



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES

PLANTEL ARAGÓN

**“DESARROLLO DE UN SISTEMA PARA FACILITAR LA
ADMINISTRACIÓN DE LAS BASES DE DATOS DE
POSTGRESQL CON UNA INTERFAZ WEB”**

T E S I S

**PARA OBTENER EL TITULO DE:
INGENIERA EN COMPUTACIÓN**

**P R E S E N T A :
JAZMÍN CABELLO PÉREZ**

ASESOR: MAT. LUÍS RAMÍREZ FLORES



FES Aragón

México

2009



Universidad Nacional
Autónoma de México



UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.



UNIVERSIDAD NACIONAL
AVENIDA DE
MEXICO

Facultad de Estudios Superiores Aragón

DIRECCIÓN

JAZMIN CABELLO PEREZ
Presente

Con fundamento en el punto 6 y siguientes, del Reglamento para Exámenes Profesionales en esta Facultad, y toda vez que la documentación presentada por usted reúne los requisitos que establece el precitado Reglamento; me permito comunicarle que ha sido aprobado su tema de tesis y asesor.

TÍTULO: "DESARROLLO DE UN SISTEMA PARA FACILITAR LA ADMINISTRACIÓN DE LAS BASES DE DATOS DE POSTGRESQL CON UNA INTERFAZ WEB"

ASESOR: Mat. LUIS RAMIREZ FLORES

Aprovecho la ocasión para reiterarle mi distinguida consideración.

Atentamente
"POR MI RAZA HABLARÁ EL ESPÍRITU"
San Juan de Aragón, México, 1 de julio de 2008.

LA DIRECTORA


ARQ. LILIA TURCOTT GONZÁLEZ



C p Secretaria Académica
C p Jefatura de Carrera de Ingeniería en Computación
C p Asesor de Tesis

LTG/AIR/agm

SEGUIMIENTO DE REGISTRO DE TESIS

FECHA	CVE.	DESCRIPCIÓN DEL TRÁMITE	SELO
28/01/2009	IMP.	"DESARROLLO DE UN SISTEMA PARA FACILITAR LA ADMINISTRACIÓN DE LAS BASES DE DATOS DE POSTGRESQL CON UNA INTERFAZ WEB"	 Lic. ALBERTO IBARRA ROSAS SECRETARIO ACADÉMICO
			Lic. ALBERTO IBARRA ROSAS SECRETARIO ACADÉMICO
			Lic. ALBERTO IBARRA ROSAS SECRETARIO ACADÉMICO
			Lic. ALBERTO IBARRA ROSAS SECRETARIO ACADÉMICO
			Lic. ALBERTO IBARRA ROSAS SECRETARIO ACADÉMICO
			Lic. ALBERTO IBARRA ROSAS SECRETARIO ACADÉMICO

TRÁMITE	CLAVE
PRÓRROGA	PR.
CAM. TÍTULO	C. T.
CAM. ASESOR	C. A.
CAM. SEM.	C. S.
VIGENCIA	VIG.
IMPRESIÓN	IMP.

DEDICATORIA

Dedico esta obra a las personas que dejaron sus vidas para dedicarse a mí, sin importar que tan extenso fuera el sacrificio y que tan arduo el trabajo siempre dieron más del 100 por mi y por su familia, a ellos que día a día llenaron mi vida de alegría de felicidad, de mucho aprendizaje, pero sobre todo de mucho amor... **¡Mis Padres!**

A ti mamá, que siempre estuviste al pendiente de mi, dedicada en cuerpo y alma hasta el día de hoy, a ti que me has educado, aconsejado y que siempre estas presente en mis triunfos y fracasos, que has llorado por mis desilusiones y has reído por mis alegrías a pesar de tantos disgustos que te he causado, siempre has estado firme y siempre has creído y confiado en mí, a ti, te dedico esto, mostrándote que has hecho de mi una mujer trabajadora, preparada, con esa sed de luchar y de vivir de la mejor manera.

Gracias por ser mi mamá. Te Amo.

GRACIELA PÉREZ MARTÍNEZ

A ti papá, que de igual forma has estado conmigo y has dedicado toda un vida en nosotras, que te has esforzado por llevar el pan a la casa, que me has enseñado a valorar el trabajo, la familia y todo lo que se tiene en la vida. A ti que me has guiado y dado consejos a veces con regaños, a veces con cariño pero siempre dejaron algo bueno que me hizo mejorar como persona, a ti que me has enseñado con sacrificio y trabajo lo que vale la pena en esta vida y que pase lo que pase nunca rendirse, por que vendrán tiempos malos y buenos pero de ambos sacaremos lo mejor.

Gracias por ser mi papá. Te Amo.

VICTOR HUGO CABELLO MELCHOR

Esto es para los mejores maestros que he tenido a lo largo de mi vida por que se que lo más difícil de educar, es que el alumno sea tu hijo(a). Gracias por haberme dado un ambiente familiar lleno de amor hacia mi hermana y a mí pero sobre todo amor entre ustedes, que eso fue lo que me hizo ser mejor. Gracias mil gracias.



Agradecimientos

Agradezco principalmente a Dios por darme la oportunidad de realizar uno de tantos sueños y una de tantas metas en mi vida, por estar conmigo en cada paso que doy, por fortalecer mi corazón y por haber puesto en mi camino a todas aquellas personas que han sido mi apoyo y compañía durante todo este tiempo pero sobre todo por llenar mi vida de tantas bendiciones y de tanta dicha.

Agradezco mi **hermanita Ivonne**, por ser mi amiga y confidente, por que en cada momento siempre está escuchándome, jalándome las orejas o dándome un consejo, este proyecto va dedicado igualmente a ti bombón, por ser una excelente maestra en todos los aspectos, por llevarme de la mano desde muy pequeña, por tener ese cuidado sobre mí y por protegerme tanto, sabes que te quiero y que esto no lo hubiera logrado sin ti.

Gracias por estar a mi lado a pesar de tantos disgustos que te cause, eres la mejor hermana que dios pudo haberme dado. *Mil gracias bombón recuerda que Te Amo.*

Agradezco hoy y siempre a toda mi familia porque a pesar de no estar presentes físicamente en todo momento, se que procuran mi bienestar, a todos mis primos en especial a **Lety, Quique, César** que siempre me han apoyado y han sentido el malestar cuando algo no me sale y que siempre están dispuestos a pasar tiempo conmigo, a mi tía **Lety** por apoyarme y estar pendiente de mi salud y de lo que me pasa, y a pesar de lo que ha vivido siempre me regala una sonrisa, a tía **Irma** por apoyar a mi familia y estar a mi lado en cada momento siempre con ánimo de sobresalir y seguir adelante a pesar de lo que pase, así como a todo mis primos y tíos que se que me brindan su apoyo .

A mi gran amigo **Alberto**, por ser la persona que ha compartido su mayor tiempo conmigo, porque en su compañía las cosas malas se convertían en buenas, la tristeza se transformaba en alegría, a ti más que a nadie estoy agradecida por haber sido el pilar más fuerte en esta etapa, por darme tantos consejos, por aguantar tantas gritos por haber vivido mi coraje, mi angustia así como cada una de mis alegrías mil gracias “AMI”.

A mis amigos y compañeros del **diplomado de administración de bases de datos 9a generación en especial a Carlos Reyes y Daniel Chinchot**, los cuales me enseñaron mucho en tan poco tiempo y que semana a semana me brindaban una sonrisa y ponían un toque de alegría a ese lugar.

Agradezco también a mis amigos **Edgar y José Luis** a los cuales estimo y quiero ya que aunque no siempre están físicamente, siempre se dan un tiempo para mí, a ellos que me han enseñado y sacado de dudas y que de alguna manera han vivido conmigo este proceso y en muchas ocasiones soportaron mi mal humor o la falta de tiempo y de atención hacia ellos.
¡Gracias por ser mis amigos!

Por último quiero agradecer a la **Universidad Nacional Autónoma de México, a la FES Aragón** y cada uno de los profesores que me brindaron su apoyo y me brindaron su tiempo y sus conocimientos, en especial a los profesores: **Ing. Roberto Blanco, Mat. Luis Ramírez, Ing. Narciso Acevedo, MTI. Omar Mendoza, Ing. Rodolfo Morales**, que me ayudaron y apoyaron durante el proceso de titulación y que a pesar de la circunstancias ahí estaban. Así como agradecerle a cada uno de los profesores de Aragón de los cuales obtuve un gran aprendizaje.

En general quisiera agradecer a todas y cada una de las personas que han vivido conmigo la realización de esta tesis, con sus altos y bajos y que no necesito nombrar porque tanto ellas como yo sabemos que desde los más profundo de mi corazón les agradezco el haberme brindado todo el apoyo, colaboración, ánimo y sobre todo cariño y amistad.

