junto con sus módulos para ser accedidos directamente;

llama al procedimiento *mostrar_manual* y envía el nombre del sistema como parámetro de entrada.

5.3.2.3.2 Procedimiento mostrar_manual. De igual forma que en el procedimiento *sistema_tb* éste procedimiento hace un listado pero esta vez, de todos los *objetos* y *componentes* de un sistema y por medio de un link se puede acceder a ellos.

5.3.2.4 Paquete Diccionario. Paquete creado para mostrar aquellos términos que necesitan de una explicación para su mejor entendimiento.

5.3.2.4.1 Procedimiento letras. Grupo de letras del abecedario que permite por medio de un link acceder al listado de palabras que empiezan por cada una de ellas.

5.3.2.4.2 Procedimiento palabra. Procedimiento que muestra el listado ordenado de palabras que comienzan por la letra seleccionada. Este procedimiento tiene como parámetro de entrada la letra por la cual comienza la palabra que busca.

5.3.2.5 Paquete Mostrar_Com. Paquete que contiene los procedimientos necesarios para mostrar toda la información de un componente.

5.3.2.5.1 Procedimiento tabla_com. Procedimiento que enlaza los demás procedimientos del paquete (datos_bas, datos_bfile y barra_ayudas) para mostrar la información respectiva del componente.

Igualmente hace el ingreso del registro del acceso al componente a la tabla *visita* creada para guardar dicha estadística, para ello se llama el paquete *rastreo* y al procedimiento *estadistico* el cual cumple dicha función.

5.3.2.5.2 Procedimiento datos_bas. Muestra los datos básicos de un componente: nombre, nombre del sistema al cual pertenece, fecha de la última actualización a los archivos y su descripción.

5.3.2.5.3 Procedimiento datos_bfile. Cuadro de links que permiten acceder a los documentos de texto (.txt, .doc o .pdf) almacenados en las columnas proceso, detalle o excepción de tipo BFile del componente. En estas columnas se almacenan aquellos archivos que permiten extender la explicación o documentación de un componente de tipo objeto *programa* (procedimientos o funciones).

Se hace un llamado el paquete *pak_com* específicamente a la función *get_com* la cual toma del archivo BFile su directory y su nombre haciendo posible el cargue del archivo.

5.3.2.5.4 Procedimiento barra_ayudas. Botones que dirigen al usuario a la información complementaria del componente ya sea la ayuda multimedia, el listado de componentes afectados o impactados o la lista de variables que posee.

5.3.2.6 Paquete Mostrar_Mul. Paquete que contiene el procedimiento que muestra la información de los archivos multimedia que contiene un componente.

5.3.2.6.1 Procedimiento tabla_mul. Lista de todos los archivos multimedia que contiene un componente. Muestra la información básica del archivo como el nombre, la fecha de creación, la descripción, el tipo de archivo (si es una imagen, un video o un sonido) y un link para poder ver las ayudas de tipo sonido o video

las cuales se abren en una ventana nueva utilizando el programa adecuado para su ejecución y para los archivos de tipo imagen muestra en un cuadro pequeño la imagen la cual es un link que muestra en otra página la imagen más grande y con la descripción completa (procedimiento creado en el paquete *complemento_mul*). Este procedimiento llama la función *get_archivo* del paquete *pak_mul* para obtener del BFile su nombre y de esta forma cargarlo.

5.3.2.7 Paquete Complemento_Mul. Se creo este paquete para mostrar el archivo multimedia y guardar el correspondiente registro de acceso.

5.3.2.7.1 Procedimiento mostrar_imagen. Se muestra en forma más amplia el archivo multimedia de tipo imagen acompañada de su descripción completa. Hace el llamado al procedimiento *estadistico* del paquete *rastreo* para guardar el registro de la visita.

5.3.2.7.2 Procedimiento mostrar_otro. Este procedimiento se creo básicamente para guardar el registro de entrada al archivo multimedia. Redirecciona la página para mostrar, con el reproductor adecuado, el archivo multimedia de tipo video y sonido.

5.3.2.8 Paquete Mostrar_Imp. Paquete que permite visualizar la lista de los componentes afectados por el componente que se esta revisando.

5.3.2.8.1 Procedimiento ver_afectado. Este procedimiento muestra el nombre del componente que es afectado o impacto por otro componente.

5.3.2.9 Paquete Mostrar_Var. Paquete creado para mostrar el listado de variables que interactúan en el componente que esta siendo revisado.

5.3.2.9.1 Procedimiento ver_var. Este procedimiento lista todas las variables que existen o que poseen un componente con su tipo, si es de entrada, salida o de entrada y salida y su descripción.

5.3.2.10 Paquete Rastreo. Al verse la necesidad de tener un control de cuáles son los componentes y las ayudas multimedia más utilizadas por los usuarios se creo este paquete para hacer la inclusión de los datos a la tabla *visita*.

5.3.2.10.1 Procedimiento estadistico. Procedimiento que hace el ingreso de los datos a la tabla *visita*.

Se utiliza el paquete *owa_util* que contiene subprogramas útiles para conseguir el valor de las variables de entorno del cgi, impresión de los datos que son retornados al cliente e impresión de los resultados de una consulta en una tabla HTML. La función *get_cgi_env* de éste paquete retorna el valor de la variable de entorno especificada del cgi las cuales pueden ser: remote_addr, remote_user, http_host, server_name, server_port entre otros. Con remote_addr se obtiene la dirección ip del equipo desde el se accedió a la ayuda.

5.3.2.10.2 Función nombre_equipo. Otro dato útil para saber desde que equipo se ingresó al sistema y se visitó determinado componente o archivo multimedia es el nombre del equipo para ello esta función que retorna dicho dato.

5.3.2.11 Paquete Pak_Func. Este paquete determina el usuario que accede al sistema y los tipos de objeto que pueden ser visibles para él.

5.3.2.11.1 Función pertenece_sistemas. Esta función retorna un booleano. Es TRUE cuando el usuario que ingresó al sistema es de la Subgerencia de

Informática y Tecnología de la Empresa el cual se debe registrarse con un login y un password al ingresar para tener acceso a todos los tipos de componentes.

5.3.2.11.2 Función objetos_visibles. Esta función retorna los tipos de objetos que pueden ser accedidos por los usuarios. Los usuarios que no pertenecen al área de sistemas solo pueden ver los archivos de documentación de los objetos *pantalla* y *reportes.*

5.3.2.12 Paquete Util. Paquete creado para facilitar la escritura de código que se repite en varios procedimientos y funciones.

5.3.2.12.1 Procedimiento titulo. Este procedimiento coloca el título a la página consultada el cual siempre será centrado y en letra cursiva.

5.3.2.12.2 Función font_text. Con esta función se cambia el tipo de letra utilizado en los textos descriptivos. Por defecto es de color negro, de tipo arial y de tamaño 2 puntos. Se creó como función y no como procedimiento porque es necesario que pueda incluirse dentro de otros procedimiento.

5.3.2.12.3 Función font_titulo. Esta función cambia el tipo de letra utilizado en los títulos que encabezan tablas o para textos a los que se les quiere hacer énfasis. Por defecto es en negrilla, de color negro y de tipo comic sans ms de 2 puntos de tamaño.

5.3.2.12.4 Función tilde. La función para colocar tilde a una vocal es algo largo y tedioso de escribir utilizando código PL/SQL por ello se creo esta función la cual retorna de una vez la vocal con su respectiva tilde.

5.3.2.13 Paquete Pak_Com. Este paquete fue creado para el manejo de los archivos de tipo BFile.

5.3.2.13.1 Función get_com. La función toma de la columna (*detalle*, *proceso*, *excepción* de la tabla *componente*) el archivo BFile y de él toma su nombre. El paquete *DBMS_LOB* proporciona subprogramas para el funcionamiento de los tipo de datos Lobs y BFile para los cuales proporciona operaciones de solo lectura. El procedimiento utilizado es el *filegetname* que determina el alias del directorio y el nombre del archivo cuando es dado el localizador del BFile.

5.3.2.14 Paquete Pak_Mul. Este paquete igual que el *pak_com* fue creado para el manejo de los archivos BFile, los archivos multimedia de documentación.

5.3.2.14.1 Función get_archivo. Esta función tiene el mismo uso que la función *get_com.* Utiliza el procedimiento *dbms_lob.filegetname* para tomar el nombre del archivo almacenado en la columna *contenido* de la tabla *multimedia* y de esa forma cargalo y mostrarlo por pantalla.

5.3.3 Algunos procesos desarrollados

En algunos paquetes se introdujo un proceso adicional que permite ver los links "*primero, anterior, siguiente, último*" que facilitan el navegar entre páginas y otro para manejar los errores, es decir, conocer el código y el nombre del error para poderlo corregir.

5.3.3.1 Links para navegar entre páginas. Este proceso se introdujo en la mayoría de los paquetes (manuales, diccionario, buscar, mostrar_com, mostrar_mul, mostrar_imp y mostrar_var) y sirve para ayudar la navegación entre las diferentes páginas.

En los procedimientos se colocaron dos parámetros de entrada adicionales que controlan el comienzo, el final y la cantidad de elementos que se pueden mostrar por página. Se hacen los respectivos links utilizando el procedimiento *htp.anchor* y colocando los respectivos valores a esta variables.

5.3.3.2 Manejo de errores. Como se explica en el Anexo D los bloques PL/SQL tienen una estructura compuesta por tres secciones: la sección declarativa en donde se declaran todas las constantes y variables que se van a utilizar; la sección de ejecución que incluye las instrucciones a ejecutar y la sección de excepciones en donde se definen los manejadores de errores que soportará el bloque PL/SQL.

Se utilizó esta última sección para conocer y de esta forma manejar los errores presentados en el desarrollo del software ya que cuando existe un error de contenido de una sentencia Sql el compilador de Sql Navigator no lo detecta y al momento de correr el procedimiento no se muestra ningún mensaje a menos de tenerlo declarado en esta sección.

5.4 ESTRUCTURA DEL PORTAL WEB

A continuación se presenta el estado general del portal web; por medio de gráficos se mostrarán las diferentes páginas realizadas.

5.4.1 Inicio

El portal está dividido en tres frames: en el frame 1 (lado izquierdo) están las opciones de ingreso y cierta información de interés, en el frame 2 (parte superior) esta el logo del sistema de información desarrollado y en el frame 3 (frame

principal) es donde se muestra toda la información y cambia según lo consultado. Los frames 1 y 2 son permanentes y no cambian.

Figura 31. Presentación del Portal Web de Documentación

5.4.2 Buscadores

Se crearon dos tipos de buscadores: un buscador sencillo ubicado al lado izquierdo de la pantalla y un buscador avanzado que mediante restricciones permite obtener un mejor resultado.

El buscador avanzado posee las restricciones del sistema donde se quiere buscar, del tipo de objeto, del tipo de componente y del rango de fechas entre las cuales puede estar actualizado o creado un componente.

La siguiente figura ilustra como quedó desarrollado el buscador avanzado.

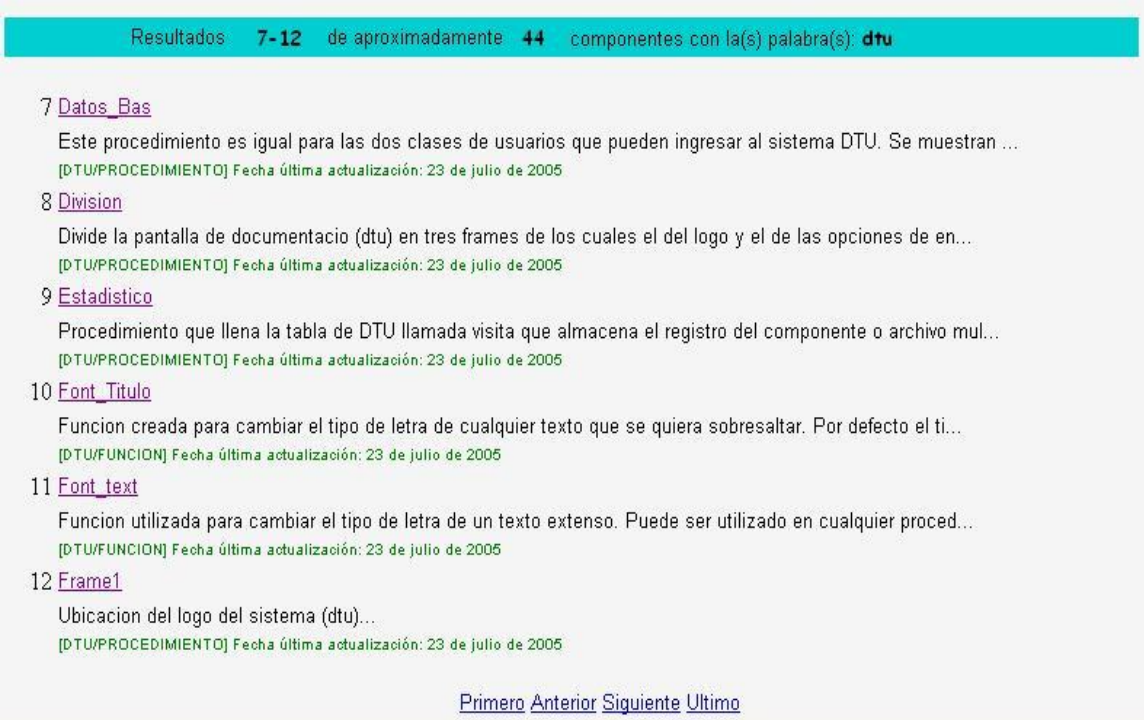
Figura 32. Buscador avanzado

En los resultados de la búsqueda se muestra de cada componente su nombre el cual es un link para acceder a toda su información, una pequeña descripción, el sistema al cual pertenece, el tipo de componente y la fecha de creación o la última actualización.

Se muestra también una barra con la información de la cantidad de componentes encontrados, la cantidad que se muestran en esa página y la o las palabras que se introdujeron en el buscador.

Cuando no se encuentra ningún resultado de la búsqueda se despliega un mensaje de error con una serie de sugerencias para realizarla nuevamente.

Figura 33. Resultados de una búsqueda

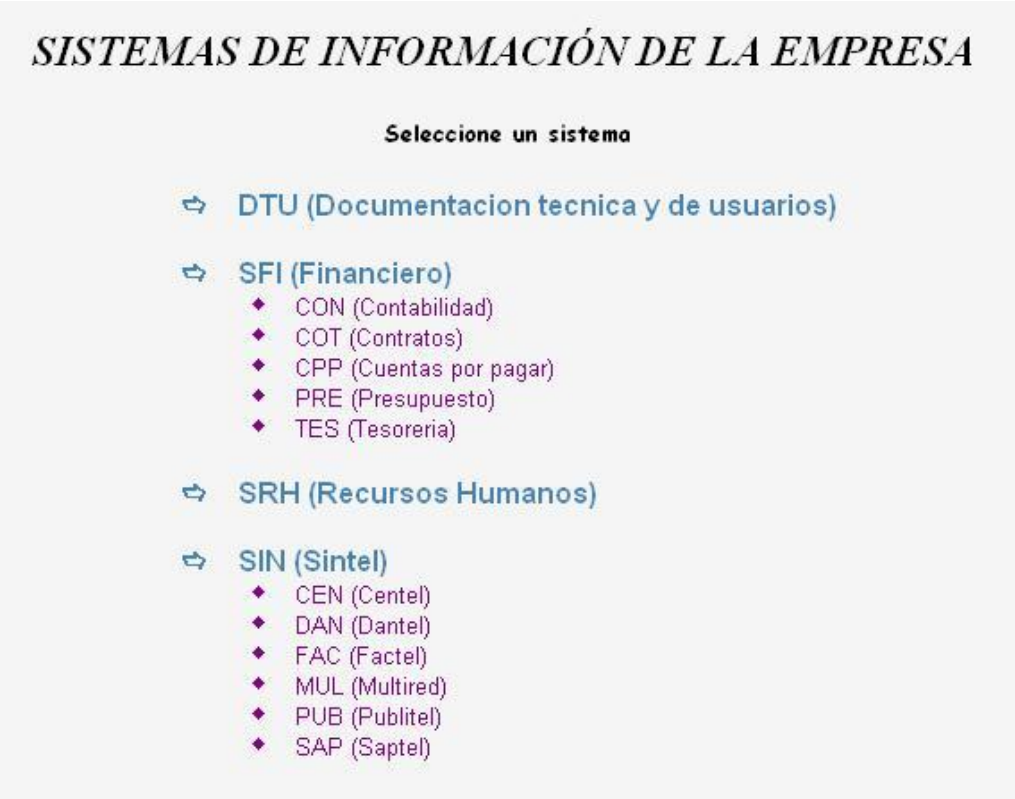


5.4.3 Manuales

Se vio la necesidad de implementar la opción de ingresar al manual completo de un sistema al pensar que el portal de documentación también sería utilizado por el personal que ingresa a la Empresa y necesita capacitarse en el funcionamiento de un sistema.

Primero se muestra el listado de todos los sistemas que posee la Empresa.

Figura 34. Listado de sistemas de Información que posee la Empresa



Después de dar clic en un sistema o módulo de un sistema se muestra su manual el cual puede ser tan amplio como documentación se tenga de él.

Figura 35. Manual completo de un sistema

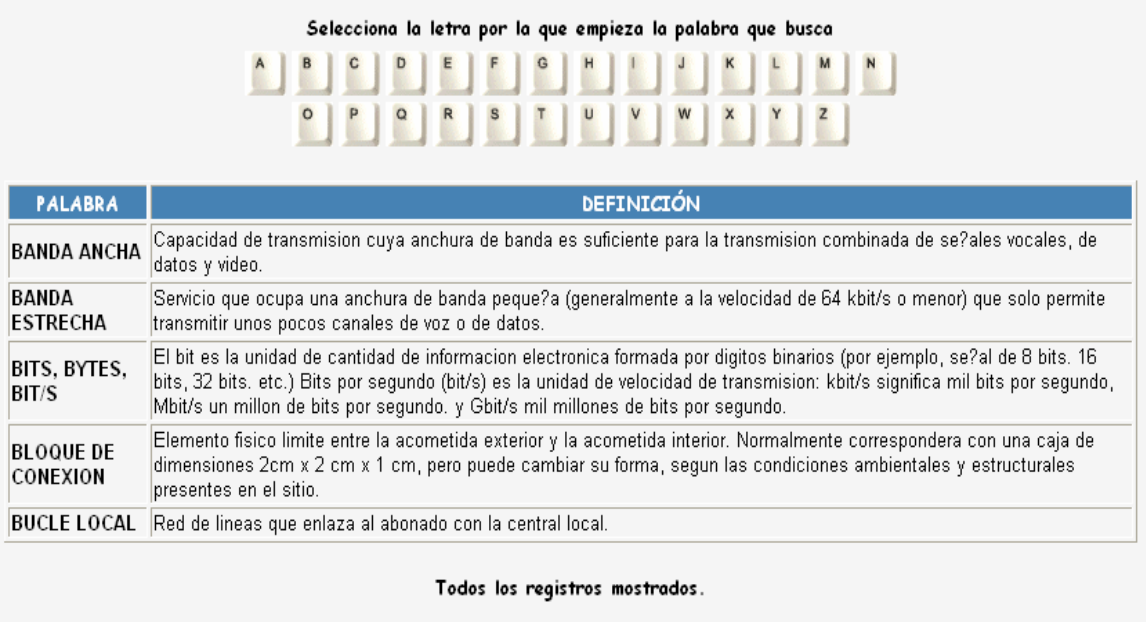
5.4.4 Glosario

Se tomó la opción de implementar un glosario de términos de uso frecuente ya que cuando se realizó el análisis inicial a la actual documentación se encontraron términos desconocidos o se vio que era necesario para el ingreso de datos en una pantalla introducir siglas o abreviaciones de las cuales se desconocía su significado.

En este momento el glosario cuenta con un listado de términos de telecomunicaciones (más de 100 palabras) que puede ser reemplazado o complementado por los ingenieros desarrolladores según sea necesario.

Desde el botón ubicado en la parte izquierda de la pantalla se puede acceder al glosario y después de seleccionar una letra se muestra el listado de palabras que empiezan por ella.

Figura 36. Glosario de términos



Selecciona la letra por la que empieza la palabra que busca

A B C D E F G H I J K L M N

O P Q R S T U V W X Y Z

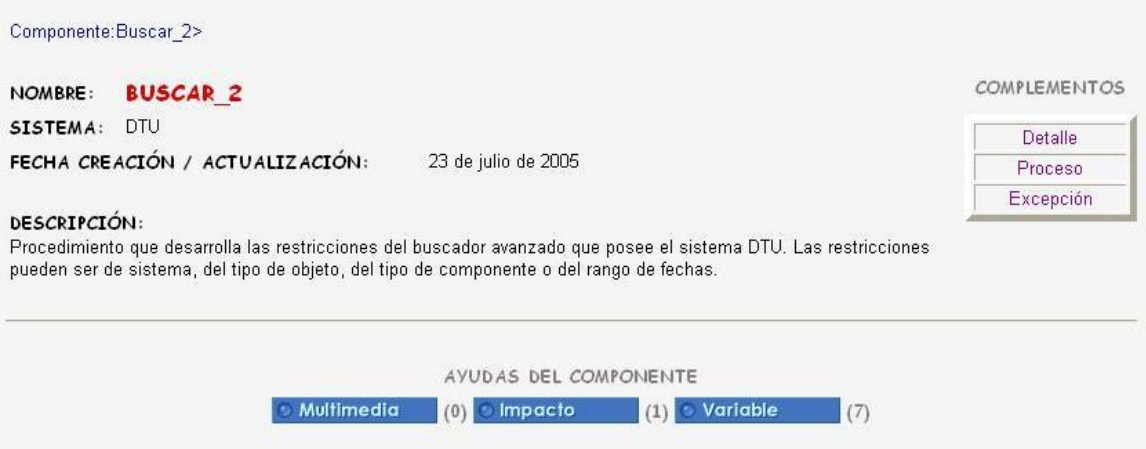
PALABRA	DEFINICIÓN
BANDA ANCHA	Capacidad de transmision cuya anchura de banda es suficiente para la transmision combinada de se?ales vocales, de datos y video.
BANDA ESTRECHA	Servicio que ocupa una anchura de banda peque?a (generalmente a la velocidad de 64 kbit/s o menor) que solo permite transmitir unos pocos canales de voz o de datos.
BITS, BYTES, BIT/S	El bit es la unidad de cantidad de informacion electronica formada por digitos binarios (por ejemplo, se?al de 8 bits. 16 bits, 32 bits. etc.) Bits por segundo (bit/s) es la unidad de velocidad de transmision: kbit/s significa mil bits por segundo, Mbit/s un millon de bits por segundo. y Gbit/s mil millones de bits por segundo.
BLOQUE DE CONEXION	Elemento fisico limite entre la acometida exterior y la acometida interior. Normalmente correspondera con una caja de dimensiones 2cm x 2 cm x 1 cm, pero puede cambiar su forma, segun las condiciones ambientales y estructurales presentes en el sitio.
BUCLE LOCAL	Red de lineas que enlaza al abonado con la central local.

Todos los registros mostrados.

5.4.5 Información de un componente

Como ya se dijo de un componente se muestra la información básica, los links para el acceso a los archivos BFile que amplían su descripción y las ayudas multimedia que posee.

Figura 37. Información de un componente



5.4.6 Ayudas multimedia

La forma de mostrar las ayudas multimedia se tomó de como lo hace la página de Oracle la cual por medio de tablas muestra todas las clases de archivos que documentan cierta aplicación.

Figura 38. Ayudas multimedia

5.4.7 Componentes impactados

Los componentes impactados o afectados por otro son los que al momento de hacer una modificación también se modifican. Está opción es implementada ya que los ingenieros desarrolladores frecuentemente ven la necesidad de conocer dicho dato.

Figura 39. Tabla de componentes impactados o afectados

5.4.8 Variables utilizadas

Listado de variables que interactúan en un procedimiento o función. Este listado indica rápidamente al ingeniero desarrollador qué variables tiene un aplicación, de qué forma están declaradas y para qué se utilizan.

Figura 40. Listado de Variables

Componente:Resultado> Variables>

VARIABLES

#	NOMBRE	TIPO DE DATO	DESCRIPCIÓN
1	Cadena	Entrada	Recibe de los buscadores la cadena de caracteres para realizar la busqueda
2	Cantador	Entrada/Salida	Ayuda a organizar la tabla para mostrar los datos
3	Cantidad_mostrada	Entrada	Cantidad de elementos que se muestran por pantalla
4	Comando	Entrada/Salida	Agrupa y arma el select
5	Comienzo_mostrado	Entrada	Marca el inicio de la pagina
6	Contador_registro	Entrada/Salida	Guarda el numero de datos que cumple con las restricciones del select
7	Contar	Entrada/Salida	Cuenta la cantidad de registros que cumplen con el select
8	Fecha1	Entrada	Recibe de los buscadores el primer rango de fechas apartir del cual se buscara
9	Fecha2	Entrada	Recibe de los buscadores el segundo rango de fecha donde se buscara
10	Sis	Entrada	Recibe de los buscadores el tipo sistema donde se buscara

Siguientes Ultimo

Inicio de registros mostrados.

6. EVALUACIÓN

Constantemente se estuvo evaluando el trabajo realizado en la práctica tanto por el tutor empresarial como por el profesor encargado por los Miembros del Comité de Proyectos de Grado de la Escuela de Ingeniería de Sistemas e Informática para hacer las evaluaciones correspondientes.

Por parte de la Universidad se evaluaban los informes de avance presentados cada cinco semanas (tal y como dice el reglamento de pregrado) por medio del profesor encargado y por parte de la Empresa, el tutor empresarial avalaba los informes antes de ser presentado a la Universidad. Adicionalmente al finalizar cada actividad o investigación se hacia entrega de un informe de lo realizado, se entregaron informes sobre: el diagnóstico sobre la actual documentación con que cuenta la Empresa; el estudio realizado sobre posibles soluciones para el portal web, los requerimientos del sistema que se apoyó en su desarrollo (en el capítulo tres se muestran estos informes) y configuración de la herramienta Oracle Text.

De igual forma aproximadamente a la mitad del periodo de la práctica se entregó un informe de avance del proyecto al Auditor Interno de la Empresa. Como ya se ha dicho una de las razones por la cual se inicia este proyecto es el informe presentado por la empresa consultora KPMG al respecto de la documentación de los sistemas de información de la Empresa. El Auditor Interno es el encargado de velar porque todas las recomendaciones sean tomadas en cuenta y se presenten los correctivos adecuados. El informe contenía el estado de avance de los dos proyectos que se estaban realizando en la Empresa. El Anexo D presenta dicho informe.

Al finalizar la práctica se realizó una última evaluación por ambas partes, la Universidad y la Empresa. El profesor encargado por parte de la Universidad constató personalmente, asistiendo a la Empresa, el trabajo que allí se realizó y los medios con que se contó para hacerlo. Por parte de la Empresa se realizaron dos exposiciones, una a los ingenieros desarrolladores de la Subgerencia de Informática y Tecnología y otra posteriormente al Subgerente de Informática y al Auditor Interno. Todos presentaron una misma opinión, el software fue realizado a conformidad con los objetivos propuestos pero hasta no tener la base de datos completamente poblada no se puede saber que tan confiable es.

Las pruebas por parte de los usuarios finales se aplazaron hasta contar con la base de datos poblada. Las pruebas de funcionamiento se hicieron por parte del tutor empresarial con los datos ya ingresados los cuales corresponden a la documentación del sistema realizado; el sistema de Documentación Técnica y de Usuario Final es un sistema más de la Empresa y como tal debe tener sus archivos de documentación completos y actualizados dentro de él mismo.

7. CONCLUSIONES

Después de terminar los seis meses de trabajo en la Empresa TELEBUCARAMANGA y de entregar la interfaz del portal web de documentación de los sistemas de información se puede concluir según los objetivos propuesto:

- ü Se recibió por parte del tutor empresarial toda la capacitación necesaria para desarrollar correctamente la práctica.

- ü Aunque la base de datos no está completamente poblada se dio un gran paso para conseguirlo, se diseñó una metodología para hacerlo, dejando ingenieros desarrolladores motivados y con herramientas didácticas que agilizan y hacen de esta una tarea menos tediosa de realizar.

- ü Se encontró la fundamentación de almacenar archivos multimedia en bases de datos y una de las mejores formas para realizarlo utilizando las herramientas que la Empresa posee.

- ü Después de sustentar el uso una página web y no un medio impreso para presentar la documentación de los sistemas de información y de estudiar aquellas reglas básicas para el desarrollo de su interfaz, se creó un portal web siempre teniendo presente que cada una de sus partes realmente fueran utilizadas por los usuarios.

- ü La implementación de un esquema de búsqueda facilita encontrar rápidamente lo requerido. La interfaz del buscador avanzado desarrollado permite mediante restricciones obtener mejores resultados de la búsqueda.

- La información que posee la Empresa sobre la importancia de documentar sistemas de información, la forma de hacer este trabajo y la mejor forma de presentarla contribuye a tener usuarios más capacitados y con herramientas a su alcance para solucionar sus inquietudes e ingenieros desarrolladores con más tiempo para realizar nuevas aplicaciones con su debida documentación.

- Cabe aclarar que por falta de una visión más amplia del proyecto al momento de plantearse los objetivos se dejó de mencionar una parte muy importante de la práctica que continuamente se estuvo realizando, la investigación. Por ello uno de los principales objetivos que se alcanzó con esta práctica fue la investigación constante sobre la importancia de documentar sistemas de información, el almacenamiento de archivos multimedia en base de datos y el desarrollo de un portal web funcional para la consulta de dicha información por parte de los usuarios finales de los sistemas y los ingenieros desarrolladores. Toda esta información se presenta en este libro y se dejó en la Empresa como parte del trabajo realizado.

8. RECOMENDACIONES

La siguiente lista sugiere como mejorar el actual estado del portal web ya que por falta de tiempo y ciertos obstáculos encontrados en el camino no se pudo dejar funcionando como inicialmente se pensaba.

- Para cargar la información en la base de datos se recomienda integrar a los usuarios finales que conocen completamente el funcionamiento de los sistemas ya que a diario trabajan con ellos.

- Se pueden tener otros tipos de archivos para documentar los sistemas. Actualmente se tienen las imágenes, videos y sonidos en esta lista de tipos de archivos pero a medida que avance el proceso de ingreso de datos puede verse la necesidad de ingresar un tipo de archivo nuevo, por ejemplo archivos de solo texto, el cual se dejó por fuera para evitar caer nuevamente en una documentación estática.

- Sería conveniente tratar de nuevo de configurar la herramienta Oracle Text para realizar las búsquedas en el sistema. Lo ideal es contar con una herramienta de esta altura para extraer de muchas formas la información de la base de datos además de poder presentar un porcentaje de exactitud entre los elementos encontrados y los buscados.

- Es necesario tener un control periódico de los archivos multimedia que más se visitan y los que menos para dejar solo lo necesario almacenado y no saturar el servidor. Es importante tener especial cuidado con este procedimiento ya que puede provocar errores al momento de mostrar los archivos en el portal web.

- La implementación del acceso al portal web desde los sistemas de información es muy práctico ya que la interacción entre estos dos sistemas hace más rápido el proceso de encontrar soluciones o respuestas a inquietudes por parte de los usuarios sin necesidad de recurrir a los ingenieros del área de sistemas.

BIBLIOGRAFÍA

REALIZACIÓN DE LA ENCUESTA

- AVILEZ, José A. Recolección de datos. (online). http://www.monografias.com/trabajos12/recoldat/recoldat.shtml

Esta página presenta una explicación sobre las diferentes técnicas para hallar datos de las cuales se decidió por la encuesta ya que sus características y los resultados que se obtienen son los adecuados para cubrir las necesidades de información requerida:

- LÓPEZ GORMAZ, Carlos. ¿Cómo determinar los requerimientos de usuario?. (online). (Agosto, 2004). http://www.uc.cl/related/atees/chile/domeyko/html/plangral/1ros/como.pdf

Este documento fue tomado como guía para el desarrollo de la encuesta ya que presenta una serie de etapas para su realización que van desde la definición de objetivos, elaboración del cuestionario y análisis e interpretación de resultados. Cuenta con un ejemplo claro que facilita la comprensión total del contenido.

- LÓPEZ RUIZ, Martha y SCHMELKES, Corina. Diseño de cuestionarios. (online). (Julio, 2002). http://www1.monografias.com/trabajos15/disenio-cuestionarios/disenio-cuestionarios.shtml

Este documento presenta pautas para la elaboración y ejecución de un cuestionario; da consejos sobre como debe ser la presentación y el lenguaje utilizado y finalmente da una breve explicación sobre como deben ser analizados los diferentes tipos de respuestas.

MARCO TEÓRICO

- MIT OpenCourseWare. Cómo documentar un sistema software. (online). http://mit.ocw.universia.net/6.170/6.170/f01/related-resources/documentation.html

MIT OCW es una iniciativa editorial electrónica a gran escala, basada en Internet y fundada conjuntamente por la Fundación William and Flora Hewlett, la Fundación Andrew W. Mellon y el Instituto Tecnológico de Massachusetts (MIT) que proporcionar un acceso libre, sencillo y coherente a los materiales de los cursos del MIT. Uno de los artículos de ésta página expone como documentar un sistema presentando un esquema base a seguir.

- DIÉGUEZ H., Rodolfo A. Documentación de Sistemas. (online). Enero, 2004. http://www.ilustrados.com/publicaciones/EpZVVZppVlxAiNyJPQ.php

Documento que presenta una definición de documentación de sistemas, la importancia de realizarla y la forma como se presentan los manuales administrativos y de usuarios.

DESARROLLO DE LA BASE DE DATOS

- LAMADRID MENDOZA, Victor Hugo. Almacenamiento Multimedia en Base de Datos Relacionales con ASP.NET. (online). Julio, 2004. http://www.informatizate.net/articulos/almacenamiento_multimedia_en_base_de_datos_relacionales_con_asp_net_05072004.html

Este documento expone las ventajas y desventajas de almacenar en una base de datos información no estructurada o información compleja como es el caso de imágenes y archivos de texto de gran tamaño.

ü MARQUÉS, Mercedes. Apuntes de ficheros y bases de datos. (online). Febrero, 2001. http://www3.uji.es/~mmarques/f47/apun/node83.html

El capítulo sobre el diseño conceptual de bases de datos de este texto explica paso a paso su desarrollo, además de presentar en forma completa los elementos que conforman un diagrama entidad-relación.

PORTAL WEB

ü ÁLVAREZ, Miguel Ángel. (Director de DesarrolloWeb.com). Manual Usabilidad en la Web. (online). http://www.desarrolloweb.com/manuales/5.

Este manual expone recomendaciones sobre como se debe desarrollar una página web que realmente sea útil y fácil de manejar por parte de los usuarios. En cada uno de los capítulos presenta ejemplos con links a otras páginas sobre cómo debe ser y cómo no la estructura de las diferentes partes que conforman una web.

ü VEGAS, Jesús. Creación de un portal web docente. (online). Marzo, 2001. http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/pordocente.html

Esta página aunque se refiere a portales web docentes presenta una serie de recomendaciones básicas sobre el diseño y la navegación para un portal de cualquier índole. Éstas recomendaciones guiaron el proceso de diseño y desarrollo del portal de documentación.

HERRAMIENTA CASE ORACLE DESIGNER

ü BARONA, Hugo Alexander. Cómo fluye la información en Oracle Designer. (online). http://www.usb.edu.co/revistas_pdf/Ingenierias2_sistemas.pdf

El primero de los artículos que se presentan en esta página explica el flujo de la información en la herramientas case Oracle Designer, considerada una de las mejores para el modelamiento de sistemas de información y desarrollo rápido de aplicaciones.

SQL NAVIGATOR

- http://www.quest.com/sql_navigator/index.asp
- SQL NAVIGATOR FOR ORACLE. User's Guide. Version 4. (pdf)
- SQL NAVIGATOR FOR ORACLE. Getting Started. Version 4. (pdf)

Página de la empresa de software Quest Software de la cual es la herramienta SQL Navigator utilizada para trabajar con la base de datos. Los archivos pdf guían el proceso de instalación, configuración y enseñan el manejo del interfaz y todas las demás opciones. (estos documentos se instalan con la herramienta).