CONTENIDO TEMÁTICO

OBJETIVOS	7
INTRODUCCIÓN	8
CAPÍTULO I. MARCO TEÓRICO	
I.1 Ciclo de vida de las aplicaciones de bases de datos.....	10
I.2 Métodos de diseño de sistemas.....	16
I.3 Base de datos.....	21
I.4 Modelo relacional.....	38
I.5 Lenguaje de consulta SQL (Structured Query Language).....	42
I.6 Programación de clientes.....	57
I.7 PHP.....	57
I.8 Administración de bases de datos.....	60
I.9 Seguridad en las bases de datos.....	65
CAPÍTULO II. ANÁLISIS Y DESARROLLO	
II.1 Funciones de la coordinación de desarrollo institucional.....	67
II.2 Planteamiento del problema.....	68
II.3 Requerimientos del sistema.....	68
II.4 Planteamiento de la solución.....	72

CAPÍTULO III. IMPLANTACIÓN Y PRUEBAS

III.1 Presentación al usuario.....	101
III.2 Administración de base de datos.....	104
III.3 Administración de usuarios.....	112
III.4 Administración de Grupos.....	115
CONCLUSIONES.....	121
BIBLIOGRAFÍA... ..	122
REFERENCIA ELECTRÓNICA... ..	123
ANEXOS.....	124

OBJETIVO GENERAL

Desarrollar e implementar un sistema de administración y control de las bases de datos de PostgreSQL, que sirva como herramienta para simplificar el proceso de las tareas administrativas cotidianas de las bases de datos.

OBJETIVOS ESPECÍFICOS

- Simplificar el proceso de administración de las bases de datos situadas en el manejador de base de datos de PostgreSQL, así como la ejecución de sentencias SQL mediante una interfaz web.
- Facilitar el proceso de administración de los objetos de la base de datos como son: tablas, vistas, etc.
- Crear lineamientos para la creación de perfiles de acceso y mediante la interfaz facilitar el proceso de administración de usuarios, grupos y privilegios.
- Crear la documentación necesaria para la actualización del sistema y que ésta pueda ser trasladada para su procesamiento a un sistema futuro.
- Facilitar la administración de los principales catálogos de la empresa.

INTRODUCCIÓN

Las grandes aplicaciones requieren de un apoyo cuidadoso. La mayoría de las organizaciones contrata a un administrador de bases de datos para que esté monitoreando el desempeño de la aplicación, evalúe las condiciones de seguridad y se asegure de la integridad de las bases de datos.

Los gerentes de sistemas de información se preocupan de manera creciente por aquellas cuestiones que implican el otorgar acceso a los datos y proporcionar respuestas a lo largo de la organización y TecnoGens no es la excepción ya que sus actividades requieren del manejo de grandes cantidades de información almacenada en bases de datos de PostgreSQL y éste proyecto pretende profundizar en los aspectos técnicos los cuales involucran la creación y ejecución de las bases de datos, así como las tareas de desempeño, respaldo y recuperación, además de la asignación y control de la integridad. Los administradores de las bases de datos están capacitados en los detalles de instalación, configuración y operación del Sistema Manejador de Base de Datos (DBMS).

La seguridad de las bases de datos es un grupo secundario dentro de los temas seguridad para la computadora. Sin embargo debido al objetivo de compartir datos, la seguridad es un problema crucial en la administración de una base de datos. Si la seguridad se asigna de manera apropiada se tiene la capacidad de reducir muchos tipos de fraudes. Si ésta se ignora los activos principales de la compañía podrían manipularse o robarse desde cualquier computadora del mundo.

De esta manera se hace evidente la necesidad de diseñar e implementar un soporte para apoyar a la administración de las bases de datos con restricciones y funcionalidades que permitan responder a las necesidades de los usuarios como de los administradores y además realizar operaciones de consulta SQL en una interfaz más agradable para los usuarios.

Dentro de los resultados esperados del sistema se encuentran:

- Simplificar el proceso de administración de las bases de datos situadas en el manejador de base de datos de PostgreSQL, así como la ejecución de sentencias SQL mediante una interfaz web.
- Crear lineamientos para la creación de cuentas de usuarios y monitoreo de la seguridad
- Administrar el respaldo y recuperación de las bases de datos
- Ofrecer a los usuarios información y manipulación sobre los objetos de las bases de datos.
- Exportar datos a diferentes formatos
- Realizar la administración de los catálogos básicos.

En la actualidad existen varias herramientas que proporcionan soporte para algunos de los requerimientos antes planteados, como es el caso de *PHPPGADMIN*, *PHPMYADMIN*, entre otros, sin embargo la manera en que se muestra la información no es la adecuada para la empresa, ellos requieren que el sistema muestre los datos de salida de una forma más completa y con el mejor diseño utilizando integrando su logotipo y características propias de las políticas de la empresa, ya que ésta pretende mostrar que cualquier sistema ofrecido en el mercado ya sea de pago o software libre, siempre se puede crear y complementar, además sugiere contar con una plataforma de diseño colorida y vistosa para que sea más agradable para cada uno de los usuarios que manipulen el sistema, además de ser la característica principal del sistema radica en la administración de los diferentes catálogos de la empresa.

Por otro lado los usuarios que cuentan con poco conocimiento en base de datos requieren de información resumida y un ambiente que facilite el aprendizaje de lo realiza el sistema por lo que para este tipos de usuarios es de gran importancia la interfaz ya que su información en cuanto a las bases de datos será presentada de diferente forma.

En el primer capítulo se hace énfasis en la conveniencia del uso de la computadora como una herramienta de administración y los beneficios que esto implica así como todos los temas involucrados en el manejo de bases de datos y en la administración de las mismas.

El segundo capítulo está dedicado a la explicación del análisis y diseño del sistema partiendo de la estructura de los archivos el software utilizado así como los diagramas para modelar el sistema y la explicación de los mismos.

El tercer capítulo explica el proceso de implantación desde la presentación al usuario y la puesta en servicio del sistema.

Finalmente se encuentran las conclusiones, bibliografía, referencia electrónica y los anexos.

CAPÍTULO I. MARCO TEÓRICO

La tecnología de la Información (TI) está cambiando la forma tradicional de hacer las cosas, las personas que trabajan en gobierno, en empresas privadas, que dirigen personal o que trabajan como profesional en cualquier campo utilizan la TI cotidianamente mediante el uso de Internet, las tarjetas de crédito, el pago electrónico de la nómina, entre otras funciones; es por eso que la función de la TI en los procesos de la empresa como manufactura y ventas se han expandido grandemente.

Utilizando eficientemente la tecnología de la información se pueden obtener ventajas competitivas, pero es preciso encontrar procedimientos acertados para mantener tales ventajas como una constante, así como disponer de cursos y recursos alternativos de acción para adaptarlas a las necesidades del momento, pues las ventajas no siempre son permanentes.

El sistema de información tiene que modificarse y actualizarse con regularidad si se desea percibir ventajas competitivas continuas. El uso creativo de la tecnología de la información puede proporcionar a los administradores una nueva herramienta para diferenciar sus recursos humanos, productos y/o servicios respecto de sus competidores. Este tipo de preeminencia competitiva puede traer consigo otro grupo de estrategias, como es el caso de un sistema flexible y las normas justo a tiempo, que permiten producir una variedad más amplia de productos a un precio más bajo y en menor tiempo que la competencia.

En la implementación de un sistema de información intervienen muchos factores siendo uno de los principales el factor humano. Es previsible que ante una situación de cambio el personal se muestre renuente a adoptar los nuevos procedimientos o que los desarrolle plenamente y de acuerdo a los lineamientos que se establecieron.

I.1 CICLO DE VIDA DE LAS APLICACIONES DE BASES DE DATOS

Las etapas del ciclo de vida de una aplicación de bases de datos son las siguientes:

1. Planificación del proyecto.
2. Definición del sistema.
3. Recolección y análisis de los requisitos.
4. Diseño de la base de datos.

5. Selección del SGBD.
6. Diseño de la aplicación.
7. Prototipado.
8. Implementación.
9. Conversión y carga de datos.
10. Prueba.
11. Mantenimiento.

Estas etapas no son estrictamente secuenciales. De hecho hay que repetir algunas de las etapas varias veces, haciendo lo que se conocen como ciclos de realimentación. Por ejemplo, los problemas que se encuentran en la etapa del diseño de la base de datos pueden requerir una recolección de requisitos adicional y su posterior análisis.

A continuación, se muestran las tareas más importantes que se realizan en cada etapa.

Planificación del proyecto

Esta etapa conlleva la planificación de cómo se pueden llevar a cabo las etapas del ciclo de vida de la manera más eficiente. Hay tres componentes principales: el trabajo que se ha de realizar, los recursos para llevarlo a cabo y el dinero para pagar por todo ello. Como apoyo a esta etapa, se necesitará un modelo de datos corporativo en donde se muestren las entidades principales de la empresa y sus relaciones, y en donde se identifiquen las principales áreas funcionales. Normalmente, este modelo de datos se representa mediante un diagrama entidad-relación. En este modelo se tiene que mostrar también qué datos comparten las distintas áreas funcionales de la empresa.