Anexo E: MANUAL SQL

- Cursos gratis MailxMail. SQL. (online). Mayo, 2004. http://www.mailxmail.com/curso/informatica/sql#indice

- Iniciación al lenguaje SQL. (online).http://www.lania.mx/biblioteca/seminarios/basedatos/sql.html

- VEGAS, Jesús. Introducción al SQL. (online). (Valladolid, España). Abril, 1998. http://www.infor.uva.es/~jvegas/cursos/bd/sqlplus/sqlplus.html

- Wikipedia, La enciclopedia libre. SQL. (online). Junio, 2005. http://es.wikipedia.org/wiki/SQL

Manuales gratuitos sobre SQL los cuales aportaron gran parte de información para es desarrollo de éste manual.

Anexo F: MANUAL DE PL/SQL

- ALCOCER, Juan Carlos. Manual de Oracle. (online). http://programatium.com/manuales/oracle/

- ZAVALETA MORALES, Heriberto y PEÑA ACEVEDO, Joaquín. Programación en PL/SQL. (online). http://www.lania.mx/biblioteca/seminarios/basedatos/plsql/index.html

Manuales de PL/SQL donde se explica la estructura de un bloque y específicamente sobre que se hace en cada una de las tres secciones que lo conforman.

Anexo G: MANUAL DE CARTRIDGE DE PL/SQL

- ODEWAHN, Andrew. Oracle Web Applications, PL/SQL Developer's Introduction. (online). (Septiembre, 1999). (capítulos 6 y 7). http://www.unix.org.ua/orelly/oracle/webapp/index.htm

Manual de Cartridge de PL/SQL.

- BROWN, Bradley D. y Otros. ORACLE APPLICATION SERVER WEB TOOLKIT REFERENCE. Pág. 426-790. 1998.

Este libro en su segunda parte muestra los procedimientos y las funciones de los paquetes HTP y HTF que permiten visualizar por pantalla datos como una página HTML.

ü ORACLE CORPORATION. Oracle9i Application Server PL/SQL Web Toolkit Reference Release 1.0.2.2. http://www.di.unipi.it/~ghelli/bdl/A97329_03/web.902/a90101/toc.htm

Manual de referencia de las herramientas web de PL/SQL que expone todos los procedimientos y funciones HTP y HTF que se pueden utilizar, al igual que todos los paquetes OWA que existen.

ANEXO A. CRONOGRAMAS COMPARATIVOS

El siguiente es el cronograma de actividades presentado a la Universidad con la propuesta de las actividades y los espacios de tiempo estipulados para llevar a cabo la práctica empresarial en la Empresa TELEBUCARAMANGA.

ACTIVIDADES	enero		febrero				marzo				abril				mayo					junio				julio			
	24	31	7	14	21	28	7	14	21	28	4	11	18	25	2	9	16	23	30	6	13	20	27	4	11	18	25
Revisión de la documentación existente	X	X																									
Diagnóstico	X	X																									
Estado del arte y posibles soluciones		X	X	X																							
Problemas con la falta de documentación		X	X																								
Búsqueda de documentación de ayuda en línea		X	X																								
Definir posibles soluciones			X	X																							
Desarrollo de la Base de Datos					X	X	X	X	X	X	X	X	X	X													
Capacitación en la herramienta CASE					X																						
Diseño de la Base de Datos					X	X																					
Poblar la Base de Datos con datos de prueba							X	X	X	X	X	X	X	X													
Desarrollo del Software								X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Capacitación y Autocapacitación								X	X	X	X	X	X	X	X	X	X										
Desarrollo de la aplicación										X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Entrega de informes de avance					24					31					3					7					12		

A continuación se presentará el comparativo entre las actividades realizadas y las planteadas para cada periodo de la práctica.

PRIMER PERIODO

Las actividades planificadas para este periodo son la revisión de la documentación existente en la Empresa y expresar un diagnóstico; definición de problemas en los cuales se puede incurrir por no tener una documentación de los sistemas o no

tener la adecuada; búsqueda de posibles soluciones que pueden ser útiles para la Empresa teniendo en cuenta que se esta usando actualmente para documentar.
En este periodo se cumplió con todas las actividades planteadas ya que para su realización se desarrollo un análisis de la documentación existente y se hizo una encuesta para saber la opinión de los ingenieros sobre la actual documentación y sobre cómo se puede mejorar. También se buscó sobre posibles soluciones u opciones para la nueva forma de documentar.

SEGUNDO PERIODO

Entre las actividades que se tenían planteadas por realizar en este periodo de trabajo están:

- Capacitación en la herramienta case.
- Diseño y desarrollo de la base de datos.
- Poblar con datos de prueba la base de datos.
- Capacitación en SQL.

Con la capacitación impartida por el ingeniero tutor de la Empresa se cumplieron con dos de las actividades planteadas ya que como se mostrará en el manual desarrollado se ampliaron y se reforzaron los conocimientos sobre SQL y se aprendió el manejo de la herramienta case ORACLE DESIGNER.

También se cumplió con la actividad de apoyar el diseño y desarrollo de la base de datos de documentación de los sistemas de información de la Empresa y como se mostró en el capitulo cuatro se elaboró el diagrama entidad – relación y el modelo de datos.

Lo correspondiente a poblar la base de datos con datos de prueba no se cumplió a cabalidad ya que se tomó más tiempo de lo planificado para el diseño y desarrollo de la base de datos, sólo se introdujo datos para probar la longitud de los atributos de las tablas.

TERCER PERIODO

Según el cronograma de actividades para éste periodo, una de las actividades por realizar, es el ingreso de datos de prueba a la base de datos de documentación, esta actividad se pospuso ya que primero se desarrollará una interfaz, haciendo uso de la herramienta de Oracle Developer Suite 10g, FORMS BUILDER, con la cual se poblará la base de datos de documentación. Por lo anterior en este periodo solo se ingresaron una pequeña cantidad de datos utilizando la herramienta SQL NAVIGATOR.

Las otras dos actividades planteadas son capacitación en herramientas de desarrollo y desarrollo de aplicaciones para lo cual se trabajó en la búsqueda de la mejor herramienta que bajo el ambiente de Oracle 9i AS y Oracle Developer Suite 10g ayude y facilite el trabajo de mostrar por pantalla los registros de la base de datos. Se probaron dos herramientas el WebUtil (que se maneja desde Forms Builder) y el Cartridge de PL/SQL (que puede utilizarse desde herramientas como Sql Navigator) quedando esta última como la más recomendada a utilizar por las razones expuestas en el capitulo cinco.

CUARTO PERIODO

En estas semanas se trabajó en el desarrollo de aquellos paquetes que, junto con los respectivos procedimientos y funciones, hacen posible visualizar los datos existentes en la base de datos creada sobre la documentación de los sistemas de información de la Empresa.

Para este desarrollo fue necesario una explicación por parte del ingeniero tutor sobre la estructura de los paquetes y sobre cómo se crean desde la herramienta de desarrollo SQL NAVIGATOR la cual se utilizó para tal fin. Una vez comprendida su estructura y con la ayuda del libro “Oracle Application Server Web Toolkit Reference” y de otras páginas web se empezó el desarrollo de dicha aplicación.

QUINTO PERIODO

En este último periodo se trabajó en el desarrollo de la aplicación que permite por medio de la web visualizar los registros de la base de datos de documentación. Se perfeccionaron los paquetes creados ampliándose o reduciéndose, según el caso, los procedimientos o funciones desarrolladas. En el periodo restante se realizó la documentación del sistema para lo cual se crearon ayudas multimedia como videos que expliquen claramente su funcionamiento y se evaluó el trabajo por parte de la Universidad y de la Empresa.

ANEXO B. ENCUESTA

ENCUESTA SOBRE LA DOCUMENTACIÓN DE LOS SISTEMAS DE INFORMACIÓN DE LA EMPRESA

OBJETIVOS

Conocer la opinión de los ingenieros desarrolladores sobre la documentación que actualmente poseen los sistemas de información de la Empresa.

Apoyar con éstas respuesta la búsqueda de soluciones para el problema de contar con una documentación actualizada, útil y amigable.

Recolectar sugerencias de cómo debería ser una metodología que ayude y agilice el proceso de documentación de las modificaciones de los sistemas.

Población a encuestar: Ingenieros desarrolladores de la Subgerencia de Informática y Tecnología

MOTIVO DE LA ENCUESTA

La Subgerencia de Informática y Tecnología de la Empresa desea conocer la opinión de los usuarios de los sistemas de información sobre la documentación que éstos poseen y la cual se encuentra tanto en archivos físicos (manuales de usuario impresos) como en la Intranet de la empresa.

Según el informe de la auditoría realizada por la empresa KPMG, la subgerencia de informática y tecnología, no posee una documentación actualizada y completa de los sistemas de información, produciendo un mal manejo de los sistemas por parte de los usuarios y pérdida de tiempo por parte de los desarrolladores, los cuales en ocasiones, tienen que volver a crear aplicaciones.

Es por eso que se cree que al contar con una documentación actualizada, completa y sencilla de manejar de los sistema se evitaría este problema, ya que el usuario podría resolver sus inquietudes y el desarrollador no se atrasaría con su trabajo.

Agradecemos su colaboración en responder las siguientes preguntas ya que ayudarán a definir el esquema de los manuales de los sistemas de información, que como producto final, se desea obtener.

PREGUNTAS

1. ¿Cuánto tiempo lleva trabajando en la subgerencia de informática y tecnología?

___ Menos de 1 año
___ Entre 1 y 3 años
___ Entre 3 y 6 años
___ Más de 6 años

2. ¿Cuál es el módulo de SINTEL que maneja?

___ CENTEL
___ DANTEL
___ FACTEL
___ MULTIRED
___ PUBLITEL
___ SAPTEL
___ Maneja otro sistema de la empresa.

3. ¿La estructura de la documentación con que se cuenta actualmente, le ha ayudado a encontrar una posible solución a sus inquietudes o las de otro usuario?

___ SI
___ NO

4. ¿Se ha visto en la necesidad de implementar alguna de las opciones del sistema que maneja olvidando o ignorando si dicha opción ya se encuentra disponible?

___ SI
___ NO

5. ¿Ha encontrado páginas web, sistemas o herramientas que posean servicios de ayuda en línea que faciliten su consulta, que sea útil y hayan llamado su atención por su funcionalidad?

___ SI
___ NO

Cuál:_____

6. ¿Cómo realizaría, de forma sencilla, la documentación de las modificaciones de los sistemas de información?

_____
_____
_____

7. ¿Qué medios de divulgación cree que son los necesarios para que la ayuda en línea realmente preste un servicio adecuado tanto a nivel de usuario como del personal de ingeniería?

_____
_____

8. ¿Considera apropiado que el mismo desarrollador alimente la documentación por medio de plantillas estándar o por el contrario, considera que esta actividad debe llevarse a cabo por un tercero?

_____
_____
_____

9. ¿Qué porcentaje de su tiempo o que tan frecuentemente es consultado para explicar una funcionalidad del software o para brindar capacitaciones particulares a usuarios?

___ Una vez al día
___ Varias veces al día
___ Una vez a la semana
___ Varias Veces en la semana
___ No muy frecuentemente

ANEXO C. FORMATO DE REQUERIMIENTOS

NOMBRE DEL MÓDULO

Descripción General del Módulo

NOMBRE SUB - MÓDULO

Descripción General del Sub - módulo

REQUERIMIENTOS FUNCIONALES

Los requerimientos deben ser:

- Definidos teniendo en mente especificar el "QUE" debe realizar el software, y no el “COMO”.
- NO AMBIGUOS: No deben incluir palabras tales como: fácil, optimo, amigable, manejable, etc.
- COMPLETOS: Hay que describir el requerimiento en sus características mas relevantes, con el cuidado de no involucrar mas de un requerimiento en una misma descripción.
- VERIFICABLES: Si no se sabe como probar el requerimiento, significa que esta mal definido.
- CONSISTENTES: Que sean razonables y realizables, cuidarse de hacer especificaciones idealizadas que no se puedan llevar a la realidad.

Tener en cuenta:

- Manejo de históricos
- Procesamiento y validaciones en lote
- Cargue de información en lotes en caso de contingencia
- Generación de alertas
- Manejo de información encriptada

- Digitalización y consulta de documentos
- Reversión de transacciones
- Pistas de auditoria y su nivel de detalle
- Definición de Parámetros (valores o formulas)
- Definición de reglas de negocio (ejemplo: reglas de liquidación y facturación de un nuevo producto)
- Importación de Información
- Manejo de excepciones
- Variables que se deben considerar en forma individual o combinada para la generación consultas, reportes, estadísticas o información gerencial
- Documentos que se imprimen en formatos predefinidos
- Generación de información en modo gráfico
- Exportación de información
- Proyecciones informes comparativos

REQUERIMIENTOS DE INTERFAZ

- Descripción general de la interfaz
- Sistemas con que se interactúa

REQUERIMIENTOS GENERALES

Adicional a los requerimientos de cada módulo hay que ir determinando los requerimientos generales del Sistema se contemplará entre otros aspectos:

- Seguridad
- Estándares en reportes
- Estándares en pantallas
- Manejo de información histórica (almacenamiento y manejo de información que por su antigüedad y volumen debe pasar a información fuera de línea)
- Manejo de múltiples monedas
- Manejo de múltiples empresas

ANEXO D: INFORME DEL PROYECTO SOBRE EL DESARROLLO DE APLICACIONES PARA LA DOCUMENTACIÓN DE LOS SISTEMAS DE INFORMACIÓN DE TELEBUCARAMANGA

OBJETIVO GENERAL

Diseñar y desarrollar herramientas software que permitan realizar consultas sobre la documentación de los sistemas de información de la Empresa por parte de los usuario finales y los ingenieros desarrolladores y a su vez que su actualización pueda hacerse de forma ágil y sencilla.

OBJETIVOS ESPECIFICOS

- Diseñar y desarrollar una base de datos de documentación que almacene los archivos multimedia de los sistemas de información de la Empresa.

- Elaborar una interfaz para el ingreso y actualización de datos de documentación por parte de los ingenieros del área de sistemas de la Empresa.

- Crear una interfaz de consulta sobre la base de datos de documentación con el fin de incorporarla en los diferentes sistemas de información manejados en la Empresa, permitiendo de esta forma el acceso a la ayuda desde las aplicaciones y desde la Intranet.

OBSERVACIONES PARA MEJORAR LA DOCUMENTACION

- La documentación existente no presta los servicios adecuados para los usuarios de los sistemas ni para los ingenieros desarrolladores.

- Un cambio en la estructura y en la presentación de los contenidos de los manuales facilitaría su uso y compresión por parte de los usuarios y evitaría que los ingenieros de la subgerencia IT inviertan más tiempo del necesario explicando el funcionamiento de los sistemas.

- El uso de la Intranet de la Empresa para colocar los manuales de los sistemas crea un acercamiento entre el sitio donde los usuarios realizan su trabajo y donde pueden resolver sus dudas.

ANALISIS DE LA ESTRUCTURA DE LA BASE DE DATOS

Se manejaron varios prototipos del diagrama Entidad-Relación antes de obtener el que más se ajustaba a las necesidades del sistema que se está desarrollando actualmente.

El diagrama esta comprendido por nueve tablas:

Sistema: encierra todos los sistemas de información que se manejan en la Empresa.

Tipo _Objeto: se llamaron objetos a aquellos elementos que conforman un sistema, ya sea a nivel de usuario o a nivel técnico. Los objetos contenidos en un sistema, pueden ser de tipo pantallas, reportes, programas, etc.

Objeto: es donde se almacenará el nombre y la descripción de los diferentes tipos de objeto que conforman los sistemas de información.

Tipo_Componente: se definieron los componentes de un objeto como todos los elementos como íconos, menús, barras de herramientas y cajas de texto incluidos en una pantalla o procedimientos y funciones incluidos en un programa.

Componente: la entidad componente almacena la descripción y la fecha de los tipo de componente que se encuentran en una pantalla y el detalle, el proceso y las excepciones de un paquete de código.

Multimedia: se almacenan todos los recursos con que cuenta un componente para su documentación. En esta entidad se definió un atributo llamado *contenido* con el tipo de dato de almacenamiento más recomendado, tipo BFILE, el cual acepta un tamaño de archivo hasta de 4 GB. Los archivos pueden ser de tipo imagen, sonido o video.

Funcionario: es la persona del área de sistema que realiza el archivo de documentación.

Variable: las variables que se manejan en un programa pueden ser de tipo entrada, salida o de entrada y salida.

Impacto: en esta entidad se define en que parte de los sistemas es utilizado cierto procedimiento o función; cuales sistemas se ven afectados al momento de realizar una modificación al código de una función o procedimiento.

DESARROLLO DE LAS INTERFACES

Actualmente se está trabajando en el desarrollo de una interfaz para el ingreso de datos utilizando la herramienta Forms Builder y otra para las consultas utilizando la herramienta de interfaz de PL/SQL, Sql Navigator para la creación de paquetes y subprogramas que permiten dicha consulta de la base de datos de documentación desde la web.

INTERFAZ PARA EL INGRESO DE DATOS

Para el ingreso de datos se dividió en dos las tablas creadas, una categoría para aquellas tablas no son tan modificables como son las tablas tipo_objeto y tipo_componente. Las tablas sistema y funcionario aunque pertenecen a esta categoría no se mencionan ya que son el empalme con los otros sistemas de información.

La otra categoría encierra las demás tablas que serán las más utilizadas por los desarrolladores. Se creó una ventana principal (para las tablas objeto y componente) que se encuentra relacionada con las demás tablas (multimedia, impacto y variable), mediante un menú emergente. Para su correcto funcionamiento fue necesario crear paquetes, procedimientos, funciones y trigger, utilizando PLSQL.

INTERFAZ PARA LA CONSULTA DE DATOS

Para la consulta de datos de la base de datos multimedia se ha trabajado en el desarrollo de pantallas de visualización en donde por medio de una búsqueda (aún en estado de desarrollo), se despliegan los datos de los componentes y los respectivos link para revisar sus ayudas multimedia.

Se esta haciendo uso del Cartridge de PL/SQL, mediante el manejo de paquetes generados de todas las tablas y paquetes creados que contienen los procedimientos y funciones necesarias para el desarrollo de este tipo de procesos.

ANEXO E: MANUAL DE SQL

HISTORIA

SQL es una herramienta para organizar, gestionar y recuperar datos almacenados en una base de datos relacionar.

El nombre "SQL" es una abreviatura de *Structured Query Languaje* (Lenguaje de consultas estructurado). Es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones sobre las mismas.

La historia de SQL empieza en 1974 con la definición, por parte de Donald Chamberlin y de otras personas que trabajaban en los laboratorios de investigación de IBM, de un lenguaje para la especificación de las características de las bases de datos que adoptaban el modelo relacional. Este lenguaje se llamaba SEQUEL (Structured English Query Language) y se implementó en un prototipo llamado SEQUEL-XRM entre 1974 y 1975.

Las experimentaciones con ese prototipo condujeron, entre 1976 y 1977, a una revisión del lenguaje (SEQUEL/2), que a partir de ese momento cambió de nombre por motivos legales, convirtiéndose en SQL. El prototipo (System R), basado en este lenguaje, se adoptó y utilizó internamente en IBM y lo adoptaron algunos de sus clientes elegidos.

Gracias al éxito de este sistema, que no estaba todavía comercializado, también otras compañías empezaron a desarrollar sus productos relacionales basados en SQL. A partir de 1981, IBM comenzó a entregar sus productos relacionales y en 1983 empezó a vender DB2.

En 1986, el ANSI adoptó SQL (sustancialmente adoptó el dialecto SQL de IBM) como estándar para los lenguajes relacionales y en 1987 se transformó en estándar ISO.

CARACTERISTICAS GENERALES

El SQL es un lenguaje de acceso a bases de datos que explota la flexibilidad y potencia de los sistemas relacionales permitiendo gran variedad de operaciones sobre los mismos.

Es un lenguaje declarativo de alto nivel o de no procedimiento, que gracias a su fuerte base teórica y su orientación al manejo de conjuntos de registros y no a registros individuales, permite una alta productividad en codificación.

- Funcionalidad

El SQL proporciona una rica funcionalidad más allá de la simple consulta (o recuperación) de datos. Asume el papel de lenguaje de definición de datos (LDD), lenguaje de definición de vistas (LDV) y lenguaje de manipulación de datos (LMD). Además permite la concesión y denegación de permisos, la implementación de restricciones de integridad y controles de transacción y la alteración de esquemas.

- Modos de uso

El SQL permite fundamentalmente dos modos de uso:
Un uso interactivo, destinado principalmente a los usuarios finales avanzados u ocasionales, en el que las diversas sentencias SQL se escriben y ejecutan en línea de comandos o un entorno semejante.

Un uso integrado, destinado al uso por parte de los programadores dentro de programas escritos en cualquier lenguaje de programación anfitrión. En este caso el SQL asume el papel de sublenguaje de datos.

En el caso de hacer un uso embebido del lenguaje podemos utilizar dos técnicas alternativas de programación. En una de ellas, en la que el lenguaje se denomina *SQL estático*, las sentencias utilizadas no cambian durante la ejecución del programa. En la otra, donde el lenguaje recibe el nombre de *SQL dinámico*, se produce una modificación total o parcial de las sentencias en el transcurso de la ejecución del programa.

La utilización de *SQL dinámico* permite mayor flexibilidad y mayor complejidad en las sentencias, pero como contra se tiene una eficiencia menor y el uso de técnicas de programación más complejas en el manejo de memoria y variables.

- Optimización

SQL es un lenguaje declarativo, especifica qué es lo que se quiere y no cómo conseguirlo, por lo que una sentencia no establece explícitamente un orden de ejecución. El orden de ejecución interno de una sentencia puede afectar gravemente a la eficiencia del SGBD, por lo que se hace necesario que éste lleve a cabo una optimización antes de la ejecución de la misma.

TIPOS DE SENTENCIAS

Las sentencias SQL pertenecen a dos categorías principales:

- Lenguaje de Definición de Datos, DDL
- Lenguaje de Manipulación de Datos, DML

Estos dos lenguajes no son lenguajes en sí mismos, sino que es una forma de clasificar las sentencias de lenguaje SQL en función de su cometido. La diferencia principal reside en que el DDL crea objetos en la base de datos y sus efectos se pueden ver en el diccionario de la base de datos; mientras que el DML es el que permite consultar, insertar, modificar y eliminar la información almacenada en los objetos de la base de datos.

Cuando se ejecutan las sentencias DDL de SQL, el SGBD confirma la transacción actual antes y después de cada una de las sentencias DDL. En cambio, las sentencias DML no llevan implícito el commit y se pueden deshacer.

Tabla 1. Sentencias del SQL.

Sentencia DDL	***Objetivo***
Alter procedure	Recompilar un procedimiento almacenado.
Alter Table	Añadir o redefinir una columna, modificar la asignación de almacenamiento.
Analyze	Recoger estadísticas de rendimiento sobre los objetos de la BD.
Create Table	Crear una tabla.
Create Index	Crear un índice.
Drop Table	Eliminar una tabla.
Drop Index	Eliminar un índice.
Grant	Conceder privilegios o papeles, roles, a un usuario o a otro rol.
Truncate	Eliminar todas las filas de una tabla.
Revoke	Retirar los privilegios de un usuario o rol de la bd.
Sentencia DML	***Objetivo***
Insert	Añadir filas de datos a una tabla.
Delete	Eliminar filas de datos de una tabla.
Update	Modificar los datos de una tabla.
Select	Recuperar datos de una tabla.
Commit	Confirmar como permanentes las modificaciones realizadas.
Rollback	Deshacer todas las modificaciones realizadas desde la última confirmación.

Fuente: VEGAS, Jesús. Introducción al SQL. (online).

TIPOS DE DATOS

Existen varios tipos de datos en SQL. De esta manera, cada columna puede albergar una información de naturaleza distinta. Los tipos de datos más comunes y sus características son:

Tabla 2. Tipos de Datos.

Tipo de Dato	***Descripción***
VARCHAR2(tamaño)	Almacena datos de tipo carácter alfanumérico de longitud variable, con un tamaño máximo de 2.000.
CHAR(tamaño)	Almacena datos de tipo carácter alfanumérico de longitud fija, con un tamaño máximo de 255.
LONG	Almacena datos de tipo carácter alfanumérico de longitud variable con un tamaño máximo de hasta 2 Gb.
NUMBER(dig,dec)	Almacena datos numéricos de dig dígitos, de los cuales dec son decimales. El tamaño máximo es de 38 dígitos.
DATE	Almacena fechas desde el 1-Ene-4712 AC hasta el 31-Dic-4712 DC.
RAW(tamaño)	Almacena datos de longitud variable, con un tamaño máximo de 255 bytes.
LONG RAW	Almacena datos de longitud variable, con un tamaño máximo de 2 Gb.

Fuente: VEGAS, Jesús. Introducción al SQL. (online).

CONSULTAS

Las consultas son el corazón del lenguaje SQL. La sentencia *SELECT*, que se utiliza para expresar consultas en SQL, es la más potente y compleja de las sentencias SQL.

La sintaxis básica de una consulta es:

```
SELECT Campos
FROM Tabla
```

En donde *campos* es la lista de campos que se deseen recuperar y *tabla* es el origen de los mismos.

Una consulta consta de seis cláusulas: las dos primeras (*SELECT* y *FROM*) obligatorias y las otras cuatro opcionales.

```
SELECT columna, ...
FROM nombre_tabla [alias_tabla] ...
[WHERE expresión1 operador expresion2 ]
[GROUP BY {columna, ...} ]
[HAVING expresión1 operador expresión2 ]
[UNION [ALL] (SELECT ...)]
[ORDER BY {expresión_orden [DESC | ASC], ... ]
```

ü Cláusula SELECT

La cláusula *SELECT* lista los datos a recuperar. Los elementos o datos a seleccionar pueden ser columnas de la base de datos o también el asterisco (*) para recuperar todos los campos de una tabla.

Sintaxis:

```
SELECT columna, ...
```

columna puede ser un simple nombre de campo. Expresiones más complejas pueden incluir operaciones matemáticas o de manipulación de caracteres.

Las columnas deben ir separadas por comas si existen más de una.

En determinadas circunstancias es necesario asignar un nombre a alguna columna determinada de un conjunto devuelto. Para asignar alias a una columna se coloca después de ella entre comillas dobles (“ ”) el nombre deseado.

Los nombres de las columnas pueden ir precedidos por el nombre de la tabla o su alias. Se utiliza esta característica para reconocer las columnas que tienen igual nombre pero que provienen de tablas diferentes.

EJEMPLOS

```
SELECT *
FROM empleado;

SELECT e.cedula, e.nombre, e.sueldo
FROM empleado e;

SELECT nombre, sueldo * 1.08
FROM empleado;
```

Consultas con Predicado

El predicado se incluye entre la cláusula SELECT y el primer nombre del campo a recuperar, los posibles predicados son:

Tabla 3. Tipos de predicados.

Predicado	***Descripción***	***Ejemplo***
ALL	Devuelve todos los campos de la tabla	SELECT ALL FROM empleados
TOP	Devuelve un determinado número de registros de la tabla	SELECT TOP 25 Nombre, Apellido FROM estudiantes ORDER BY Nombre DESC
DISTINCT	Omite los registros cuyos campos seleccionados coincidan totalmente	SELECT DISTINCT Apellido FROM empleados

Fuente: Cursos gratis MailxMail. SQL. (online).

ü Cláusula FROM

La cláusula *FROM* define las tablas de las que se van a seleccionar las columnas que contienen los datos a recuperar por la consulta. El formato de esta cláusula es:

```
FROM nombre_tabla [alias_tabla] ...
```

donde *nombre_tabla* puede ser el nombre de una o más tablas.
alias_tabla es un nombre que se usa para referirse a la tabla en el resto de la sentencia *SELECT* para abreviar el nombre original y hacerlo más manejable, en el caso de existir más de una tabla en la consulta y también para poder realizar consultas uniendo varias veces la misma tabla.

EJEMPLO

```
SELECT e.cedula, e.nombre, d.nombre
FROM empleado e, departamento d
WHERE d.dpto = e.dpto
```

es mucho más práctico y sencillo que:

```
SELECT empleado.cedula, empleado.nombre, departamento.nombre
FROM empleado, departamento
WHERE departamento.dpto = empleado.dpto
```

ü Cláusula WHERE

La cláusula *WHERE* dice a SQL que incluya solo ciertas filas o registros de datos en los resultados de la consulta, es decir, qué tienen que cumplir los registros que se desean ver. La cláusula *WHERE* se expresa de la forma:

```
WHERE expresión1 operador expresion2
```

donde *expresión1* y *expresion2* pueden ser nombres de campos, valores constantes o expresiones.
operador es un operador relacional que une dos expresiones.

EJEMPLO

```
SELECT *
FROM empleado
WHERE sueldo BETWEEN 1000000 AND 2000000;
SELECT *
FROM empleado
WHERE apellido LIKE 'RUEDA%'
```

ü Cláusula GROUP BY

La cláusula *GROUP BY* especifica una consulta sumaria. En vez de producir un fila de resultados por cada fila de datos de la base de datos, una consulta sumaria agrupa todas las filas similares y luego produce una fila sumaria de resultados para cada grupo.

Seguido de la cláusula GROUP BY se especifican los nombres de uno o más campos cuyos resultados se desean agrupados. Tiene la forma:

```
GROUP BY columna
```

columna debe coincidir con la columna utilizada en la cláusula *SELECT*. Puede ser uno o más nombres de campo de una tabla, separados por coma.

EJEMPLO

```
SELECT cargo, SUM(sueldo)
FROM empleado
WHERE cargo LIKE 'JEFE%'
GROUP BY cargo
```

ü Cláusula HAVING

La cláusula *HAVING* dice a SQL que incluya solo ciertos grupos producidos por la cláusula *GROUP BY* en los resultados de la consulta. Al igual que la cláusula *WHERE*, utiliza una condición de búsqueda para especificar los grupos deseados. En otras palabras, especifica la condición que deben de cumplir los grupos. Sólo es válida si previamente se ha especificado la cláusula *GROUP BY*.

La cláusula HAVING tiene la forma:

```
HAVING expresión1 operador expresión2
```

donde *expresión1* y *expresión2* pueden ser nombres de campos, valores constantes o expresiones y estas deben coincidir con una expresión de columna en la cláusula *SELECT*.

operador es un operador relacional que une las dos expresiones.

EJEMPLO

```
SELECT cargo, SUM(sueldo)
FROM empleado
WHERE cargo LIKE 'JEFE%'
GROUP BY cargo
HAVING SUM(sueldo) > 10000000
```

ü Operador UNION

El operador *UNION* combina el resultado de dos sentencias *SELECT* en un único resultado. Este resultado se compone de todos los registros devueltos en ambas sentencias. Por defecto, los registros repetidos se omiten. Para no quitarlos se empleará la palabra *ALL*.

Tiene la forma:

```
SELECT sentencia UNION [ALL] SELECT sentencia
```

Cuando se utilice el operador *UNION*, la lista de selección para cada sentencia *SELECT* debe tener el mismo número de expresiones de columnas con el mismo tipo de datos y en el mismo orden.

EJEMPLO

```
SELECT nombre
FROM empleado
UNION ALL
SELECT 'Alberto'
FROM dual;
```

Este ejemplo devuelve todos los *nombres* de la tabla *empleados* junto con el nombre “Alberto”.

ü Cláusula ORDER BY

La cláusula *ORDER BY* ordena los resultados de la consulta en base a los datos de una o más columnas. Si se omite, los resultados saldrán ordenados por el primer campo que se haya utilizado. Tiene la forma:

```
[ORDER BY {expresión_orden [DESC | ASC], ... ]
```

expresión_orden puede ser el nombre de un campo, expresión o el número de posición que ocupa la expresión de columna en la cláusula *SELECT*.

Por defecto se ordenan ASCendentemente (de menor a mayor). Si se deseará de mayor a menor se empleará DESC (DESCendente).

EJEMPLO

```
SELECT cargo, SUM(sueldo)
FROM empleado
```

```
WHERE cargo LIKE 'JEFE%'
GROUP BY cargo
HAVING SUM(sueldo) > 10000000
ORDER BY SUM(sueldo) DESC
```

OPERADORES

ü Operadores Numéricos

Utilizados para realizar operaciones numéricas.

Tabla 4. Operadores Numéricos.

Operador	***Operación***	***Ejemplo***
+	Suma	`select nombre, salario+comision from emp where oficio='VENDEDOR';`
-	Resta	`select nombre from emp where sysdate-fecha_alta > 365;`
*	Producto	`select nombre, salario*12 from emp;`
/	División	`select nombre, salario/31 from emp;`

Fuente: VEGAS, Jesús. Introducción al SQL. (online).

ü Operadores de caracteres

El operador de caracteres más utilizado es el de concatenación.

Tabla 5. Operador de caracteres.

Operador	***Operación***	***Ejemplo***
||	Concatenación	`select nombre||oficio from emp;`

Fuente: VEGAS, Jesús. Introducción al SQL. (online).

ü Operadores de comparación

Los operadores de relación que separan dos expresiones pueden ser:

Tabla 6. Operadores de comparación

Operador	***Operación***	***Ejemplo***
=	Igualdad	select * from emp where cod_dep = 100;
!=, <>, ^=	Desigualdad	select * from emp where cod_dep != 100;
<	Menor que	select * from emp where cod_dep < 200;
>	Mayor que	select * from emp where cod_dep > 200;
<=	Menor o igual que	select * from emp where cod_dep <= 200;
>=	Mayor o igual que	select * from emp where cod_dep >= 200;
In	Igual a cualquiera de los miembros entre paréntesis	select * from emp where cod_dep in (100, 300);
not in	Distinto a cualquiera de los miembros entre paréntesis	select * from emp where cod_dep not in (200);
between	Contenido en el rango	select * from emp where cod_emp between 100 and 199;
not between	Fuera del rango	select * from emp where cod_emp not between 100 and 199;
like '_abc%'	Contiene la cadena 'abc' a partir del segundo carácter y luego cualquier cadena de caracteres	select * from emp where nombre like 'Ma%';

Fuente: VEGAS, Jesús. Introducción al SQL. (online).

ü Operadores Lógicos

Dos o más condiciones pueden ser combinadas para formar expresiones más complejas con distintos criterios.

Tabla 7. Operadores lógicos.

Operador	*Operación*	*Descripción*
AND	“y” Lógico	Evalúa dos condiciones y devuelve un valor de verdad sólo si ambas son ciertas.
OR	“o” Lógico	Evalúa dos condiciones y devuelve un valor de verdad si alguna de las dos es cierta.
NOT	Negación Lógica	Devuelve el valor contrario de la expresión.

Fuente: Iniciación al lenguaje SQL. (online).

ü Prioridad de los operadores

En expresiones con más de una condición el orden en el que se evalúan es muy importante. La siguiente tabla muestra el orden en el que son evaluados los operadores. Los operadores que figuran en la primera línea se evalúa primero, luego los de la segunda y así sucesivamente.

Los operadores que figuren en la misma línea se evalúan de izquierda a derecha según aparezcan en la expresión.

Tabla 8. Prioridad entre operadores.

Orden	*Operador*
1	- unitario, + unitario
2	** ó ^
3	*, /
4	+, -
5	=, <>, <, >, >=, >=, Like, Not Like, Is Null, Is Not Null, Between, In, Exists, Any, All
6	NOT
7	AND
8	OR

Fuente: Iniciación al lenguaje SQL. (online).

El siguiente ejemplo muestra la importancia de la prioridad de los operadores:

```
WHERE  TIPO = 'Compra'
OR     FREGISTRO > '3/30/1996'
AND    RESP_CLAVE > 3
```

Ya que el AND se evalúa primero, esta consulta devuelve la clave de los responsables con número mayor que 3 y que hayan sido registrado con posterioridad al 30 de marzo de 1996 y que el TIPO haya sido 'Compra'.

FUNCIONES

Las funciones permiten realizar con los datos operaciones adicionales a las ya vistas, pudiendo participar como operadores en las expresiones.

Una función representa un valor único que se obtiene aplicando unas determinadas operaciones a otros valores dados, que se llaman argumentos. Se especifica como una palabra predefinida seguida de los argumentos entre paréntesis y separados por comas.

Las funciones se pueden incluir en las cláusulas *SELECT*, *WHERE* y *ORDER BY*. Pueden anidarse funciones dentro de funciones. Y existe una gran variedad de funciones para cada tipo de datos.

ü Funciones numéricas

Tabla 9. Funciones numéricas.