La planificación de la base de datos también incluye el desarrollo de estándares que especifiquen cómo realizar la recolección de datos, cómo especificar su formato, qué documentación será necesaria y cómo se va a llevar a cabo el diseño y la implementación.

Definición del sistema

En esta etapa se especifica el ámbito y los límites de la aplicación de bases de datos, así como con qué otros sistemas interactúan. También hay que determinar quiénes son los usuarios y las áreas de aplicación.

Recolección y análisis de los requisitos

En esta etapa se recogen y analizan los requerimientos de los usuarios y de las áreas de aplicación.

Esta información se puede recoger de varias formas:

- Entrevistando al personal de la empresa, concretamente, a aquellos que son considerados expertos en las áreas de interés.
- Observando el funcionamiento de la empresa.
- Examinando documentos, sobre todo aquellos que se utilizan para recoger o visualizar información.
- Utilizando cuestionarios para recoger información de grandes grupos de usuarios.
- Utilizando la experiencia adquirida en el diseño de sistemas similares.

La información recogida debe incluir las principales áreas de aplicación y los grupos de usuarios, la documentación utilizada o generada por estas áreas de aplicación o grupos de usuarios, las transacciones requeridas por cada área de aplicación o grupo de usuarios y una lista priorizada de los requerimientos de cada área de aplicación o grupo de usuarios.

Esta etapa tiene como resultado un conjunto de documentos con las especificaciones de requisitos de los usuarios, en donde se describen las operaciones que se realizan en la empresa desde distintos puntos de vista.

Diseño de la base de datos

Esta etapa consta de tres fases: diseño conceptual, diseño lógico y diseño físico de la base de datos. La primera fase consiste en la producción de un esquema conceptual, que es independiente de todas las consideraciones físicas. Este modelo se refina después en un esquema lógico eliminando las construcciones que no se pueden representar en el modelo de base de datos escogido (relacional, orientado a objetos, etc.). En la tercera fase, el esquema lógico se traduce en un esquema físico para el SGBD escogido. La fase de diseño físico considera las estructuras de almacenamiento y los métodos de acceso necesarios para proporcionar un acceso eficiente a la base de datos en memoria secundaria.

Los objetivos del diseño de la base de datos son:

- Representar los datos que requieren las principales áreas de aplicación y los grupos de usuarios, y representar las relaciones entre dichos datos.
- Proporcionar un modelo de datos que soporte las transacciones que se vayan a realizar sobre los datos.
- Especificar un esquema que alcance las prestaciones requeridas para el sistema.

Hay varias estrategias a seguir para realizar el diseño: de abajo a arriba, de arriba a abajo, de dentro a fuera y la estrategia mixta. La estrategia de abajo a arriba parte de todos los atributos y los va agrupando en entidades y relaciones. Es apropiada cuando la base de datos es simple, con pocos atributos. La estrategia de arriba a abajo es más apropiada cuando se trata de bases de datos complejas. Se comienza con un esquema con entidades de alto nivel, que se van refinando para obtener entidades de bajo nivel, atributos y relaciones.

La estrategia de dentro a fuera es similar a la estrategia de abajo a arriba, pero difiere en que se parte de los conceptos principales y se va extendiendo el esquema para considerar también otros conceptos, asociados con los que se han identificado en primer lugar. La estrategia mixta utiliza ambas estrategias, de abajo a arriba y de arriba a abajo, con un esquema de divide y vencerás. Se obtiene un esquema inicial de alto nivel, se divide en partes, y de cada parte se obtiene un subesquema. Estos subesquemas se integran después para obtener el modelo final.

Selección del SGBD

Si no se dispone de un SGBD, o el que hay se encuentra obsoleto, se debe escoger un SGBD que sea adecuado para el sistema de información. Esta elección se debe hacer en cualquier momento antes del diseño lógico.

Diseño de la aplicación

En esta etapa se diseñan los programas de aplicación que usarán y procesarán la base de datos. Esta etapa y el diseño de la base de datos, son paralelas. En la mayor parte de los casos no se puede finalizar el diseño de las aplicaciones hasta que se ha terminado con el diseño de la base de datos. Por otro lado, la base de datos existe para dar soporte a las aplicaciones, por lo que habrá una realimentación desde el diseño de las aplicaciones al diseño de la base de datos.

En esta etapa hay que asegurarse de que toda la funcionalidad especificada en los requisitos de usuario se encuentra en el diseño de la aplicación. Habrá algunos programas que utilicen y procesen los datos de la base de datos.

Además, habrá que diseñar las interfaces de usuario, aspecto muy importante que se suele ignorar. El sistema debe ser fácil de aprender, fácil de usar, ser directo y estar "dispuesto a perdonar". Si la interface no tiene estas características, el sistema dará problemas, sin lugar a dudas.

Prototipado

Esta etapa, que es opcional, es para construir prototipos de la aplicación que permitan a los diseñadores y a los usuarios probar el sistema. Un prototipo es un modelo de trabajo de las aplicaciones del sistema. El prototipo no tiene toda la funcionalidad del sistema final, pero es suficiente para que los usuarios puedan utilizar el sistema e identificar qué aspectos están bien y cuáles no son adecuados, además de poder sugerir mejoras o la inclusión de nuevos elementos. Este proceso permite que quienes diseñan e implementan el sistema sepan si han interpretado correctamente los requisitos de los usuarios. Otra ventaja de los prototipos es que se construyen rápidamente.

Esta etapa es imprescindible cuando el sistema que se va a implementar tiene un gran coste, alto riesgo o utiliza nuevas tecnologías.

Implementación

En esta etapa se crean las definiciones de la base de datos a nivel conceptual, externo e interno, así como los programas de aplicación. La implementación de la base de datos se realiza mediante las sentencias del lenguaje de definición de datos (LDD) del SGBD escogido. Estas sentencias se encargan de crear el esquema de la base de datos, los ficheros en donde se almacenarán los datos y las vistas de los usuarios.

Los programas de aplicación se implementan utilizando lenguajes de tercera o cuarta generación. Partes de estas aplicaciones son transacciones sobre la base de datos, que se implementan mediante el lenguaje de manejo de datos (LMD) del SGBD. Las sentencias de este lenguaje se pueden embeber en un lenguaje de programación anfitrión como Visual Basic, Delphi, C, C++, Java, Fortran, etc. En esta etapa, también se implementan los menús, los formularios para la introducción

de datos y los informes de visualización de datos. Para ello, el SGBD puede disponer de lenguajes de cuarta generación que permiten el desarrollo rápido de aplicaciones mediante lenguajes de consultas no procedurales, generadores de informes, generadores de formularios, generadores de gráficos y generadores de aplicaciones.

También se implementan en esta etapa todos los controles de seguridad e integridad. Algunos de estos controles se pueden implementar mediante el LDD y otros puede que haya que implementarlos mediante utilidades del SGBD o mediante programas de aplicación.

Conversión y carga de datos

Esta etapa es necesaria cuando se está reemplazando un sistema antiguo por uno nuevo. Los datos se cargan desde el sistema viejo al nuevo directamente o, si es necesario, se convierten al formato que requiera el nuevo SGBD y luego se cargan. Si es posible, los programas de aplicación del sistema antiguo también se convierten para que se puedan utilizar en el sistema nuevo.

Pruebas

En esta etapa se prueba y valida el sistema con los requisitos especificados por los usuarios. Para ello, se debe diseñar una batería de tests con datos reales, que se deben llevar a cabo de manera metódica y rigurosa. Es importante darse cuenta de que la fase de prueba no sirve para demostrar que no hay fallos, sirve para encontrarlos.

Mantenimiento

Una vez que el sistema está completamente implementado y probado, se pone en marcha. El sistema está ahora en la fase de mantenimiento en la que se llevan a cabo las siguientes tareas:

- Monitorización de las prestaciones del sistema. Si las prestaciones caen por debajo de un determinado nivel, puede ser necesario reorganizar la base de datos.
- Mantenimiento y actualización del sistema. Cuando sea necesario, los nuevos requisitos que vayan surgiendo se incorporarán al sistema, siguiendo de nuevo las etapas del ciclo de vida que se acaban de presentar.

I.2 MÉTODOS DE DISEÑO DE SISTEMAS

Estrategias basadas en modelos

El diseño estructurado, la ingeniería de la información y el diseño orientado a objetos son ejemplos de estrategias basadas en modelos. El diseño basado en modelos pone énfasis en el trazado de modelos de sistemas pictóricos para documentar los aspectos técnicos o de implantación de un nuevo sistema.