Función	***Descripción***	***Ejemplo***	***Rta***
ABS(n)	Calcula el valor absoluto de *n*.	select abs(-15) from dual;	15

CEIL(n)	Calcula el valor entero inmediatamente superior o igual a *n*.	select ceil(15.7) from dual;	16
FLOOR(n)	Calcula el valor entero inmediatamente inferior o igual a *n*.	select floor(15.7) from dual;	15
MOD(m,n)	Calcula el resto resultante de dividir *m* entre *n*.	select mod(11,4) from dual;	3
POWER(m, n)	Calcula la potencia *n*-esima de *m*.	select power(3,2) from dual;	9
ROUND(m, n)	Calcula el redondeo de *m* a *n* decimales. Si *n*<0 el redondeo se efectúa a por la izquierda del punto decimal.	select round(123.456,1) from dual;	123.5
SQRT(n)	Calcula la raíz cuadrada de *n*.	select sqrt(4) from dual;	2
TRUNC(m, n)	Calcula *m* truncado a *n* decimales (*n* puede ser negativo).	select trunc(123.456,1) from dual;	123.4
SIGN(n)	Calcula el signo de *n*, devolviendo -1 si *n*<0, 0 si *n*=0 y 1 si *n*>0.	select sign(-12) from dual;	-1

Fuente: VEGAS, Jesús. Introducción al SQL. (online).

ü Funciones de cadenas de caracteres

Tabla 10. Funciones de cadena de caracteres.

Función	***Descripción***	***Ejemplo***	***Resultado***
CHR(n)	Devuelve el carácter cuyo valor codificado es *n*.	select chr(65) from dual;	A
ASCII(cad)	Devuelve el valor ascii de *cad*.	select ascii('A') from dual;	65
CONCAT(cad1,cad2)	Devuelve *cad1* concatenada con *cad2*. Esta función es equivalente al operador ||.	select concat(concat(nombre,' es '),oficio) from emp;	Cano es Presidente etc.

LOWER(cad)	Devuelve la cadena *cad* con todas sus letras convertidas a minúsculas.	select lower('MinUsCulAs') from dual;	Minúsculas
UPPER(cad)	Devuelve la cadena *cad* con todas sus letras convertidas a mayúsculas.	select upper('maYuSCulAs') from dual;	MAYÚSCULAS
INITCAP(cad)	Devuelve *cad* con el primer caracter en mayúsculas.	select initcap('isabel') from dual;	Isabel
LPAD(cad1, n,cad2)	Devuelve *cad1* con longitud *n*, y ajustada a la derecha, rellenando por la izquierda con *cad2*.	select lpad('P',5,'*') from dual;	****P
RPAD(cad1, n,cad2)	Devuelve *cad1* con longitud *n*, y ajustada a la izquierda, rellenando por la derecha con *cad2*.	select rpad('P',5,'*') from dual;	P****
REPLACE(cad,ant,nue)	Devuelve *cad* en la que cada ocurrencia de la cadena *ant* ha sido sustituida por la cadena *nue*.	select replace('digo','i','ie') from dual;	diego
SUBSTR(cad,m,n)	Devuelve la subcadena de *cad* compuesta por *n* caracteres a partir de la posicion *m*.	select substr('ABCDEFG',3,2) from dual;	CD
LENGTH(cad)	Devuelve la longitud de *cad*.	select length('cadena') from dual;	6

Fuente: VEGAS, Jesús. Introducción al SQL. (online).

ü Funciones de manejo de fechas

Tabla 11. Funciones de manejo de fechas.

Función	***Descripción***	***Ejemplo***	***Resultado***
SYSDATE	Devuelve la fecha y hora actuales.	select sysdate from dual;	14-MAR-97

ADD_MONTHS(d, n)	Devuelve la fecha *d* incrementada en *n* meses.	select add_months(sysdate,4) from dual;	14-JUL-97
LAST_DAY(d)	Devuelve la fecha del último día del mes de *d*.	select last_day(sysdate) from dual;	31-MAR-97
MONTHS_BETWEEN(d1, d2)	Devuelve la diferencia en meses entre las fechas *d1* y *d2*.	select months_between(sysdate,'01-JAN-97') from dual;	2.4340942
NEXT_DAY(d,cad)	Devuelve la fecha del primer día de la semana *cad* después de la fecha *d*.	select next_day(sysdate, 'sunday') from dual;	16-MAR-97

Fuente: VEGAS, Jesús. Introducción al SQL. (online).

ü Funciones de conversión

Tabla 12. Funciones de conversión.

Función	***Descripción***	***Ejemplo***	***Resultado***
TO_NUMBER(cad,fmto)	Convierte la cadena *cad* a un número, opcionalmente de acuerdo con el formato *fmto*.	select to_number('12345') from dual;	124345
TO_CHAR(d, fmto)	Convierte la fecha *d* a una cadena de caracteres, opcionalmente de acuerdo con el formato *fmto*.	select to_char(sysdate) from dual;	'14-MAR-97'
TO_DATE(cad,fmto)	Convierte la cadena *cad* de tipo varchar2 a fecha, opcionalmente de acuerdo con el formato *fmto*.	select to_date('1-JAN-97') from dual;	01-JAN-97

Fuente: VEGAS, Jesús. Introducción al SQL. (online).

ü Funciones generales

Tabla 13. Funciones generales.

Función	*Descripción*	*Ejemplo*	*Resultado*
DECODE(var, val1, cod1, val2, cod2, ..., defecto)	Convierte el valor de *var*, de acuerdo con la codificación.	select decode(oficio, 'Presidente', 'P', 'Director', 'D', 'X') from emp;	P, D, X..
GREATEST(exp1, exp2, ...)	Devuelve el mayor valor de una lista.	sin ejemplo.	Sin ejemplo.
LEAST(cad,fmto)	Devuelve el menor valor de una lista.	sin ejemplo.	Sin ejemplo.
NVL(val, exp)	Devuelve la expresión exp si *val* es NULL, y *val* si en otro caso.	select salario+nvl(comision,0) from emp;	450000, 350000...

Fuente: VEGAS, Jesús. Introducción al SQL. (online).

ü Funciones de grupo

Tabla 14. Funciones de grupo.

Función	*Descripción*	*Ejemplo*
COUNT(col)	Cuenta el número de filas agrupadas.	select count(nombre),oficio from emp group by oficio;
AVG(col)	Calcula el valor medio de todos los valores de la columna *col*.	select avg(salario),oficio from emp group by oficio;
MAX(col)	Calcula el valor máximo de todos los valores de la columna *col*.	select max(salario),oficio from emp group by oficio;
MIN(col)	Calcula el valor mínimo de todos los valores de la columna *col*.	select min(salario),oficio from emp group by oficio;
SUM(col)	Calcula la suma de los valores de la columna *col.*	select sum(salario), oficio from emp group by oficio;
STDDEV(col)	Calcula la desviación típica de los valores de la columna *col* sin tener en cuenta los valores nulos.	select stddev(salario), oficio from emp group by oficio;

VARIANCE(col)	Calcula la varianza de los valores de la columna *col* sin tener en cuenta los valores nulos.	select variance(salario), oficio from emp group by oficio;

Fuente: VEGAS, Jesús. Introducción al SQL. (online).

DATOS DE MULTIPLES TABLAS

Unir o relacionar dos tablas para realizar una consulta, se conoce con el término de JOIN. Existen cuatro tipos de joins:

- **Equijoin**

El join entre tablas se especifica mediante una igualdad (podría ser entre varias columnas).

EJEMPLO

```
SELECT cedula, nombre, nombre_dpto
FROM empleado e, departamento d
WHERE d.dpto = e.dpto
```

- **Non-equijoins**

La unión entre las tablas se hace mediante un operador distinto del "=".

EJEMPLO

```
SELECT e.nombre, e.sueldo, gs.grado
FROM empleado e, grado_salario gs
WHERE sueldo BETWEEN desde AND hasta;
```

- **Outer Joins**

Cuando se unen tablas y se quieren visualizar los maestros que no tienen un detalle.

EJEMPLO

```
SELECT nombre_dpto, nombre
FROM empleado e, departamento d
WHERE e.dpto(+) = d.dpto
```

- **Self Joins**

Para relaciones recursivas sobre una misma tabla.

EJEMPLO

```
SELECT t.nombre || ' trabaja para ' ||  j.nombre
FROM empleado t, empleado j
WHERE t.jefe = j.cedula;
```

SUBQUERIES

- Es un *SELECT* embebido en otro.
- Se utiliza para consultas complejas, hechas por partes e integradas mediante subqueries.
- Pueden emplearse en las cláusulas *FROM*, *WHERE* y *HAVING*.
- Los subqueries deben encerrarse entre ().
- Siempre se colocan a la derecha del operador de comparación.
- Se deben usar operadores sencillos para resultados individuales y operadores de conjunto para resultados múltiples.

- **Subqueries Sencillos**

El query embebido retorna un solo dato.

EJEMPLO

```
SELECT *
FROM empleado
WHERE sueldo > (SELECT sueldo
                  FROM empleado
                  WHERE cedula=91263541);
```

El query embebido devuelve el valor del sueldo del empleado cuya cédula es 91635541 para compararlo con los demás sueldos y saber quienes ganan más que él.

- **Subqueries Múltiples**

Retornan más de una fila.

Usa operadores especiales de comparación:

IN: es TRUE si el valor existe en el conjunto.

ANY: compara el valor para cada fila retornada por la subconsulta

ALL: compara el valor con todas las filas retornadas por la subconsulta.

Análisis Top N: selecciona los “n” registros que cumplan una condición.

- IN. indica pertenencia a un conjunto de valores o si se es miembro de una subconsulta.

EJEMPLO

```
SELECT *
FROM empleado
WHERE dpto IN (SELECT dpto
                  FROM departamento
                  WHERE region='ORI');
```

Devuelve los empleados ubicados en los dptos de la región oriental.

Puede cubrir varias columnas

```
SELECT *
FROM pagos
WHERE (banco, sucursal) IN (SELECT cod_banco, cod_suc
                                  FROM bancos
                                  WHERE abierto = 'S');
```

- ANY. compara un valor con cada valor devuelto por una subconsulta retornando verdadero si uno, cualquiera de ellos, cumple la condición. *Any* debe ir precedido de =, <>, <, >, >= o >=.

EJEMPLO

```
SELECT *
FROM empleado
WHERE sueldo < ANY (SELECT sueldo
                    FROM empleado
                    WHERE dpto = 20)
AND dpto != 20;
```

Empleados que no sean de Sistemas y que ganen menos que cualquiera de ellos.

- ALL. compara un valor con cada valor devuelto por una subconsulta retornando verdadero si todos ellos cumplen la condición. *All* debe ir precedido de =, <>, <, >, >= o >=.

EJEMPLO

```
SELECT *
FROM empleado
WHERE sueldo < ALL (SELECT sueldo
                    FROM empleado
                    WHERE dpto = 20)
AND dpto != 20;
```

Empleados que no sean de Sistemas y que ganen menos que todos ellos.

- Análisis Top N. selecciona los “n” registros que más!... (los que cumplen con una condición).

EJEMPLO

Cuáles son las 10 películas más alquiladas?.

```
SELECT c.nombre, c.genero, a.cuantos
FROM (SELECT pelicula, SUM(alquileres) cuantos
            FROM prestamos
           GROUP BY pelicula
           ORDER BY 2 DESC) a, cintas c
WHERE a.pelicula = c.pelicula
AND ROWNUM <= 10;
```

MANIPULACION DE DATOS

- DML (Data Manipulation Language) es el lenguaje utilizado para manipular información.
- Para alterar la información se utiliza las sentencias: insert, update, delete y merge.
- Todas las sentencias son controladas por la Integridad Referencial.
- Una transacción es un conjunto de operaciones DML que se ejecutan como una sola unidad.

- **INSERT**

La sentencia *INSERT* se utiliza para añadir registros a las tablas de la base de datos. Se pueden insertar datos de dos formas: Insertar un único registro ó Insertar en una tabla los registros contenidos en otra tabla.

El formato de la sentencia es:

```
INSERT INTO nombre_tabla [(nombre_columna, ...)]
VALUES (expr, ...)

INSERT INTO nombre_tabla [(nombre_columna, ...)]
SELECT ...
```

nombre_tabla nombre de la tabla a la cual se le insertan los datos.
nombre_columna es una lista opcional de nombres de campo en los que se insertarán valores en el mismo número y orden que se especificarán en la cláusula *VALUES*. Si no se especifica la lista de campos, los valores de *expr* en la cláusula *VALUES* deben ser tantos como campos tenga la tabla y en el mismo orden que se definieron al crear la tabla.
expr es una lista de expresiones o valores constantes, separados por comas, para dar valor a los distintos campos del registro que se añadirá a la tabla. Las cadenas de caracteres deberán estar encerradas entre comillas (‘ ').

EJEMPLO

```
INSERT INTO departamento (dpto, nombre,region)
VALUES (10, 'SISTEMAS', NULL);

INSERT INTO departamento
VALUES (&cod, &nom, &reg);
```

De esta última forma se insertan datos a una tabla sin necesidad de repetir toda la sentencias solo dándole los valores correspondientes a las variables cuantas veces sea necesario.

ü UPDATE

Actualiza los registros que satisfacen una condición. *UPDATE* es especialmente útil cuando se desea cambiar un gran número de registros o cuando éstos se encuentran en múltiples tablas. Se pueden cambiar varios campos a la vez.

Su formato es:

```
UPDATE nombre_tabla
SET nombre_columna = expr, ...
[WHERE { condición }]
```

nombre_tabla puede ser únicamente el nombre de la tabla.

nombre_columna es el nombre de columna o campo cuyo valor se desea cambiar. En una misma sentencia *UPDATE* pueden actualizarse varios campos de cada registro de la tabla.

expr es el nuevo valor que se desea asignar al campo que le precede. La expresión puede ser un valor constante o una subconsulta. Las cadenas de caracteres deberán estar encerradas entre comillas ('), las subconsultas entre paréntesis.

El valor de la condición de la cláusula *WHERE* puede ser especificado mediante un Subquery. Si en una consulta de actualización se suprime la cláusula *where* todos los registros de la tabla señalada serán actualizados.

EJEMPLO

```
UPDATE Empleados
SET Salario = Salario * 1.1

UPDATE linea_cuenta lc
SET sdo_Fact = (SELECT SUM(sdo_fact)
                  FROM SALDO s
                  WHERE s.linea = lc.linea),
    sdo_refc = (SELECT SUM(sdo_refc)
                  FROM SALDO s
                  WHERE s.linea = lc.linea)
WHERE lc.linea = 6344878;
```

ü DELETE

Elimina los registros de las tablas listadas en la cláusula *FROM* que satisfagan la cláusula *WHERE*. Esta consulta elimina los registros completos, no es posible eliminar el contenido de algún campo en concreto.

El formato de la sentencia es:

```
DELETE FROM nombre_tabla
[WHERE { condición }]
```

nombre_tabla nombre de la tabla a la cual se le borrarán los registros.

La cláusula *where* sigue el mismo formato que la vista en la sentencia *SELECT* y determina que registros se borrarán. La condición también puede ser especificada mediante un *subquery*.

Cada sentencia *DELETE* borra los registros que cumplen la condición impuesta o todos si no se indica cláusula *WHERE*.

ü MERGE

Consolidación de las sentencias *UPDATE* e *INSERT* en una sola operación.

El formato de la sentencia es:

```
MERGE INTO  nombre_tabla
USING condición de join
WHEN MATCHED THEN
      UPDATE nombre_tabla
      SET nombre_columna = expr, ...
WHEN NOT MATCHED THEN
      INSERT INTO nombre_tabla [(nombre_columna, ...)]
      VALUES (expr, ...)
```

EJEMPLO

```
MERGE INTO copia_empleados c
USING empleado e ON (c.cedula = e.cedula)
WHEN MATCHED THEN
      UPDATE SET c.sueldo = e.sueldo,
                  c.direccion = e.direccion
WHEN NOT MATCHED THEN
      INSERT INTO copia_empleado
      VALUES (e.cedula, e.nombre, e.direccion, e.sueldo);
```

En el anterior ejemplo se actualiza la tabla copia_empleados con los cambios efectuados sobre la tabla empleados.

- **Transacciones**

Una transacción es una serie de cambios en la base de datos que deben ser tratados como una solo, en otras palabras, que se realicen todos los cambios o que no se haga ninguno, de lo contrario se podrían producir inconsistencias en la base de datos. Cuando no se tiene activada una transacción el gestor de base de datos ejecuta inmediatamente cada sentencia *INSERT*, *UPDATE* o *DELETE* que se le encomiende, sin posibilidad de deshacer los cambio en caso de ocurrir cualquier percance.

Cuando se activa una transacción los cambios que se van realizando quedan en un estado de provisionalidad hasta que se realiza un *COMMIT*, el cual hará definitivos los cambios o hasta realizar un *ROLLBACK* que deshará todos los cambios producidos desde que se inició la transacción.

Las consultas de acción son aquellas que no devuelven ningún registro, son las encargadas de acciones como añadir y borrar y modificar registros. Tanto las sentencias de actualización como las de borrado desencadenan (según el motor de datos) las actualizaciones en cascada, borrados en cascada, restricciones y valores por defecto definidos para los diferentes campos o tablas afectadas por la consulta.

Objeto de las Transacciones

- Asegurar la consistencia de los datos.
- Permite a la sesión ver los cambios a manera de “preview” antes de hacerlos permanentes.
- Agrupa sentencias relacionadas lógicamente.

Control de la Transacción: una transacción puede estar compuesta por:

- Un conjunto de sentencias DML.
- UNA sentencia DDL.
- UNA sentencia DCL.

Estado de los datos ANTES DE

- Los datos previos pueden recuperarse con un ROLLBACK.
- Solo la sesión que hizo los cambios, puede ver su resultado.
- Las demás sesiones de la BD, observan los datos previos.
- Los registros alterados son marcados con candados para impedir su alteración por otras sesiones.

Estado de los Datos DESPUÉS DE

- Los cambios se hacen permanentes
- Los datos anteriores se pierden definitivamente.
- Todas las sesiones pueden ver los cambios.
- Se liberan todos los candados sobre los registros afectados.
- Si alguna sesión estaba bloqueada, continúa su ejecución normal.

Finalización de una Transacción

- Las sentencias COMMIT o ROLLBACK.
- Cualquier sentencia DDL o DCL.
- Saliéndose de SQL*Plus (exit).
- Error del Sistema (crash).

CREACION DE OBJETOS

Las sentencias DDL son las encargadas de crear objetos en la base de datos. Los objetos que se pueden crear son:

- TABLE: tabla de datos.
- VIEW (vistas): SELECT almacenado en la BD.
- INDEX: índices sobre tablas.
- SYNONYM: alias o sinónimos sobre tablas.

ü TABLAS

Crear

La sentencia para crear una tabla tiene la forma:

```
CREATE TABLE nombre_tabla
(definición_columna, ...)
```

nombre_tabla nombre que se le quiere dar a la tabla.

definición_columna esta compuesto por el nombre de la columna o campo, seguida del tipo de dato de dicha columna.

Modificar

Si después de crear una tabla se ve la necesidad de añadir una columna adicional o modificar la definición de una columna existente, se utiliza el comando *ALTER TABLE*.

```
ALTER TABLE nombre_tabla
{ADD | MODIFY | DROP} ({columna tipoColumna [NOT NULL]});
```

EJEMPLOS

```
ALTER TABLE empleados
ADD (sueldo_deseado NUMBER(8));
```

Inserta una columna en la tabla, al final del registro

```
ALTER TABLE empleado
MODIFY (sueldo_deseado NUMBER(10));
```

En una columna puede cambiarse el tamaño, el tipo de dato y el valor por default.

```
ALTER TABLE empleado
DROP COLUMN sueldo_deseado;
```

Una vez eliminada una columna no puede recuperar.

Al momento de modificar una tabla hay que tener en cuenta:

- No es posible disminuir el tamaño de un columna.
- En las modificaciones, los tipos anterior y nuevo deben ser compatibles o la tabla debe estar vacía.
- La opción ADD ... NOT NULL sólo será posible si la tabla está vacía.
- La opción MODIFY ... NOT NULL sólo podrá realizarse cuando la tabla no contenga ninguna fila con valor nulo en la columna en cuestión.

Eliminar

Esta operación se puede realizar con el comando *DROP TABLE*.

El formato para destruir o borrar un tabla es:

```
DROP TABLE nombre_tabla
```

nombre_tabla puede ser únicamente el nombre de la tabla que se eliminará.

EJEMPLO

```
DROP TABLE empleado;
```

- Los datos y la estructura de la tabla son borrados.
- Todos los índices y constraints son eliminados.
- Esta sentencia no tiene reversa.

- **VISTAS**

Una vista es un objeto que no contiene datos por sí mismo. Es una clase de tabla cuyos contenidos son tomados de otras tablas por medio de la ejecución de una consulta.

Una vista es como una ventana a través de la cual se puede consultar o cambiar información de la tabla a la que está asociada.

Las vistas tienen la misma estructura que una tabla: filas y columnas. La única diferencia es que sólo se almacena de ellas la definición, no los datos. Los datos que se recuperan mediante una consulta a una vista se presentarán igual que los de una tabla. De hecho, si no se sabe que se está trabajando con una vista, nada hace suponer que es así. Al igual que sucede con una tabla, se pueden insertar, actualizar, borrar y seleccionar datos en una vista.

Creación de una Vista

```
CREATE VIEW vista [({columna ,}] AS consulta ;
```

La vista se crea con las columnas que devuelve una consulta.

Borrado de una Vista

```
DROP VIEW vista ;
```

EJEMPLO

```
CREATE VIEW vrevistas AS
SELECT a.*, r.*
FROM activos a, revistas r
WHERE r.activo_clave = a.clave
DROP VIEW vrevistas;
```

ü ÍNDICES

Las tablas de una aplicación disponen de sus propios índices que no deben ser modificados ni destruidos. Sin embargo, se da la posibilidad de crear índices propios para las tablas o añadir otros a los de la aplicación.

Para crear un índice la sentencia es

```
CREATE INDEX nombre_indice
```

y para destruirlo

```
DROP INDEX nombre_indice
```

ü SINÓNIMOS

SQL permite crear un sinónimo para una tabla o vista. Esto supone que pueden utilizarse dos nombres diferentes para un mismo objeto.

Una primera utilidad de los sinónimos es la posibilidad de independizar las aplicaciones de los nombres físicos de las tablas que manejan. Así, las aplicaciones harán referencia a un sinónimo de tabla, que en cada caso puede estar asociado a una tabla distinta.

Otra utilidad es la posibilidad de que un usuario acceda a las tablas de otro usuario como si fueran suyas, siempre que tenga permiso para hacerlo, si al definir el sinónimo incluye el nombre del usuario en la denominación de la tabla.

Creación de un sinónimo:

```
CREATE SYNONYM sinonimo FOR [usuario.]{tabla | vista} ;
```

Borrado de un sinónimo:

```
DROP SYNONYM sinonimo ;
```

ANEXO F: MANUAL DE PL/SQL

INTRODUCCIÓN

PL/SQL (Procedural Language/SQL) es una extensión de SQL, que agrega ciertas construcciones propias de lenguajes procedimentales, obteniéndose como resultado un lenguaje estructural más poderoso que SQL.

Su utilización es dentro del administrador de bases de datos Oracle y sus principales características son la posibilidad que brinda de utilizar sentencias SQL para manipular datos en Oracle y sentencias de control de flujo para organizar esta manipulación de datos.

Dentro del lenguaje, es posible declarar constantes y variables, definir procedimientos y funciones y atrapar errores en tiempo de ejecución. Combina la el poder de la manipulación de datos con SQL y las facilidades del procesamiento de los mismos, tal como en los más modernos lenguajes de programación.

La unidad de programación utilizada por PL/SQL es el bloque. Todos los programas de PL/SQL están conformados por bloques. Típicamente, cada bloque lleva a cabo una acción lógica en el programa.

VENTAJAS EN LA UTILIZACIÓN DE PL/SQL

PL/SQL es un lenguaje de procesamiento de transacciones completamente portable y con un alto rendimiento, que proporciona las siguientes ventajas al ser utilizado:

ü Soporte para SQL

SQL se ha convertido en el lenguaje estándar de bases de datos por su flexibilidad de uso y facilidad de aprenderlo. Unos pocos comandos permiten la fácil manipulación de prácticamente toda la información almacenada en una base de datos.

SQL es no-procedural, lo cual significa que es Oracle quien se preocupará de cómo ejecutar de la mejor manera un requerimiento señalado en una sentencia SQL. No es necesaria la conexión entre varias sentencias porque Oracle las ejecuta de a una a la vez.

PL/SQL permite una completa manipulación de los datos almacenados en una base Oracle, proporciona comandos de control de transacciones y permite utilizar las funciones de SQL, operadores y seudo columnas. PL/SQL soporta tipos de datos de SQL, lo que reduce la necesidad de convertir los datos al pasar de una a otra aplicación. También soporta SQL dinámico, una avanzada técnica de programación que convierte a sus aplicaciones en más flexibles y versátiles.

ü Soporte para Programación Orientada a Objetos

Los objetos se han convertido en una herramienta ideal para modelar situaciones de la vida real. Con su utilización es posible reducir el costo y tiempo de construcción de aplicaciones complejas. Otra ventaja es que utilizando una metodología de este tipo es posible mantener diferentes equipos de programadores construyendo aplicaciones basadas en el mismo grupo de objetos.

Permitir el encapsulamiento del código en bloques es el primer paso para la implementación de métodos asociados a diferentes tipos de objetos construidos también con PL/SQL.

ü Mejor rendimiento

Sin PL/SQL, Oracle tendría que procesar las instrucciones una a una. Cada llamada produciría un *overhead* considerable, sobre todo si consideramos que estas consultas viajan a través de la red.

Por el contrario, con PL/SQL, un bloque completo de sentencias puede ser enviado cada vez a Oracle, lo que reduce drásticamente la intensidad de comunicación con la base de datos. Los procedimientos almacenados escritos con PL/SQL son compilados una vez y almacenados en formato ejecutable, lo que produce que las llamadas sean más rápidas y eficientes. Además, ya que los procedimientos almacenados se ejecutan en el propio servidor, el tráfico por la red se reduce a la simple llamada y el envío de los parámetros necesarios para su ejecución. El código ejecutable se almacena en caché y se comparte a todos los usuarios, redundando en mínimos requerimientos de memoria y disminuyendo el *overhead* al mínimo.

- Alta productividad

Si se decide utilizar otros productos de Oracle como Oracle Forms y Oracle Reports, es posible integrar bloques completos de PL/SQL en un trigger de Oracle Forms, debido a que PL/SQL es el mismo en todos los ambientes.

- Completa portabilidad

Las aplicaciones escritas con PL/SQL son portables a cualquier sistema operativo y plataforma en la cual se encuentre corriendo Oracle. En otras palabras, PL/SQL corre donde se encuentre corriendo Oracle también. Esto significa que se pueden codificar librerías que podrán ser reutilizadas en otros ambientes.

- Integración con Oracle

PL/SQL y los lenguajes SQL en general se encuentran perfectamente integrados. PL/SQL soporta todos los tipos de datos de SQL. Los atributos %TYPE y %ROWTYPE integran PL/SQL con SQL, permitiendo la declaración de variables basado en tipos de columnas de tablas de la base de datos. Lo anterior provee independencia de los datos y permite a los programas adaptarse a los cambios en la base de datos para cumplir con las nuevas necesidades del negocio.

- Seguridad

Los procedimientos almacenados construidos con PL/SQL habilitan la división de la lógica del cliente con la del servidor. De esta manera, se previene que se efectúe manipulación de los datos desde el cliente. Además, se puede restringir el acceso a los datos de Oracle, permitiendo a los usuarios la ejecución de los procedimientos almacenados para los cuales tengan privilegios solamente.

BLOQUES PL/SQL

Un bloque PL/SQL es la unidad básica de todo programa. Todos los programas PL/SQL están compuestos por bloques, los cuales pueden ser secuenciales o anidados. Los tipos de bloques son:

- Bloques anónimos: se construyen generalmente de forma dinámica y sólo se ejecutan una vez.

- Bloques con nombre: son como los bloques anónimos, pero tienen una etiqueta que le da al bloque un nombre.

- Subprogramas: son paquetes, procedimientos y funciones que son almacenados en base de datos y se ejecutan cuando son invocados.

- Triggers: son bloques con nombre que son almacenados en la base de datos. Son ejecutados implícitamente cuando cierto evento ocurre.

ESTRUCTURA DE UN BLOQUE

Un bloque permite agrupar en forma lógica un grupo de sentencias. De esta manera se pueden efectuar declaraciones de variables que sólo tendrán validez en los bloques donde éstas se definan.

Los bloques PL/SQL presentan una estructura específica compuesta de tres partes bien diferenciadas:

- La *sección declarativa* en donde se declaran todas las constantes y variables que se van a utilizar en la ejecución del bloque.
- La *sección de ejecución* que incluye las instrucciones a ejecutar en el bloque PL/SQL.
- La *sección de excepciones* en donde se definen los manejadores de errores que soportará el bloque PL/SQL.

Cada una de las partes de un bloque se delimita por una palabra reservada, de modo que un bloque PL/SQL se puede representar de la siguiente manera:

```
DECLARE
/* Parte Declarativa */
BEGIN
/* Parte de Ejecución */
EXCEPTION
/* Parte de Excepciones */
END;
```

Solo se requiere que aparezca la sección ejecutable. Las demás son opcional. Las únicas instrucciones SQL permitidas en un bloque PL/SQL son INSERT, UPDATE, DELETE y SELECT, además de algunas instrucciones para manipulación de datos e instrucciones para control de transacciones. Otras instrucciones de SQL como DROP, CREATE o ALTER no son permitidas. PL/SQL no es *case sensitive* por lo que no hay distinción entre nombres con mayúsculas y minúsculas.

Ejemplo de un bloque PL/SQL genérico:

```
DECLARE
nombre_variable VARCHAR2(5);
nombre_excepcion EXCEPTION;
```

```
BEGIN
      SELECT nombre_columna
      INTO nombre_variable
      FROM nombre_tabla;
EXCEPTION
WHEN nombre_excepcion THEN ...
END;
```

Sección Declarativa

En esta parte se declaran todos los tipos de datos, las constantes y variables utilizadas en el bloque de ejecución. También se declaran cursores, de gran utilidad para la consulta de datos y excepciones definidas por el usuario.

ü Variables y constantes

PL/SQL permite declarar constantes y variables para ser utilizadas en cualquier expresión dentro de un programa. La única condición exigida por PL/SQL es que cada variable (o constante) debe estar declarada antes de ser utilizada en una expresión.

La sintaxis para declarar una variable o una constante es:

```
nombre_variable [CONSTANT] tipo [NOT NULL] [:= valor_inicial];
```

tipo puede ser: tipo_escalar (NUMBER | DATE | CHAR | VARCHAR | BOOLEAN), identificador%TYPE o identificador%ROWTYPE.

La cláusula *constant* indica la definición de una constante cuyo valor no puede ser modificado. Se debe incluir la inicialización de la constante en su declaración.

La cláusula *not null* impide que a una variable se le asigne el valor nulo y por tanto debe inicializarse a un valor diferente de *null*.
Las variables que no son inicializadas toman el valor inicial *null*.

Los tipo de datos *boolean* pueden tomar los valores *true*, *false* y *null* y se suele utilizar para almacenar el resultado de alguna operación lógica. Por su parte, *varchar* es un sinónimo de *char*.

También es posible definir el tipo de una variable o constante, dependiendo del tipo de otro identificador, mediante la utilización de las cláusulas %TYPE y %ROWTYPE. Mediante la primera opción se define una variable o constante escalar y con la segunda se define una variable fila, donde identificador puede ser otra variable fila o una tabla.

EJEMPLO

```
DECLARE
v_location VARCHAR2(15) := ’Granada’;
c_comm CONSTANT NUMBER(3) := 160;
v_nombre tabla_empleados.nombre%TYPE;
```

Cursores

El resultado de una consulta puede almacenarse en variables, en las que se almacenan cada una de las tuplas del resultado o bien en una variable de tupla que sea compatible con el resultado de la consulta. Si aparece más de una fila como resultado de una consulta, resulta conveniente la utilización de cursores que permiten recorrer todas sus filas.

Los cursores son áreas de trabajo que permiten ejecutar sentencias SQL y procesar la información obtenida de ellos.

Hay dos tipos de cursores: implícitos y explícitos. PL/SQL declara implícitamente un cursor para todas las sentencias de manipulación de datos, incluyendo las consultas que retornan sólo una fila. Para consultas que devuelven más de una fila, es posible declarar explícitamente un cursor que procese las filas en forma individual.

La sintaxis para declarar un cursor es:

```
CURSOR nombre_cursor [parámetros] IS
consulta_SQL;
```

La consulta no debe contener la cláusula *into*.

EJEMPLO

```
DECLARE
CURSOR emp_cursor IS
      SELECT empnombre, empcargo
      FROM empleados;
```

Los parámetros de un cursor se pueden utilizar para definir variables con valores de entrada que determinen el resultado de cada ejecución de la consulta SQL asociada.

Excepciones

PL/SQL provee una fácil manera de detectar y procesar ciertas condiciones de error predefinidas (o definidas por el usuario), llamadas excepciones.

Cuando ocurre un error se procesa una excepción, esto es, se detiene la ejecución normal del programa y se transfiere el control a un segmento especial del programa que tiene por objeto manejar estas situaciones excepcionales. Estas rutinas que se codifican en forma separada se conocen con el nombre de *exception handlers*.

En PL/SQL existe un conjunto de excepciones predefinidas que informan de los errores producidos en la ejecución de las sentencias SQL por parte del sistema de gestión de bases de datos. Además de éstas, el programador puede definir excepciones de uso específico, utilizando la sentencia *raise*.

La manera de definir las excepciones es:

```
nombre_excepción EXCEPTION;
```

Las excepciones no son variables sino que su utilización debe realizarse mediante sentencias específicas de PL/SQL. Tampoco pueden ser utilizadas como argumentos en funciones ni procedimientos.

EJEMPLO

```
DECLARE
demasiados_empleados EXCEPTION;
```

Sección de Ejecución

Como todo lenguaje de programación, en PL/SQL se pueden distinguir tres clases de instrucciones: instrucciones de asignación, instrucciones de control de flujo y bucles.

Instrucciones de Asignación

La forma de asignarle un valor a una variable es la siguiente:

```
variable_objetivo := expresión_PL/SQL;
```

Las *expresiones PL/SQL* pueden incluir literales, variables y constantes definidas en el bloque, así como funciones aplicadas sobre literales, constantes y variables.

Los literales son similares a los utilizados en SQL, es decir:

- Las cadenas de caracteres se delimitan por la comilla simple.
- Los números reales pueden especificarse tanto en formato decimal como científico.
- Operadores sobre números: +, -, *, /, ** (exponencial), MOD (resto).
- Operadores sobre cadenas: || (concatenación).
- Operadores lógicos: AND, OR, NOT.
- Operadores sobre cursores: %ROWCOUNT, %NOTFOUND, %FOUND, %ISOPEN.

Los valores lógicos aparecen como resultado de alguna comparación o verificación de valor. En PL/SQL se pueden utilizar los comparadores definidos en SQL:

- Comparadores clásicos: <, <=, =, !=, ^=, <, =
- Comparadores SQL: [NOT] LIKE, IS [NOT] NULL, [NOT] BETWEEN..AND.., [NOT] IN.

Las funciones definidas en SQL también aparecen en PL/SQL, además existen funciones propias del lenguaje:

- Funciones sobre cadenas de caracteres: ASCII, CHR, INITCAP, INSTR, LENGTH, LOWER, LPAD, LTRIM, REPLACE, RPAD, RTRIM, SOUNDEX, SUBSTR, TRANSLATE, UPPER.
- Funciones numéricas: ABS, CEIL, FLOOR, MOD, POWER, ROUND, SIGN, SQRT, TRUNC.
- Funciones sobre fechas: ADD_MONTHS, LAST_DAY, MONTHS_BETWEEN, NEW_TIME, NEXT_DAY, ROUND, SYSDATE, TRUNC.
- Funciones de conversión: TO_CHAR, TO_DATE, TO_NUMBER.
- Funciones de control de errores: SQLCODE, SQLERRM.
- Funciones varias: UID, USER, DECODE, GREATEST, LEAST, NVL, USERENV.