Los modelos de diseño suelen derivarse de modelos lógicos. Hoy las estrategias basadas en modelo casi siempre se mejoran con el uso de herramientas automatizadas. Algunos diseñadores trazan modelos del sistema con software de gráficos de propósito general como Visio profesional. Otros diseñadores y organizaciones requieren de herramientas CASE basado en repositorios o herramientas de modelado como son Microsoft Visio, Rational de IBM, System Architect, etc. Las herramientas CASE brindan consistencia e integridad además de verificación de errores basada en reglas.

Diseño estructurado moderno. Las técnicas de diseño estructurado ayudan a los desarrolladores a manejar el tamaño y la complejidad de los programas. El diseño estructurado moderno es una técnica orientada a procesos para dividir un programa grande en una jerarquía de módulos lo que da por resultado un programa de computadora más fácil de implantar y mantener.

El diseño estructurado se considera como técnica orientada a procesos por que hace énfasis a los componentes de proceso del sistema de manera específica. Los módulos deben tener mucha cohesión, es decir cada modulo debe encargarse de una y solo una función. Esto hace que los módulos sean reutilizables en programas futuros.

Los módulos deben acoplarse lo cual significa que su dependencia mutua debe ser mínima. Ello minimiza el efecto que los cambios futuros en un modulo tendrán en otros módulos

Ingeniería de la información. Se centra en la planeación de datos y en la estrategia para generar

Elaboración de prototipos es un proceso iterativo que implica una relación de trabajo estrecha entre los diseñadores y los usuarios para producir el modelo de un nuevo sistema.

Desarrollo rápido de aplicaciones (RAD) construye una fusión de las diversas técnicas estructuradas con la elaboración de prototipos para acelerar el desarrollo de los sistemas.

Diseño orientado a objetos. Nueva estrategia de diseño que da seguimiento al análisis orientado a objetos para refinar las definiciones de requerimientos de objetos y definir nuevos objetos específicos del diseño en cuestión.

Lenguaje de Modelado Unificado (UML)

El lenguaje Unificado de modelado, es un lenguaje estándar para escribir planos de software, puede utilizarse para visualizar, especificar, construir y documentar los artefactos de un sistema que involucre una gran cantidad de software.

UML es apropiado para modelar desde sistemas de información empresariales hasta aplicaciones distribuidas basadas en Web e incluso para sistemas embebidos en tiempo real muy exigentes.

La visión general de UML se basa en:

- Visualizar
- Especificar
- Construir
- Documentar

UML incluye trece tipos de diagramas, pero los más utilizados son los siguientes:

- Clases
- Objetos
- Componentes
- Casos de uso
- Secuencia
- Estados
- Actividades
- Despliegue
- Paquetes

Visión Arquitectónica

Dos cualidades comunes para virtualmente todos los proyectos Orientado a Objetos son:

- La existencia de una visión arquitectónica fuerte
- La aplicación de un ciclo de vida incremental bien manejado e interactivo
- La arquitectura debe ser simple
- El logro de comportamiento común a través de abstracciones y mecanismos comunes

Para describir completamente una arquitectura, se necesitan cuatro vistas (fig. 1):

- La vista lógica para proporcionar una imagen estática de las clases primarias y sus relaciones
- La vista de componente para mostrar cómo está el código organizado en paquetes y librerías
- La vista de proceso para mostrar los procesos y tareas
- La vista de distribución para mostrar los procesadores, dispositivos y ligas en el ambiente operacional
- Finalmente, una vista de escenario explica cómo trabajan juntas las otras cuatro vistas

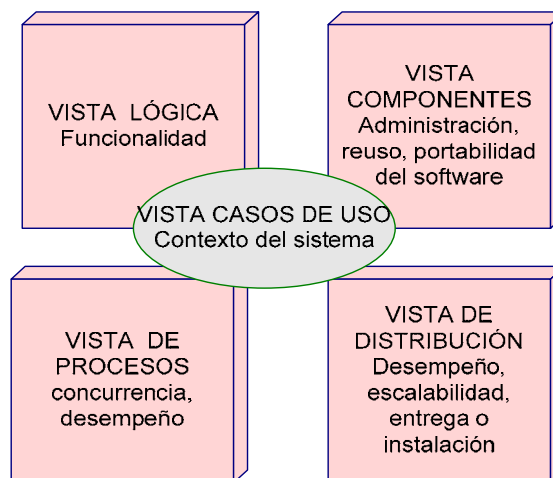


Figura 1. Arquitectura de UML

El Modelo de Caso de Uso

El modelo de casos de uso describe la funcionalidad propuesta del nuevo sistema. Un caso de uso representa una unidad discreta de interacción entre un usuario (humano o máquina) y el sistema. Un

Caso de Uso es una unidad simple de trabajo significativo; por ejemplo, "Validarse en el sistema", "Registrarse en el sistema" y "Crear un pedido" son todos casos de uso (fig. 2).

Cada caso de uso tiene una descripción que describe la funcionalidad que se construirá en el sistema propuesto. Un caso de uso puede "incluir" la funcionalidad de otro caso de uso o "extender" a otro caso de uso con su propio comportamiento.

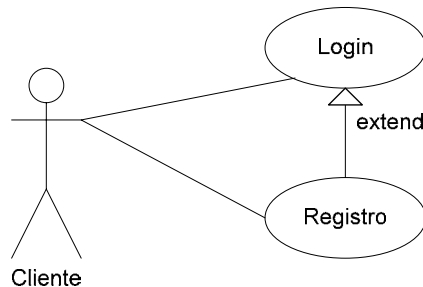


Figura 2. Ejemplo de un modelo de casos de uso

Una descripción de caso de uso generalmente incluirá:

- Comentarios generales y notas describiendo el caso de uso
- Requisitos -cosas que el caso de uso debe permitir hacer al usuario, tales como <capacidad para actualizar pedido>, <capacidad para modificar pedido>, etc.
- Restricciones -reglas acerca de qué se puede y qué no se puede hacer-. Incluye:
 - Pre-condiciones que deben ser verdaderas antes de que el caso de uso se ejecute, por ejemplo <crear pedido> debe preceder a <modificar pedido>
 - Post-condiciones que deben ser verdaderas una vez que el caso de uso se ejecutó, por ejemplo <el pedido está modificado y es consistente>
 - invariantes: éstas son siempre verdaderas -por ejemplo, un pedido debe tener siempre un número de cliente.
- Escenarios -descripciones secuenciales de los pasos que se toman para llevar a cabo el caso de uso. Pueden incluir escenarios múltiples, para satisfacer circunstancias excepcionales y caminos de proceso alternativos
- Diagramas de escenarios -diagramas de secuencia para describir el flujo de trabajo- similar al punto 4 pero descrito gráficamente.
- Atributos adicionales como fase de implementación, número de versión, rango de complejidad, estereotipo y estado

Actores

Un actor es un usuario del sistema. Incluye usuarios humanos y otros sistemas computarizados. Un actor usa un caso de uso para desempeñar alguna porción de trabajo que es de valor para el negocio. El conjunto de casos de uso al que un actor tiene acceso define su rol global en el sistema y el alcance de su acción (fig.3).

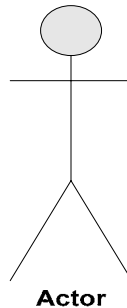


Figura 3. Representación de un actor en casos de uso

Diagramas de secuencia

Los diagramas de secuencia se usan para mostrar la interacción entre los usuarios, las pantallas y las instancias de los objetos en el sistema. Proveen un mapa secuencial del paso de los mensajes entre los objetos a lo largo del tiempo. Frecuentemente, estos diagramas se ubican bajo los casos de uso o los componentes en el modelo para ilustrar un escenario -cómo interactúa un usuario con el sistema y qué sucede internamente para que el trabajo se lleve a cabo-. Muchas veces, los objetos se representan utilizando íconos especialmente estereotipados, como en el ejemplo de abajo. El objeto etiquetado "PantallaDeIngreso" (Login Screen) se muestra empleando el ícono de interfaz. El objeto etiquetado "AdministradorDeSeguridad" (SecurityManager) se muestra usando el ícono controlador. El objeto etiquetado "Usuarios" (Users) se muestra usando el ícono entidad.

Diagramas de actividad

Los diagramas de actividad se usan para mostrar cómo se construyen los diferentes flujos de trabajo o los procesos dentro de un sistema, cómo se inician, los variados caminos alternativos que se pueden tomar desde el inicio hasta el fin. También pueden ilustrar dónde puede ocurrir procesamiento en paralelo durante la ejecución de algunas actividades.

El diagrama de componentes

El diagrama de componentes muestra la relación entre componentes de software, sus dependencias, su comunicación su ubicación y otras condiciones.

Interfaces

Los componentes también pueden exponer las interfaces. Estas son los puntos visibles de entrada o los servicios que un componente está ofreciendo y dejando disponibles a otros componentes de software y clases. Típicamente, un componente está compuesto por numerosas clases y paquetes de clases internos. También se puede crear a partir de una colección de componentes más pequeños.

Diagrama de clases.

Muestra un conjunto de clases, interfaces y colaboraciones así como relaciones, estos diagramas son más comunes para el modelo de sistemas orientado a objetos.

Diagrama de estado.

Muestra una máquina de estados, que consta de estados, transiciones, eventos y actividades. Son importantes para el modelado del comportamiento de una interfaz una clase o una colaboración

I.3 BASE DE DATOS

Definición de una base de datos

Una base de datos es una colección de datos integrados, con redundancia controlada y con una estructura que refleje las interrelaciones y restricciones existentes en el mundo real; los datos que han de ser compartidos por diferentes usuarios y aplicaciones, deben mantenerse independientes de éstas, y su definición y descripción, únicas para cada tipo de datos, han de estar almacenadas junto con los mismos. Los procedimientos de actualización y recuperación, comunes, y bien determinados, habrán de ser capaces de conservar la integridad, seguridad y confidencialidad del conjunto de datos.

Arquitectura para las bases de datos

Hace unos años atrás, las bases de datos eran el resultado de una compleja programación y de complicados mecanismos de almacenamiento. Con la popularización de la informática, la aparición de aplicaciones específicas también trajo con ella la disponibilidad de herramientas de gestión de datos que dieron lugar a los denominados Sistemas de Gestión de Bases de Datos, identificados por sus siglas SGBD por su acrónimo en inglés DBMS (Data Base Management Systems).

Las bases de datos respetan la arquitectura de tres niveles definida, para cualquier tipo de base de datos, por el grupo National Standard Institute - Standards Planning and Requirements Committee (ANSI/SPARC). En esta arquitectura la base de datos se divide en los niveles externo, conceptual e interno.

Hay tres características importantes inherentes a los sistemas de bases de datos: la separación entre los programas de aplicación y los datos, el manejo de múltiples vistas por parte de los usuarios y el uso de un catálogo para almacenar el esquema de la base de datos.

El objetivo de la arquitectura de tres niveles es el de separar los programas de aplicación de la base de datos física. En esta arquitectura, el esquema de una base de datos se define en tres niveles de abstracción distintos (fig. 4):

1. En el *nivel interno* se describe la estructura física de la base de datos mediante un *esquema interno*. Este esquema se especifica mediante un modelo físico y describe todos los detalles para el almacenamiento de la base de datos, así como los métodos de acceso.
2. En el *nivel conceptual* se describe la estructura de toda la base de datos para una comunidad de usuarios (todos los de una empresa u organización), mediante un *esquema conceptual*. Este esquema oculta los detalles de las estructuras de almacenamiento y se concentra en describir entidades, atributos, relaciones, operaciones de los usuarios y restricciones. En este nivel se puede utilizar un modelo conceptual o un modelo lógico para especificar el esquema.
3. En el *nivel externo* se describen varios *esquemas externos* o *vistas de usuario*. Cada esquema externo describe la parte de la base de datos que interesa a un grupo de usuarios determinado y oculto a ese grupo el resto de la base de datos. En este nivel se puede utilizar un modelo conceptual o un modelo lógico para especificar los esquemas.



Figura 4. Arquitectura de tres niveles

La mayoría de los SGDB no distinguen del todo los tres niveles. Algunos incluyen detalles del nivel físico en el esquema conceptual. En casi todos los SGBD que se manejan vistas de usuario, los esquemas externos se especifican con el mismo modelo de datos que describe la información a nivel conceptual, aunque en algunos se pueden utilizar diferentes modelos de datos en los niveles conceptuales y externo.

Características de las bases de datos

El término de bases de datos fue escuchado por primera vez en 1963, en un simposio celebrado en California, USA. Una base de datos se puede definir como un conjunto de información relacionada que se encuentra agrupada ó estructurada.

Cada base de datos se compone de una o más tablas que guarda un conjunto de datos. Cada tabla tiene una o más columnas y filas. Las columnas guardan una parte de la información sobre cada elemento que queremos guardar en la tabla, cada fila de la tabla conforma un registro.

Entre las principales características de los sistemas de base de datos podemos mencionar:

- Independencia lógica y física de los datos.
- Redundancia mínima.
- Acceso concurrente por parte de múltiples usuarios.
- Integridad de los datos.
- Consultas complejas optimizadas.
- Seguridad de acceso y auditoria.
- Respaldo y recuperación.
- Acceso a través de lenguajes de programación estándar.

Sistema de gestión de base de datos (SGBD)

Los Sistemas de Gestión de Base de Datos (en inglés DataBase Management System) son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Se compone de un lenguaje de definición de datos (DDL), de un lenguaje de manipulación de datos (DML) y de un lenguaje de consulta (SQL).

Ventajas de las bases de datos

El control sobre la redundancia de datos:

La redundancia de datos se refiere, a la existencia de información repetida o duplicada en diferentes tablas dentro de una base de datos.

La redundancia conduce a muchos problemas que tienen que ver con la integridad y consistencia de los datos. La redundancia de los datos requiere múltiples procedimientos de entrada y actualización de los mismos.

Dentro de una base de datos relacional la redundancia debe ser mínima y controlada. En ocasiones existirán motivos válidos de negocios o técnicos para mantener varias copias de los mismos datos almacenados.

Los sistemas almacenan varias copias de los mismos datos en archivos distintos. Esto hace que se desperdicie espacio de almacenamiento, además de provocar la falta de consistencia de datos.

En los sistemas de bases de datos todos estos archivos están integrados, por lo que no se almacenan varias copias de los mismos datos. Sin embargo, en una base de datos no se puede eliminar la redundancia completamente, ya que en ocasiones es necesaria para modelar las relaciones entre los datos.

Consistencia de datos:

Es muy probable que surjan incongruencias al almacenar la misma información en más de un lugar; ya que al modificar, eliminar o agregar un dato, en esas condiciones, debe realizarse en cada una de las instancias del mismo con el riesgo de no realizarlo en su totalidad, generando en este caso datos inconsistentes.

Eliminando o controlando las redundancias de datos se reduce en gran medida el riesgo de que haya inconsistencias. Si un dato está almacenado una sola vez, cualquier actualización se debe realizar sólo una vez, y está disponible para todos los usuarios inmediatamente. Si un dato está duplicado y el sistema conoce esta redundancia, el propio sistema puede encargarse de garantizar que todas las copias se mantienen consistentes.

Compartición de datos:

En los sistemas de archivos, los archivos pertenecen a las personas o a los departamentos que los utilizan. Pero en los sistemas de bases de datos, la base de datos pertenece a la empresa y puede ser compartida por todos los usuarios que estén autorizados.

Mantenimiento de estándares:

Gracias a la integración es más fácil respetar los estándares necesarios, tanto los establecidos a nivel de la empresa como los nacionales e internacionales. Estos estándares pueden establecerse sobre el formato de los datos para facilitar su intercambio, pueden ser estándares de documentación, procedimientos de actualización y también reglas de acceso.

Mejora en la integridad de datos:

La integridad de una base de datos se refiere no solo a que los datos sean consistentes dentro de la base, sino además, que los valores que posean los datos sean válidos de acuerdo a las dependencias funcionales entre tablas y de acuerdo a las políticas de negocio.

La inconsistencia entre dos entradas que representan al mismo “hecho” es un ejemplo de falta de integridad que, por supuesto, sólo ocurre si existe redundancia en los datos almacenados.

Mejora en la seguridad:

Hoy en día se considera a la información de una empresa como uno de los activos más valiosos, por lo que la seguridad de la misma es muy importante.

La seguridad implica asegurar que los usuarios están autorizados para llevar a cabo lo que tratan de hacer.

La seguridad de una base de datos se refiere principalmente al control de acceso, modificación y definición, tanto de los datos como de la estructura de la base de datos por parte de los diferentes usuarios a la misma.

Algunos sistemas operativos proporcionan algún nivel de seguridad en el control de acceso a usuarios, sin embargo ésta debe radicar principalmente en el SGBD o en la aplicación que maneje la base de datos, sobre todo para evitar la dependencia de entidades externas.

Por otro lado, una base de datos debe cumplir con las siguientes condiciones:

Tanto los usuarios finales como los programas de aplicación no necesitan conocer los detalles de las estructuras de almacenamiento ya que lo importante para estos usuarios es la información contenida.

Los datos son compartidos por diferentes usuarios y programas de aplicación; existe un mecanismo común para inserción, actualización, borrado y consulta de los datos.

Los procedimientos de actualización y recuperación, comunes, y bien determinados, habrán de ser capaces de conservar la integridad, seguridad y confidencialidad del conjunto de datos.

Tanto datos como procedimientos pueden ser transportables conceptualmente a través de diferentes SGBD.

Mejora en la productividad:

El SGBD proporciona muchas de las funciones estándar que el programador necesita escribir en un sistema. A nivel básico, el SGBD proporciona todas las rutinas de manejo de archivos típicas de los programas de aplicación.

El hecho de disponer de estas funciones permite al programador centrarse mejor en la función específica requerida por los usuarios, sin tener que preocuparse de los detalles de implementación de bajo nivel.