Por lo que respecta a las asignaciones asociadas a las sentencias SQL, existen dos alternativas, asignar el resultado a una lista de variables o a un cursor. Si se utiliza una sentencia *select* para realizar una asignación a una lista de variables, la consulta asociada sólo debe dar como resultado una única fila, ya que en caso contrario se genera una excepción. Por esta razón, cuando no se conoce a priori el número de filas del resultado, resulta más conveniente utilizar cursores.

Para asignar el resultado de una sentencia *select* a una lista de variables se utiliza la sintaxis:

```
SELECT lista_select
INTO lista_variables
FROM ...
WHERE...;
```

EJEMPLO

```
DECLARE
v_numdep NUMBER(2);
v_local VERCHAR2(15);

BEGIN

      SELECT numdep, local I
      NTO v_numdep, v_local
      FROM departamentos
      WHERE nombre='Informática'; --Seguro que sólo devuelve una fila

END;
```

El número de variables escalares en la lista de variables debe corresponder con el número de atributos del *select* o cursor asociado. La lista de variables puede sustituirse por una variable fila del tipo correspondiente.

Para asignar el resultado de una sentencia *select* a un cursor se utiliza la sintaxis :

```
DECLARE
CURSOR cur_emp IS
      SELECT num_empleado, nom_empleado
      FROM empleado;
emp_registro cur_emp%ROWTYPE;

BEGIN
OPEN cur_emp;
      LOOP
      FETCH cur_emp INTO emp_registro;
      EXIT WHEN cur_emp%NOTFOUND;
```

```
      ...
      END LOOP;
CLOSE cur_emp;
END;
```

Utilizando la sentencia *FOR* se pueden implementar operaciones con cursores de manera más fácil, ya que no es necesario declarar la variable que lo recorre, ni abrir / cerrar el cursor.

Instrucciones de Control de Flujo

En PL/SQL es posible ejecutar un bloque de instrucciones u otro en función del valor de alguna expresión lógica, mediante la utilización de la sentencia IF:

```
IF expresión_lógica THEN intrucciones_PL/SQL;
      [ELSIF expresión_lógica THEN intrucciones_PL/SQL;]
      [ELSE intrucciones_PL/SQL;]
END IF;
```

Tal y como aparece en la sintaxis de esta sentencia, se pueden presentar diferentes alternativas, aunque al menos deben presentarse las cláusulas IF ... THEN ... END IF.

Bucles

Los bucles permiten repetir un número de veces un conjunto de instrucciones PL/SQL. En PL/SQL los bucles se identifican con la cláusula LOOP, pudiéndose presentar cuatro sintaxis diferentes:

```
LOOP   intrucciones_PL/SQL   END LOOP;
WHILE  expresión_lógica   LOOP   intrucciones_PL/SQL   END LOOP;
FOR    control_numérico   LOOP   intrucciones_PL/SQL   END LOOP;
FOR    control_cursor     LOOP   intrucciones_PL/SQL   END LOOP;
```

donde *control_numérico* corresponde a:

```
índice IN [REVERSE] expr_entera .. expr_entera
```

donde *control_cursor* corresponde a:

```
variable_fila IN cursor | variable_fila IN sentencia_SQL
```

La finalización del bucle básico (*LOOP*) debe forzarse mediante la inclusión de la sentencia EXIT:

```
EXIT [WHEN expresión_lógica];
```

La utilización de los bucles numéricos y sobre cursores presenta efectos laterales de gran interés:

- El índice asociado a un bucle numérico no necesita declararse, sino que se declara de modo implícito. En el caso de declararse otra variable con el mismo nombre, ambas variables serían diferentes. Además, al final del bucle aparece un incremento, o decremento, implícito del índice.

- En un bucle sobre cursores también se declara de modo implícito la variable fila asociada. Asimismo, se realiza un OPEN del cursor al entrar en el bucle y un CLOSE al salir, aunque la salida se produjera mediante la utilización de la sentencia EXIT y se produce un FETCH implícito en cada iteración del bucle.

EJEMPLO

```
DECLARE
CURSOR cur_emp IS
      SELECT num_empleado, nomb_empleado, edad
      FROM empleado;
BEGIN
FOR emp_registro IN cur_emp LOOP
      -- Apertura y declaración del cursor ímplicitamente
      IF emp_registro.edad > 30 THEN ...
      END LOOP;
      -- Cierre ímplicito del cursor
END;
```

Sección de Excepciones

En PL/SQL una advertencia o condición de error es llamada una excepción. Estas pueden ser definidas en forma interna (en tiempo de ejecución de un programa) o explícitamente por el usuario. Ejemplos de excepciones definidas en forma interna son la división por cero y la falta de memoria en tiempo de ejecución. Estas mismas condiciones excepcionales tienen sus nombres propios y pueden ser referenciadas con ellos: zero_divide y storage_error.

Cuando ocurre un error se alcanza la excepción, esto quiere decir que se ejecuta la porción del programa donde ésta se encuentra implementada, transfiriéndose el control a ese bloque de sentencias. Las excepciones definidas por el usuario deben ser alcanzadas explícitamente utilizando la sentencia raise.

Con las excepciones ser pueden manejar los errores cómodamente sin necesidad de mantener múltiples chequeos por cada sentencia escrita. También provee claridad en el código desde el momento en que permite mantener las rutinas correspondientes al tratamiento de los errores en forma separada de la lógica del negocio.

- **Excepciones predefinidas**

Las excepciones predefinidas no necesitan ser declaradas. Simplemente se utilizan cuando estas son llamadas por un determinado error.

Existe un conjunto de excepciones predefinidas por Oracle cuyos nombres aparecen a continuación agrupados por su funcionalidad:

NO_DATA_FOUND, TOO_MANY_ROWS: ocurren cuando un select no selecciona nada o selecciona varias filas cuando sólo se esperaba una.

NVALID_NUMBER, VALUE_ERROR, ZERO_DIVIDE, DUP_VAL_ON_INDEX: las tres primeras situaciones se producen por operaciones invalidas de tratamiento de números y la última cuando se intenta insertar una clave primaria duplicada.

CURSOR_ALREADY_OPEN, INVALID_CURSOR: la primera situación ocurre al intentar abrir un cursor ya abierto y la segunda al intentar hacer una operación invalida sobre un cursor

PROGRAM_ERROR, STORAGE_ERROR, TIMEOUT_ON_RESOURCE: detectan errores de almacenamiento o de ejecución.

- **Excepciones definidas por el usuario**

PL/SQL permite al usuario definir sus propias excepciones, las que deberán ser declaradas y llamadas explícitamente utilizando otros comandos del lenguaje.

- **Declaración**

Las excepciones sólo pueden ser declaradas en el segmento *Declare* de un bloque, subprograma o paquete. Se declara una excepción escribiendo su nombre seguida de la palabra clave *EXCEPTION*. Las declaraciones son similares a las de variables, pero una excepción es una condición de error, no un ítem de datos. Aun así, las mismas reglas de alcance aplican tanto sobre variables como sobre las excepciones.

- **Reglas de Alcance**

Una excepción no puede ser declarada dos veces en un mismo bloque. Tal como las variables, una excepción declarada en un bloque es local a ese bloque y global a todos los sub-bloques que comprende.

- **Sentencia "RAISE"**

La sentencia *RAISE* permite llamar una excepción en forma explícita. Es factible utilizar esta sentencia en cualquier lugar que se encuentre dentro del alcance de la excepción.

EJEMPLO

```
DECLARE
out_of_stock      EXCEPTION;  -- declaración de la excepción
total             NUMBER(4);
BEGIN
   IF total < 1 THEN
      RAISE out_of_stock; -- llamado a la excepción
   END IF;

EXCEPTION
   WHEN out_of_stock THEN
      -- manejar el error aquí
   WHEN OTHERS THEN
      -- otros posibles errores
END;
```

Finalmente, cabe destacar la existencia de la excepción *OTHERS*, que simboliza cualquier condición de excepción que no ha sido declarada. Se utiliza comúnmente al final del bloque de excepciones para absorber cualquier tipo de error que no ha sido previsto por el programador. En ese caso, es común observar la sentencia *ROLLBACK* en el grupo de sentencias de la excepción o alguna de las funciones *SQLCODE – SQLERRM*.

ü Uso de SQLCODE y SQLERRM

Al manejar una excepción es posible apoyarse con las funciones predefinidas *SQLCode* y *SQLerrm* para aclarar al usuario la situación de error acontecida.

SQLCode siempre retornará el número del error de Oracle y un “0” (cero) en caso exitoso al ejecutarse una sentencia SQL.

Por otra parte, *SQLerrm* retornará el correspondiente mensaje de error para la situación ocurrida. También es posible entregarle a la función *SQLerrm* un número negativo que represente un error de Oracle y ésta devolverá el mensaje asociado.

EJEMPLO

```
DECLARE
   err_num  NUMBER;
   err_msg  VARCHAR2(100);
BEGIN
   …
EXCEPTION
   WHEN OTHERS THEN
      err_num := SQLCODE;
      err_msg := SUBSTR(SQLERRM, 1, 100);
      INSERT INTO errores VALUES(err_num, err_msg);
END;
```

TIPOS DE DATOS ESTRUCTURADOS

Entre los tipos de datos estructurados que proporciona PL/SQL están los registros *RECORD* y los vectores *TABLE* y *VARRAY*. Se declaran en la sección *DECLARE*.

Para declarar un tipo registro se emplea la siguiente sintaxis:

```
TYPE nombre_tipo IS RECORD (campo [, campo] ...);
```

EJEMPLO

```
TYPE tipo_empleado_reg IS RECORD (nombre VARCHAR2(10),
                                  puesto VARCHAR2(8),
                                  sueldo NUMBER(6));

v_empleado_reg tipo_empleado_reg;
```

Alternativamente se puede indicar que el tipo de la variable sea el de los registros de una tabla existente:

```
v_empleado_reg empleado%ROWTYPE;
```

Para declarar los vectores se emplea la siguiente sintaxis:

```
TYPE nombre_tipo IS VARRAY (tamaño_maximo) OF tipo_datos [NOT NULL];
TYPE nombre_tipo IS TABLE OF tipo_datos [NOT NULL];
```

En ambos casos los índices se empiezan a contar a partir de 1, y para poder utilizar los vectores, deben ser previamente creados vacíos o con elementos. A partir de entonces para insertar elementos adicionales se tienen que extender, como muestra el siguiente ejemplo:

```
DECLARE
TYPE t_varray IS VARRAY (50) OF empleado.nombre%TYPE;
v_varray1 t_varray;
v_varray2 t_varray;
BEGIN
...
v_varray1 := t_varray('Ana', 'Lola');-- se crea con dos elementos
v_varray1.EXTEND;
v_varray1(3) := 'Luis';
/* v_varray1(4) := 'Juan'; Esto sería un error porque no se ha
                                extendido*/
v_varray2 := t_varray(); -- se crea vacío
IF v_varray2 IS NULL -- cierto
THEN v_varray2 := v_varray1; -- asignación de vectores
....
END;
```

Una diferencia que hay entre *TABLE* y *VARRAY* es que en el primer caso el vector puede crecer ilimitadamente, pero en el segundo caso solo puede crecer hasta el tamaño máximo definido.

Otra diferencia es que en los tipos *VARRAY* no se pueden borrar elementos, por lo que sus posiciones se deben ocupar consecutivamente. Sin embargo, en los tipos *TABLE* se pueden borrar elementos con la instrucción *DELETE*, pudiendo quedar huecos intermedios vacíos y sin poderse referenciar, aunque se pueden volver a llenar con una asignación. La función *EXISTS* nos permite saber si un elemento se puede referenciar o ha sido borrado.

PAQUETES

Un paquete es un esquema u objeto que agrupa tipos de PL/SQL relacionados, ítems y subprogramas. Los paquetes se constituyen de dos partes: la especificación y el cuerpo.

La especificación es la interfaz con las aplicaciones. En ella es posible declarar los tipos, variables, constantes, excepciones, cursores y subprogramas disponibles para su uso posterior. El cuerpo define completamente a cursores y subprogramas e implementa lo que se declaró inicialmente en la especificación.

Es posible depurar y modificar cuantas veces se desee el cuerpo de un paquete sin necesidad de alterar por ello la especificación del mismo.
Las declaraciones del los subprogramas en la especificación y en el cuerpo del paquete tienen que ser idénticas.

Para hacer referencia a los tipos, objetos y subprogramas declarados dentro de un paquete debe emplearse la notación siguiente:

```
package_nombre.type_name
package_nombre.objeto_name
package_nombre.subprograma_name
```

La sintaxis para crear un paquete es:

```
CREATE [OR REPLACE] PACKAGE nombre_paquete IS
especificación de procedimiento o funcion
|declaración de variable
|declaración de tipo
|declaración de excepcion
|declaración de cursor
END nombre_paquete;

CREATE [OR REPLACE] PACKAGE BODY nombre_paquete IS
cuerpo del procedimiento
END nombre_paquete;
```

Después de compilar la especificación y el cuerpo de un paquete quedan almacenados en el diccionario de datos y se pueden invocar en cualquier momento a sus elementos precediéndolos del nombre del paquete y un punto.

EJEMPLO

```
CREATE OR REPLACE PACKAGE paquete_cont IS
g_cont NUMBER := 0; -- se inicializa la variable global
PROCEDURE reset_cont (v_nuevovalor IN NUMBER);
END paquete_cont;
CREATE OR REPLACE PACKAGE BODY paquete_cont IS
PROCEDURE reset_cont (v_nuevovalor IN NUMBER) IS
BEGIN
g_cont := v_nuevovalor;
END reset_cont;
END paquete_cont;
```

PROCEDIMIENTOS

Un procedimiento almacenado es un conjunto de instrucciones en PL/SQL que pueden ser llamados usando el nombre que se le haya asignado.

La sintaxis para crear un procedimiento es la siguiente:

```
CREATE [OR REPLACE] PROCEDURE nombre [(param [IN|OUT|IN OUT|] datatype) .
. .] [IS]
   [declaraciones_locales]
BEGIN
   sentencias_ejecutables
[EXCEPTION
   condiciones_de_excepción]
END [nombre] ;
```

El uso de *OR REPLACE* permite sobrescribir un procedimiento existente. Si se omite, y el procedimiento ya existe, se producirá un error. Los modificadores in, out, in out indican si el parámetro es de entrada, salida o ambos.

Un procedimiento posee dos partes: una especificación y un cuerpo. La especificación es simple, comienza con la palabra *PROCEDURE* y termina con el nombre del procedimiento o la lista de parámetros (que es opcional).

El cuerpo del procedimiento comienza con la palabra reservada *IS* y termina con *END*, seguido opcionalmente por el nombre del procedimiento.

El siguiente ejemplo ilustra cómo declarar y ejecutar un procedimiento:

```
CREATE OR REPLACE PROCEDURE consulta_emp
(v_id IN empleado.numemp%TYPE,
v_nombre OUT empleado.nombre%TYPE,
v_sueldo OUT empleado.sueldo%TYPE,
v_comis OUT empleado.comision%TYPE)
IS
BEGIN
      SELECT nombre, sueldo, comision
      INTO v_nombre, v_sueldo, v_comis
      FROM empleado
      WHERE numemp=v_id;
END consulta_emp;
```

FUNCIONES

Una función es un conjunto de instrucciones en PL/SQL, que pueden ser llamados usando el nombre con que se le haya creado. Se diferencian de los procedimientos, en que las funciones retornan un valor al ambiente desde donde fueron llamadas.

La sintaxis para crear una función es la siguiente:

```
CREATE [OR REPLACE] FUNCTION name [(param [IN] datatype) . . .]
RETURN datatype [IS]
pl/sql_subprogram
```

El uso de *OR REPLACE* permite sobrescribir una función existente. Si se omite, y la función ya existe, se producirá, un error. El único modificador permitido para los parámetros es in, y si se omite, se tomará por defecto. Es decir, solo se permiten parámetros de entrada.

La función también posee una especificación y un cuerpo. El segmento de especificación comienza con la palabra *FUNCTION* y termina con la cláusula *RETURN*, la cual especifica el tipo de dato retornado por la función.
El cuerpo comienza con la palabra *IS* y termina con la palabra *END*, es decir, incluye las secciones de declaraciones, sentencias ejecutables y una parte opcional de manejo de excepciones.

La estructura *RETURN* completa inmediatamente la ejecución del subprograma y regresa el control al programa que la llamó.

EJEMPLO

```
FUNCTION revisa_salario (salario REAL,
                            cargo CHAR(10))
          RETURN BOOLEAN IS
salario_minimo REAL;
salario_maximo REAL;

BEGIN
   SELECT lowsal, highsal
   INTO salario_minimo, salario_maximo
   FROM salarios
   WHERE job = cargo ;
RETURN (salario >= salario_minimo) AND (salario <= salario_maximo)

END revisa_salario ;
```

Esta misma función de ejemplo puede ser llamada desde una sentencia PL/SQL que reciba un valor booleano, como por ejemplo, en:

```
DECLARE
renta_actual REAL;
codcargo CHAR(10);
BEGIN
   …
   IF revisa_salario (renta_actual, codcargo) THEN
   …
```

La función revisa_salario actúa como una variable de tipo booleano, cuyo valor depende de los parámetros recibidos.

TRIGGERS

Un trigger es un bloque PL/SQL asociado a una tabla o vista, que se ejecuta cuando una determinada instrucción en SQL se va a ejecutar sobre ella.

La sintaxis para crear un trigger es la siguiente:

```
CREATE [OR REPLACE] TRIGGER
{BEFORE |AFTER |INSTEAD OF}
{DELETE|INSERT|UPDATE [OF col1, col2, . . ., colN]
[OR {DELETE|INSERT|UPDATE [OF col1, col2, . . ., colN]. . .]}
ON table | vista
[REFERENCING OLD AS oldname, NEW as newname]
[FOR EACH [STATEMENT | ROW [WHEN (condition)]]]
bloque pl/sql
```

El uso de *OR REPLACE* permite sobrescribir un trigger existente. Si se omite, y el trigger existe, se producirá, un error.

Después del nombre del trigger, hay que indicar si se tiene que ejecutar antes, después o en lugar de la ejecución del suceso (por ejemplo la sentencia SQL (delete, insert, update)) que ha causado su disparo. Un mismo trigger puede tener

varios sucesos asociados, y una instrucción SQL sólo se corresponde con un suceso, aunque afecte a varias tuplas.