Integridad: Los datos en la base de datos deben de ser válidos de acuerdo a las políticas del negocio y se puede lograr mediante:

- El mantenimiento de una redundancia mínima y controlada.
- El establecimiento de llaves primarias o índices primarios.

- La validación de las dependencias entre las tablas.
- La creación de reglas de validación durante la edición de datos.

Seguridad. Se encarga del controlar el acceso, modificación y definición, tanto de los datos como de la estructura de la base de datos por parte de los diferentes usuarios a la misma.

Clasificación de los sistemas de gestión de bases de datos

El criterio principal que se utiliza para clasificar los SGBD es el modelo lógico en que se basan. Los modelos lógicos empleados con mayor frecuencia en los SGBD comerciales actuales son el relacional, el de red y el jerárquico. Algunos SGBD más modernos se basan en modelos orientados a objetos.

Modelo de datos

Es una colección de herramientas conceptuales para describir los datos, las relaciones que existen entre ellos, semántica asociada a los datos y restricciones de consistencia.

Los modelos de datos se dividen en tres grupos:

- Modelos lógicos basados en objetos.
- Modelos lógicos basados en registros.
- Modelos físicos de datos.

Modelos lógicos basados en objetos

Se usan para describir datos en los niveles conceptual y de visión, es decir, con este modelo representamos los datos de tal forma como nosotros los captamos en el mundo real, tienen una capacidad de estructuración bastante flexible y permiten especificar restricciones de datos explícitamente. Existen diferentes modelos de este tipo, pero el más utilizado por su sencillez y eficiencia es el modelo Entidad-Relación.

Modelos lógicos basados en registros

Se utilizan para describir datos en los niveles conceptual y físico. Estos modelos utilizan registro se instancias para representar la realidad, así como las relaciones que existen entre estos registros

(ligas) o apuntadores. A diferencia de los modelos de datos basados en objetos, se usan para especificar la estructura lógica global de la base de datos y para proporcionar una descripción a nivel más alto de la implementación.

Los cuatro modelos de datos más ampliamente aceptados son:

- Modelo jerárquico.
- Modelo de red.
- Modelo relacional.
- Modelo orientado a objetos.

Modelo jerárquico

La forma de representar las relaciones y datos es por medio de registros y sus ligas. La diferencia radica en que están organizados por conjuntos de árboles en lugar de gráficas arbitrarias (Fig. 5).

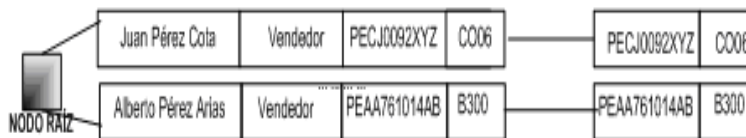


Figura 5. Modelo jerárquico

En este tipo de modelos la organización se establece en forma de árbol, donde la raíz es un nodo ficticio. Así tenemos que, una base de datos jerárquica es una colección de árboles de este tipo.

Modelo de red

Este modelo representa los datos mediante colecciones de registros, sus relaciones se representan por medio de ligas o enlaces, los cuales pueden verse como punteros (Fig. 6).

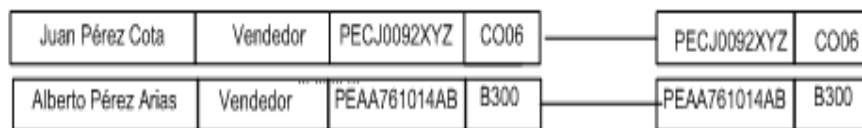


Figura 6. Modelo de red

Una base de datos de red como su nombre lo indica, esta formada por una colección de registros, los cuales están conectados entre sí por medio de enlaces. El registro es similar a una entidad como las empleadas en el modelo entidad-relación. Un registro es una colección de campos (atributos), cada uno de los cuales contiene solamente almacenado un valor, el enlace es la asociación entre dos registros exclusivamente, así que podemos verla como una relación estrictamente binaria.

El modelo relacional

En este modelo se representan los datos y las relaciones entre estos, a través de una colección de tablas, en las cuales los renglones (tuplas) equivalen a cada uno de los registros que contendrá la base de datos y las columnas corresponden a las características (atributos) de cada registro localizado en la tupla, por ejemplo, para la tabla empleado, como lo que muestra la fig. 7:

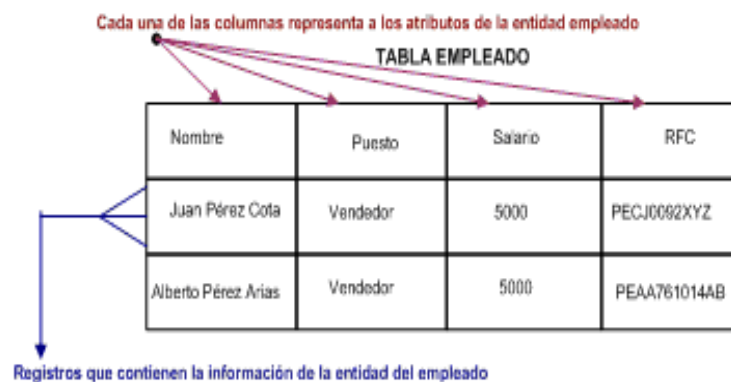


Figura 7. Modelo relacional

En el modelo relacional el único elemento de representación es la tabla y consta de varias equivalencias como se muestra a continuación (fig. 8):



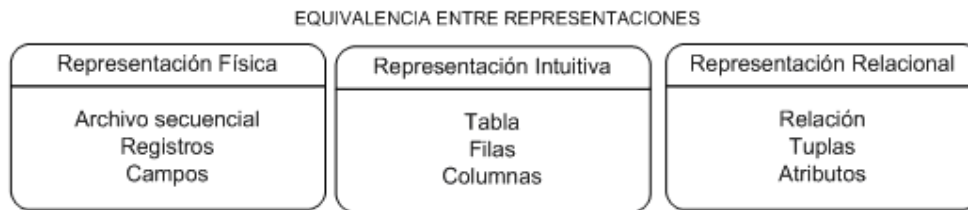


Figura 8. Equivalencia entre representaciones

En el modelo relacional la base de datos es percibida por el usuario como un conjunto de tablas. Esta percepción es sólo a nivel lógico (en los niveles externo y conceptual de la arquitectura de tres niveles), ya que a nivel físico puede estar implementada mediante distintas estructuras de almacenamiento.

La mayoría de los SGBD comerciales actuales están basados en el modelo relacional, mientras que los sistemas más antiguos estaban basados en el modelo de red o el modelo jerárquico. Estos dos últimos modelos requieren que el usuario tenga conocimiento de la estructura física de la base de datos a la que se accede, mientras que el modelo relacional proporciona una mayor independencia de datos.

Modelo orientado a objetos

El modelo de bases de datos orientado a objetos es una adaptación a los sistemas de bases de datos. Se basa en el concepto de encapsulamiento de datos y código que opera sobre éstos en un objeto. Los objetos estructurados se agrupan en clases.

El propósito de los sistemas de bases de datos es la gestión de grandes cantidades de información. Las primeras bases de datos surgieron del desarrollo de los sistemas de gestión de archivos. Estos sistemas primero evolucionaron en bases de datos de red o en bases de datos jerárquicas y, más tarde, en bases de datos relacionales.

La interfaz entre un objeto y el resto del sistema se define mediante un conjunto de mensajes. Un método, es un trozo de código para implementar cada mensaje. Un método devuelve un valor como respuesta al mensaje. Un segundo criterio para clasificar los SGBD es el número de usuarios a los que da servicio el sistema. Los sistemas monousuario sólo atienden a un usuario a la vez, y su

principal uso se da en los ordenadores personales. Los sistemas *multiusuario*, entre los que se encuentran la mayor parte de los SGBD, atienden a varios usuarios al mismo tiempo.

Ventajas

- Facilita la comprensión y organización de un sistema de información, en términos de objetos.
- Facilita documentar las reglas de negocio, si las organizamos y agrupamos en torno a los objetos con los que se relacionan.

Desventajas:

- No hay variedad de software que permitan el modelado y después su alimentación a diversos DBMS.
- La desventaja anterior, obliga a usar herramientas no integradas; es decir, primero diseñar Objeto, por ejemplo, y después elaborar manualmente el diseño derivado de los objetos en otra herramienta, o a emular representación de Objetos semánticos bajo ER en diseñadores de tablas.

Modelos físicos de datos

Se usan para describir a los datos en el nivel más bajo, aunque existen muy pocos modelos de este tipo, básicamente capturan aspectos de la implementación de los sistemas de base de datos.

Funciones de los sistemas de gestión de bases de datos

Edgar Codd, el creador del modelo relacional, ha establecido una lista con los ocho servicios que debe ofrecer todo SGBD.

1. Un SGBD debe proporcionar a los usuarios la capacidad de almacenar datos en la base de datos, acceder a ellos y actualizarlos. Esta es la función fundamental de un SGBD y por supuesto, el SGBD debe ocultar al usuario la estructura física interna (la organización de los ficheros y las estructuras de almacenamiento).
2. Un SGBD debe proporcionar un catálogo en el que se almacenen las descripciones de los datos y que sea accesible por los usuarios. Este catálogo es lo que se denomina diccionario

de datos y contiene información que describe los datos de la base de datos (metadatos).

Normalmente, un diccionario de datos almacena:

- Nombre, tipo y tamaño de los datos.
- Nombre de las relaciones entre los datos.
- Restricciones de integridad sobre los datos.
- Nombre de los usuarios autorizados a acceder a la base de datos.
- Esquema externo, conceptual e interno, y correspondencia entre los esquemas.
- Estadísticas de utilización, tales como la frecuencia de las transacciones y el número de accesos realizados a los objetos de la base de datos.

Algunos de los beneficios que reporta el diccionario de datos son los siguientes:

- La información sobre los datos se puede almacenar de un modo centralizado. Esto ayuda a mantener el control sobre los datos, como un recurso que son.
 - El significado de los datos se puede definir, lo que ayudará a los usuarios a entender el propósito de los mismos.
 - La comunicación se simplifica ya que se almacena el significado exacto. El diccionario de datos también puede identificar al usuario o usuarios que poseen los datos o que los acceden.
 - Las redundancias y las inconsistencias se pueden identificar más fácilmente ya que los datos están centralizados.
 - Se puede tener un historial de los cambios realizados sobre la base de datos.
 - El impacto que puede producir un cambio se puede determinar antes de que sea implementado, ya que el diccionario de datos mantiene información sobre cada tipo de dato, todas sus relaciones y todos sus usuarios.
 - Se puede hacer respetar la seguridad.
 - Se puede garantizar la integridad.
 - Se puede proporcionar información para auditorías.
3. Un SGBD debe proporcionar un mecanismo que garantice que todas las actualizaciones correspondientes a una determinada transacción se realicen, o que no se realice ninguna. Una *transacción* es un conjunto de acciones que cambian el contenido de la base de datos. Una transacción en el sistema informático de la empresa inmobiliaria sería dar de alta a un empleado o eliminar un inmueble. Una transacción un poco más complicada sería eliminar un empleado y reasignar sus inmuebles a otro empleado. En este caso hay que realizar varios

- cambios sobre la base de datos. Si la transacción falla durante su realización, por ejemplo porque falla el hardware, la base de datos quedará en un estado inconsistente. Algunos de los cambios se habrán hecho y otros no, por lo tanto, los cambios realizados deberán ser deshechos para devolver la base de datos a un estado consistente.
4. Un SGBD debe proporcionar un mecanismo que asegure que la base de datos se actualice correctamente cuando varios usuarios la están actualizando concurrentemente. Uno de los principales objetivos de los SGBD es el permitir que varios usuarios tengan acceso concurrente a los datos que comparten. El acceso concurrente es relativamente fácil de gestionar si todos los usuarios se dedican a leer datos, ya que no pueden interferir unos con otros. Sin embargo, cuando dos o más usuarios están accediendo a la base de datos y al menos uno de ellos está actualizando datos, pueden interferir de modo que se produzcan inconsistencias en la base de datos. El SGBD se debe encargar de que estas interferencias no se produzcan en el acceso simultáneo.
 5. Un SGBD debe proporcionar un mecanismo capaz de recuperar la base de datos en caso de que ocurra algún suceso que la dañe. Como se ha comentado antes, cuando el sistema falla en medio de una transacción, la base de datos se debe devolver a un estado consistente. Este fallo puede ser a causa de un fallo en algún dispositivo hardware o un error del software, que hagan que el SGBD aborte, o puede ser a causa de que el usuario detecte un error durante la transacción y la aborte antes de que finalice. En todos estos casos, el SGBD debe proporcionar un mecanismo capaz de recuperar la base de datos llevándola a un estado consistente.
 6. Un SGBD debe proporcionar un mecanismo que garantice que sólo los usuarios autorizados pueden acceder a la base de datos. La protección debe ser contra accesos no autorizados, tanto intencionados como accidentales.
 7. Un SGBD debe ser capaz de integrarse con algún software de comunicación. Muchos usuarios acceden a la base de datos desde terminales. En ocasiones estos terminales se encuentran conectados directamente a la máquina sobre la que funciona el SGBD. En otras ocasiones los terminales están en lugares remotos, por lo que la comunicación con la máquina que alberga al SGBD se debe hacer a través de una red. En cualquiera de los dos casos, el SGBD recibe peticiones en forma de mensajes y responde de modo similar. Todas estas transmisiones de mensajes las maneja el gestor de comunicaciones de datos. Aunque este gestor no forma parte del SGBD, es necesario que el SGBD se pueda integrar con él para que el sistema sea comercialmente viable.

-
-
8. Un SGBD debe proporcionar los medios necesarios para garantizar que tanto los datos de la base de datos, como los cambios que se realizan sobre estos datos, sigan ciertas reglas. La integridad de la base de datos requiere la validez y consistencia de los datos almacenados. Se puede considerar como otro modo de proteger la base de datos, pero además de tener que ver con la seguridad, tiene otras implicaciones. La integridad se ocupa de la calidad de los datos. Normalmente se expresa mediante restricciones, que son una serie de reglas que la base de datos no puede violar. Por ejemplo, se puede establecer la restricción de que cada empleado no puede tener asignados más de diez inmuebles. En este caso sería deseable que el SGBD controlara que no se sobrepase este límite cada vez que se asigne un inmueble a un empleado.

Además, de estos ocho servicios, es razonable esperar que los SGBD proporcionen un par de servicios más:

1. Un SGBD debe permitir que se mantenga la independencia entre los programas y la estructura de la base de datos. La independencia de datos se alcanza mediante las vistas o subesquemas. La independencia de datos física es más fácil de alcanzar, de hecho hay varios tipos de cambios que se pueden realizar sobre la estructura física de la base de datos sin afectar a las vistas. Sin embargo, lograr una completa independencia de datos lógica es más difícil. Añadir una nueva entidad, un atributo o una relación puede ser sencillo, pero no es tan sencillo eliminarlos.
2. Un SGBD debe proporcionar una serie de herramientas que permitan administrar la base de datos de modo efectivo. Algunas herramientas trabajan a nivel externo, por lo que habrán sido producidas por el administrador de la base de datos. Las herramientas que trabajan a nivel interno deben ser proporcionadas por el distribuidor del SGBD. Algunas de ellas son:
 - Herramientas para importar y exportar datos.
 - Herramientas para monitorizar el uso y el funcionamiento de la base de datos.
 - Programas de análisis estadístico para examinar las prestaciones o las estadísticas de utilización.
 - Herramientas para reorganización de índices.
 - Herramientas para aprovechar el espacio dejado en el almacenamiento físico por los registros borrados y que consoliden el espacio liberado para reutilizarlo cuando sea necesario.

Componentes de un sistema de gestión de bases de datos

Los SGBD son paquetes de software complejos y sofisticados que deben proporcionar los servicios comentados en la sección anterior. No se puede generalizar sobre los elementos que componen un SGBD ya que varían mucho unos de otros. Sin embargo, es muy útil conocer sus componentes y cómo se relacionan cuando se trata de comprender lo que es un sistema de bases de datos.

Un SGBD tiene varios módulos, cada uno de los cuales realiza una función específica. El sistema operativo proporciona servicios básicos al SGBD, que es construido sobre él.

- El *procesador de consultas* es el componente principal de un SGBD. Transforma las consultas en un conjunto de instrucciones de bajo nivel que se dirigen al gestor de la base de datos.
- El *gestor de la base de datos* es el interface con los programas de aplicación y las consultas de los usuarios. El gestor de la base de datos acepta consultas y examina los esquemas externo y conceptual para determinar qué registros se requieren para satisfacer la petición. Entonces el gestor de la base de datos realiza una llamada al gestor de ficheros para ejecutar la petición.
- El *gestor de ficheros* maneja los ficheros en disco en donde se almacena la base de datos. Este gestor establece y mantiene la lista de estructuras e índices definidos en el esquema interno. Si se utilizan ficheros dispersos, llama a la función de dispersión para generar la dirección de los registros. Pero el gestor de ficheros no realiza directamente la entrada y salida de datos. Lo que hace es pasar la petición a los métodos de acceso del sistema operativo que se encargan de leer o escribir los datos en el buffer del sistema.
- El *preprocesador del LMD* convierte las sentencias del LMD embebidas en los programas de aplicación, en llamadas a funciones estándar escritas en el lenguaje anfitrión. El preprocesador del LMD debe trabajar con el procesador de consultas para generar el código apropiado.
- El *compilador del LDD* convierte las sentencias del LDD en un conjunto de tablas que contienen metadatos. Estas tablas se almacenan en el diccionario de datos.
- El *gestor del diccionario* controla los accesos al diccionario de datos y se encarga de mantenerlo. La mayoría de los componentes del SGBD acceden al diccionario de datos.