Los triggers de tipo *INSTEAD OF* solamente pueden definirse sobre vistas (no sobre tablas) y se ejecutan en lugar de la instrucción que los lance. Estos son siempre de tipo *FOR EACH ROW*, por lo que se pueden utilizar junto con la cláusula *WHEN* y las variables *OLD* y *NEW*.

A continuación va la palabra *ON* y el nombre de la *tabla o vista* a la que se asocia.

Las variables externas *OLD* y *NEW* que almacenan respectivamente los valores de cada tupla antes y después de ejecutar sobre ella la instrucción que lanza al trigger. Sin embargo, estas variables solamente se pueden utilizar cuando se escoge la opción de ejecutarlo una vez por cada tupla (*FOR EACH ROW*), es decir, ejecutarlo separadamente para cada una de las tuplas afectadas por la instrucción SQL que lo ha activado. En este caso la cláusula *WHEN* sirve para especificar una condición adicional que debe cumplir una tupla afectada por el trigger para que éste sea ejecutado sobre ella.

Pueden ser asociados un máximo de 12 triggers con una tabla de la base de datos.

Los triggers pueden ser empleados para :

- **ü** Verificar la modificación de datos en una tabla.
- **ü** Transparencia de algún evento.
- **ü** Derivación de valores en columnas de manera automática.
- **ü** Implementación de sistemas de seguridad complejos.
- **ü** Mantenimiento general de las tablas.

EJEMPLO

```
CREATE OR REPLACE TRIGGER Sal_Total
AFTER INSERT OR UPDATE OF salario ON empleado
FOR EACH ROW WHEN (new.dep_num IS NOT NULL)
/*El disparador solamente se va a ejecutar para aquellas tuplas
insertadas o modificadas que tengan un dep_num no nulo*/
BEGIN
      UPDATE departamento
      SET sal_total= sal_total + :new.salario
      WHERE dep_num = :new.dep_num;
END;
```

Nótese que las variables externas *OLD* y *NEW* deben ser precedidas por dos puntos para que puedan ser accedidas desde dentro del bloque PL/SQL. En el caso de que el trigger se tenga que ejecutar una sola vez para la instrucción que causa su disparo (*for each statement*), entonces la cláusula *WHEN* y las variables *OLD* y *NEW* no se pueden utilizar, ya que la instrucción puede afectar a varias tuplas.

Para activar un trigger se tiene que compilar con éxito la instrucción:

```
ALTER TRIGGER nombre_trigger [DISABLE|ENABLE]
```

Para borrar un trigger se ejecuta la acción:

```
DROP TRIGGER nombre_trigger
```

Para consultar la información de los triggers se puede ejecutar la siguiente consulta sobre el diccionario de datos:

```
SELECT trigger_type, table_name, triggering_event
FROM user_triggers
WHERE trigger_name = '...';
```

ANEXO G: MANUAL DE CARTRIDGE DE PL/SQL

Las herramientas web de PL/SQL es un grupo de paquetes suministrados por Oracle para el desarrollo de aplicaciones de web. Estos paquetes son usados para generar HTML dinámicamente, realizar operaciones de texto y mejorar la productividad del desarrollo.

USOS FRECUENTES

- Listados: cuando es necesario tener a disposición vía web el listado de tablas sin necesidad de tener instalado ningún producto de Oracle en el ordenador en el que se realiza la consulta.

- Formularios: cuando es necesario obtener la entrada de datos del usuario. Los formularios son la mejor forma que se tiene para recoger información por parte de los visitantes de una página.

- Cookies: en ocasiones es necesario que cierta información se guarde en el ordenador del cliente. Los cookies son pequeños archivos que envían los servidores con información que interesa esten en el lado del cliente.

LLAMADO A UNA PÁGINA

Para invocar el cartridge de PL/SQL el URL debe estar en el siguiente formato:

```
http://host_y_nombre_dominio[:puerto]/dirección_virtual/
      [paquete.]nom_procedimiento[?parámetro_entrada]
```

donde:

host_y_nombre_dominio especifica el dominio y la máquina en la cual el servidor del web está funcionando.

puerto especifica el puerto del servidor del web. Si se omite, se asume el puerto 80.

dirección_virtual especifica una trayectoria virtual tras el cartridge de PL/SQL. El primer elemento en la trayectoria especifica el agente PL/SQL a utilizar. Por ejemplo, si la trayectoria virtual *es /público/plsql*, el nombre del agente PL/SQL será *público*. Es necesario por lo menos, especificar un agente PL/SQL en la trayectoria virtual.

paquete especifica el paquete que contiene el procedimiento que se está llamando. Si se omite, el procedimiento es independiente.

nom_procedimiento especifica el procedimiento almacenado para funcionar. No puede ser una función almacenada.

parámetro_entrada especifica los parámetros del procedimiento almacenado. La secuencia sigue el formato del método GET. Por ejemplo, los parámetros múltiples se separan con el carácter & y los caracteres de espacio en los valores que se pasan se substituyen por el caracter +.

EJEMPLO

http://www.acme.com:9000/hr/plsql/

      get_emp?fname='john'&lname='doe'&role='office+manager'

PAQUETES CONTENIDOS

Los paquetes contenidos en este grupo de herramientas son:
HTP: Genera HTML.
HTF: Analiza HTML.
OWA_COOKIE: Almacenamiento de cookies.
OWA_IMAGE: Manejo de imágenes.
OWA_OPT_LOCK: Realiza cierre de registros.
OWA_PATTERN: Maneja búsquedas y sustituye texto.
OWA_SEC: Maneja la seguridad.
OWA_TEXT: Representa y muestra texto.
OWA_UTIL: Mejora la productividad.

Paquetes HTP y HTF. Estos paquetes son las principales bibliotecas usadas en la generación de salidas para las páginas web dinámicas. HTP es la biblioteca de procedimientos de hipertexto y HTF es la biblioteca de funciones de hipertexto.

HTP es un paquete que se utiliza para crear las páginas HTML desde PLSQL. Con él se crean tablas, listas, formularios, etc. Los procedimientos de este paquete se pueden agrupar en las siguientes categorías:

- Procedimientos cuya estructura colocan las partes principales del documento HTML.
- Procedimientos que son usados en la sección <head> de un documento HTML. Las etiquetas HTML generadas por estos procedimientos se deben colocar entre las etiquetas htp.headOpen y htp.headClose.
- Procedimientos de impresión que se utilizan con funciones HTF para generar una secuencia en el documento HTML que es construido.

- Procedimientos del cuerpo que se utilizan en la sección <body> del documento HTML. Se utiliza para dar formato a párrafos, agregar comentarios, links e imágenes al documento.
- Los procedimientos de lista que permiten mostrar la información enlistada:
- Los procedimientos de carácter que se utilizan para especificar o para alterar el aspecto del texto marcado. Las etiquetas del formato del carácter tienen elementos que abren y cierran y afectan solamente el texto que se encuentra entre ellas.
- Los procedimientos de tabla que insertan las tablas HTML en un documento.

Paquete OWA_INIT. Este paquete contiene los procedimientos y funciones que inicializan el cartridge. También proporciona constantes que se elimina para fijar el huso horario utilizado por los cookies.

Paquete OWA_COOKIE. Este paquete contiene tipos de datos, procedimientos, y funciones que permiten enviar cookies del HTTP y conseguirlos del browser del cliente. Los cookies del HTTP son secuencias opacas enviadas al browser para mantener el estado entre llamadas de HTTP. El estado puede ser mantenido a través de la sesión del cliente o más tiempo si una fecha de caducidad es incluida.

Paquete OWA_IMAGE. Este paquete contiene tipos de datos y funciones que se usan para conseguir las coordenadas de donde el usuario hizo clic en una imagen.

Paquete OWA_OPT_LOCK. Este paquete contiene funciones y procedimientos que permiten imponer a la base de datos estrategias de cierre optimistas, para prevenir actualizaciones perdidas.

Etiquetas HTML, HEAD y BODY

- htp.htmlOpen, htp.htmlClose - <HTML> and </HTML>
  Marcan el principio y el final de un documento de HTML.
- htp.headOpen, htp.headClose - <HEAD> and </HEAD>
  Marcan la sección de encabezado de un documento de HTML.
- htp.bodyOpen, htp.bodyClose - <BODY> and </BODY>
  Marcan la sección de cuerpo de un documento de HTML.

Etiqueta de Comentario

- htp.comment - <!-- and -->
  Genera etiquetas de comentario.

Etiquetas del área <HEAD>

- htp.base - <BASE>
  Registra el URL del documento.
- htp.linkRel - <LINK> with the REL attribute
- htp.linkRev - <LINK> with the REV attribute
  Estas etiquetas indican relación entre documentos, pero no crean un link.
- htp.title - <TITLE>
  Especifica el texto para mostrar en la barra de título de la ventana del navegador.
- htp.meta - <META>
  Introduce la meta-información sobre el documento y también especifica valores para los headers HTTP.
- htp.script - <SCRIPT>
  Contienen una escritura en lenguas como JavaScript y Vbscript.
- htp.style - <STYLE>
  Incluya una hoja de estilo en su Página Web.
- htp.isindex - <ISINDEX>
  Crea un campo de entrada con un texto digitado y lo envía al URL de la página o programa.

Etiquetas de Applet

- htp.appletopen, htp.appletclose - <APPLET> and </APPLET>
  Invoca un applet Java.
- htp.param - <PARAM>
  Especifica valores de parámetro para un applet Java.

Etiquetas de Lista

- htp.olistOpen, htp.olistClose - <OL> and </OL>
  Defina una lista ordenada. Una lista ordenada presenta una lista de artículos numerados.
- htp.ulistOpen, htp.ulistClose - <UL> and </UL>
  Presenta un lista deseordenada.
- htp.dlistOpen, htp.dlistClose - <DL> and </DL>
  htp.dlistTerm - <DT>
  htp.dlistDef - <DD>
  Cree una lista de definición. Una lista de definición parece a un glosario: esto contiene términos y definiciones. Los términos son insertados usando htp.dlistTerm y las definiciones son insertadas usando htp.dlistDef.
- htp.dirlistOpen, htp.dirlistClose - generate <DIR> and </DIR>
  Cree una sección de lista de directorio. Una lista de directorio presenta una lista de artículos que contiene hasta 20 carácteres.
- htp.listHeader - <LH>
  Imprime una etiqueta de HTML a principios de la lista.
- htp.listingOpen, htp.listingClose - <LISTING> and </LISTING>
  Marca una sección del texto de anchura fija en el cuerpo de una página de HTML.
- htp.menulistOpen, htp.menulistClose - <MENU> and </MENU>
  Crea una lista que presenta una línea por artículo. Los artículos en la lista parecen más compactos que una lista desordenada.
- htp.listItem - <LI>
  Indica una lista de elementos.

Etiquetas de Forma

- htp.formOpen, htp.formClose - <FORM> and </FORM>
  Crea una sección de forma en un documento de HTML.
- htp.formCheckbox - <INPUT TYPE="CHECKBOX">
  Iinserta un elemento checkbox en una forma.
- htp.formHidden - <INPUT TYPE="HIDDEN">
  Inserta un elemento de forma escondido. Este elemento no es visto por el usuario.
- htp.formImage - <INPUT TYPE="IMAGE">
  Crea un campo de imagen que el usuario pulsa para presentar la forma inmediatamente.
- htp.formPassword - <INPUT TYPE="PASSWORD">
  Crea un campo de entrada de texto de línea sola. Cuando el usuario entra en el texto en el campo, cada carácter es representado por un asterisco. Este es usado para entrar en contraseñas.
- htp.formRadio - <INPUT TYPE="RADIO">
  Crea un botón de radio sobre la forma HTML. Dentro de un juego de botones de radio, el usuario selecciona sólo un. Cada botón de radio en el mismo juego tiene el mismo nombre, pero valores diferentes.
- htp.formSelectOpen, htp.formSelectClose -<SELECT> and </SELECT>
  Crea un elemento de forma SELECT. Un elemento de este tipo es una lista donde el usuario selecciona uno o varios valores.
- htp.formSelectOption - <OPTION>
  Representa una opción en un elemento SELECT.
- htp.formText - <INPUT TYPE="TEXT">
  Crea un campo para una sola línea de texto.
- htp.formTextarea, htp.formTextarea2 - <TEXTAREA>
  Crea un campo de texto que no tiene ningún texto predefinido en el área de texto. Este campo permite la entrada en varias líneas del texto.

- htp.formReset - <INPUT TYPE="RESET">
  Crea un botón que, cuando seleccionado, reinicializa los campos de forma a sus valores iniciales.
- htp.formSubmit - <INPUT TYPE="SUBMIT">
  Crea un botón que, cuando pulsado, presenta la forma.

Etiquetas de Tablas

- htp.tableOpen, htp.tableClose - <TABLE> and </TABLE>
  Define una tabla HTML.
- htp.tableCaption - <CAPTION>
  Coloca un título en una tabla HTML.
- htp.tableRowOpen, htp.tableRowClose - <TR> and </TR>
  Inserta una nueva fila en una tabla HTML.
- htp.tableHeader - <TH>
  Inserta una celda de cabecera en una tabla HTML.
- htp.tableData - <TD>
  Insertan datos en una celda de una tabla HTML.
- htf.format_cell - <TD>
  Formatea valores de columna dentro de una tabla HTML que usa htf.tableData. Esto permite el mejor control de las tablas HTML.

Etiquetas de Linea, Imagen, Hipervinculo y Mapas

- htp.line, htp.hr - <HR>
  Genera una línea en el documento de HTML.
- htp.img, htp.img2 - <IMG>
  Dirige el navegador para cargar una imagen en la página de HTML.
- htp.anchor, htp.anchor2 - <A>
  Especifican la fuente o el destino de un link de hipertexto.
- htp.mapOpen, htp.mapClose - <MAP> and </MAP>
  Marca un juego de regiones en un mapa de imagen del lado cliente.

Etiquetas para formatear Párrafos

- htp.header - heading tags (<H1> to <H6>)

  Genera la apertura de encabezado sobre etiquetas (<H1> <a H6>) y su correspondiente que cierra etiquetas (</H1> <a/h6>).

- htp.para, htp.paragraph - <P>

  Indica que el texto que viene después de la etiqueta debe ser formateado como un párrafo. El htp.paragraph le permite añadir atributos a la etiqueta.

- htp.print, htp.prn - generate any text that is passed in

  El htp.prn genera el parámetro especificado como un string. A diferencia de htp.print, el string no se terminada con el caracter nueva linea.

- htp.prints, htp.ps - generate any text that is passed in; special characters in HTML are escaped

  Estos subprogramas generan una cadena y sustituyen los carácteres siguientes por la secuencia de escape correspondiente.

- htp.preOpen, htp.preClose - <PRE> and </PRE>

  Marcan una sección del texto preformateado en el cuerpo de la página de HTML.

- htp.blockquoteOpen, htp.blockquoteClose - <BLOCKQUOTE> and </BLOCKQUOTE>

  Marcan una sección del texto cotizado.

- htp.div - <DIV>

  Crea divisiones de documento.

- htp.nl, htp.br -

  Comienza una nueva línea del texto.

- htp.nobr - <NOBR>

  apagan la rotura de línea en una sección de texto.

- htp.wbr - <WBR>

  Inserta una ruptura de línea suave dentro de una sección del texto NOBR.

- htp.plaintext - <PLAINTEXT>

  Dirigen el navegador para dar el texto que ellos rodean en el tipo de anchura fija.

- htp.address - <ADDRESS>

  Especifican la dirección, el autor y la firma de un documento.
- htp.area - <AREA>

  La etiqueta define áreas dentro de la imagen y destinos para las áreas.

Etiquetas para formatear Caracteres

- htp.basefont - <BASEFONT>

  Especifica el tamaño de la fuente para una Página Web.
- htp.big - <BIG>

  Dirige el navegador para dar al texto una fuente más grande.
- htp.bold - <B>

  Muestra el texto en negrilla.
- htp.center - <CENTER> and </CENTER>

  Centra una sección de texto dentro de una página Web.
- htp.centerOpen, htp.centerClose - <CENTER> and </CENTER>

  Marca la sección de texto a centrarse.
- htp.cite - <CITE>

  Dirige el navegador para dar el texto como la cita.
- htp.code - <CODE>

  Dirigen el navegador para dar al texto un solo espacio.
- htp.dfn - <DFN>

  Dirigen el navegador para dar el texto en la cursiva.
- htp.fontOpen, htp.fontClose - <FONT> and </FONT>

  Marca una sección de texto con las características de fuente especificadas.
- htp.italic - <I>

  Dirige el navegador para dar el texto en la cursiva
- htp.keyboard, htp.kbd - <KBD> and </KBD>

  Dirigen el navegador para dar al texto un solo espacio.
- htp.s - <S>

  Dirigen el navegador para dar el texto en el que ellos rodean el tipo tachado.

- ü htp.sample - <SAMP>
  Dirigen el navegador para dar el texto que ellos rodean en la fuente de un solo espacio.
- ü htp.small - <SMALL>
  Dirigen el navegador para dar el texto ellos rodean la utilización de una pequeña fuente.
- ü htp.strike - <STRIKE>
  Dirigen el navegador para dar el texto en el que ellos rodean el tipo tachado.
- ü htp.strong - <STRONG>
  Dirigen el navegador para dar el texto que ellos rodean en negrilla.
- ü htp.sub - <SUB>
  Dirigen el navegador para dar el texto que ellos rodean como el subíndice.
- ü htp.sup - <SUP>
  Dirigen el navegador para dar el texto que ellos rodean como la superescritura.
- ü htp.teletype - <TT>
  Dirigen el navegador para dar el texto que ellos rodean en una fuente de máquina de escribir de anchura fija, por ejemplo, la fuente de mensajero.
- ü htp.underline - <U>
  Dirigen el navegador para dar el texto que ellos rodean de subrayar.
- ü htp.variable - <VAR>
  Dirigen el navegador para dar el texto que ellos rodean en la cursiva.

Etiquetas de Marco

- ü htp.frame - <FRAME>
  Define las características de un marco creado por la etiqueta <FRAMESET>.
- ü htp.framesetOpen, htp.framesetClose - <FRAMESET> and </FRAMESET>
  Definen una sección frameset.
- ü htp.noframesOpen, htp.noframesClose - <NOFRAMES> and </NOFRAMES>
  Marcan una sección sin marcos.