Los principales componentes del gestor de la base de datos son los siguientes:

- *Control de autorización.* Este módulo comprueba que el usuario tiene los permisos necesarios para llevar a cabo la operación que solicita.
- *Procesador de comandos.* Una vez que el sistema ha comprobado los permisos del usuario, se pasa el control al procesador de comandos.
- *Control de la integridad.* Cuando una operación cambia los datos de la base de datos, este módulo debe comprobar que la operación a realizar satisface todas las restricciones de integridad necesarias.
- *Optimizador de consultas.* Este módulo determina la estrategia óptima para la ejecución de las consultas.
- *Gestor de transacciones.* Este módulo realiza el procesamiento de las transacciones.
- *Planificador (scheduler).* Este módulo es el responsable de asegurar que las operaciones que se realizan concurrentemente sobre la base de datos tienen lugar sin conflictos.
- *Gestor de recuperación.* Este módulo garantiza que la base de datos permanece en un estado consistente en caso de que se produzca algún fallo.
- *Gestor de buffers.* Este módulo es el responsable de transferir los datos entre memoria principal y los dispositivos de almacenamiento secundario. A este módulo también se le denomina *gestor de datos*.

Arquitectura Cliente/Servidor

Podemos entender el término Cliente-Servidor como un sistema en el que una máquina cliente solicita a una segunda máquina llamada servidor que ejecute una tarea específica, el cliente suele ser una PC conectada a una red LAN y el servidor, como un servidor de archivos PC o un servidor de archivos UNIX.

El modelo Cliente / Servidor se define como la tecnología que proporciona al usuario final el acceso transparente a las aplicaciones, datos, servicios de cómputo o a cualquier otro recurso dentro del grupo de trabajo y/o a través de la empresa en diferentes plataformas (Fig. 9).

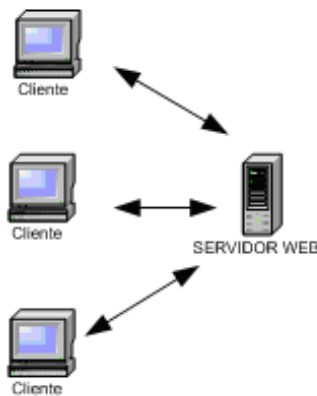


Figura 9. Arquitectura Cliente/servidor

Los clientes interactúan con el usuario, usualmente de forma gráfica y a su vez se comunican con los procesos auxiliares que se encargan de establecer la conexión con el servidor, de enviar el pedido, de recibir la respuesta, manejar las fallas y realizar actividades tanto de sincronización como de seguridad.

Los servidores proporcionan un servicio al cliente y devuelven los resultados, en algunos casos existen procesos auxiliares que se encargan de recibir las solicitudes del cliente, verificar la protección, activar un proceso servidor para satisfacer el pedido, recibir su respuesta y enviarla al cliente.

Una de las cosas por la que se distingue el modelo Cliente/Servidor, es que su procesamiento es distribuido entre aplicaciones independientes.

Esta arquitectura se conforma de dos tipos componentes que se comunican a través de una red: La red proporciona la conexión entre los clientes y los servidores

- Back-end: Servidor
- Front-end: Cliente

El Servidor procesa las peticiones que hacen los clientes, y cuando es posible regresa el resultado. Mantiene la integridad lógica y el acceso de los datos. El Cliente envía peticiones al servidor y manipula las respuestas. Éste puede: desplegar y manejar el ambiente de trabajo de la aplicación y la interfaz de usuario, llevar a cabo la validación de datos, desplegar reportes y representar datos gráficamente (Fig. 10).

La secuencia de eventos cuando un usuario accesa al servidor de bases de datos se puede generalizar en los siguientes pasos.

- El usuario crea su consulta sobre los datos.
- El cliente formatea la consulta en lenguaje SQL y la envía a través de la red.
- El servidor de base de datos verifica los permisos sobre los datos a consultar.
- El servidor de base de datos procesa la consulta y regresa los resultados.
- El cliente recibe la respuesta y la presenta al usuario.
- El usuario visualiza y manipula los datos y reinicia el proceso.

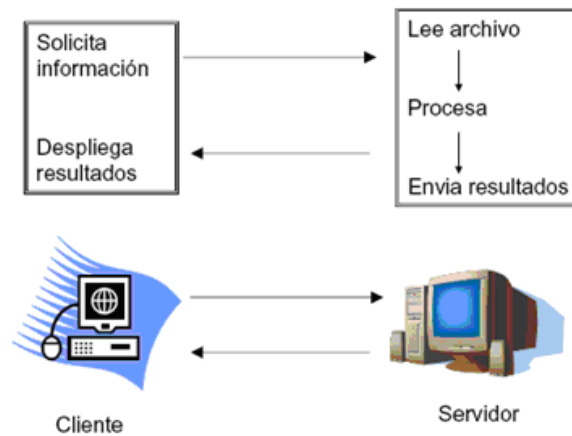


Figura 10. Secuencia de eventos para el acceso al servidor

I.4 MODELO RELACIONAL.

El trabajo publicado por Codd en ACM (1970) presentaba un nuevo modelo de datos que perseguía una serie de objetivos, que se resumen en las siguientes líneas:

Independencia física. El modo en el que se almacenan los datos no influye en su manipulación lógica y por tanto, los usuarios que acceden a esos datos no tienen que modificar sus programas por cambios en el almacenamiento físico.

Independencia lógica. El añadir, eliminar o modificar objetos de la base de datos no repercute en los programas y/o usuarios que están accediendo a subconjuntos parciales de los mismos (vistas).

Flexibilidad. En el sentido de poder presentar a cada usuario los datos de la forma en que éste prefiera.

Uniformidad. Las estructuras lógicas de los datos presentan un aspecto uniforme, lo que facilita la concepción y manipulación de la base de datos por parte de los usuarios. Sencillez. Las características anteriores, así como unos lenguajes de usuario muy sencillos, producen como resultado que el modelo de datos relacional sea fácil de comprender y de utilizar por parte del usuario final.

En el modelo relacional Los datos se almacenan como tablas y las relaciones entre las filas y las tablas son visibles en los datos. Este enfoque permite a los usuarios obtener información de la base de datos sin asistencia de sistemas profesionales de administración de información.

Es importante saber que las entradas en la tabla tienen un solo valor (son atómicos); no se admiten valores múltiples, por lo tanto la intersección de un renglón con una columna tiene un solo valor, nunca un conjunto de valores.

Todas las entradas de cualquier columna son de un solo tipo. Por ejemplo, una columna puede contener nombres de clientes, y en otra puede tener fechas de nacimiento. Cada columna posee un nombre único, el orden de las columnas no es de importancia para la tabla, las columnas de una tabla se conocen como atributos. Cada atributo tiene un dominio, que es una descripción física y lógica de valores permitidos.

No existen 2 filas en la tabla que sean idénticas. La información en las bases de datos son representados como datos explícitos, no existen apuntadores o ligas entre las tablas.

La arquitectura relacional consta de los siguientes componentes:

Modelo relacional de datos: En el nivel conceptual, el modelo relacional de datos está representado por una colección de relaciones almacenadas. Cada registro de tipo conceptual en un modelo relacional de datos se implanta como un archivo almacenado distinto.

Sub-modelo de datos: Los esquemas externos de un sistema relacional se llaman sub-modelos relacionales de datos; cada uno consta de uno a más escenarios (vistas) para describir los datos requeridos por una aplicación dada. Un escenario puede incluir datos de una o más tablas de datos. Cada programa de aplicación está provisto de un buffer ("Área de trabajo de usuario") donde el DBMS puede depositar los datos recuperados de la base para su procesamiento, o puede guardar temporalmente sus salidas antes de que el DBMS las escriba en la base de datos.

Esquema de almacenamiento: En el nivel interno, cada tabla base se implanta como un archivo almacenado. Para las recuperaciones sobre las llaves principal o secundaria se pueden establecer uno o más índices para acceder a un archivo almacenado.

Sub-lenguaje de datos: Es un lenguaje de manejo de datos para el sistema relacional, el álgebra relacional y cálculo relacional, ambos lenguajes son "relacionalmente completos", esto es, cualquier relación que pueda derivarse de una o más tablas de datos, también se puede derivar con un solo comando del sub-lenguaje. Por tanto, el modo de operación de entrada/Salida en un sistema relacional se puede procesar en la forma: una tabla a la vez en lugar de: un registro a la vez; en otras palabras, se puede recuperar una tabla en vez de un solo registro con la ejecución de un comando del sub-lenguaje de datos.

Reglas de Cood

1. Cualquier BDMS que proclame ser relacional, deberá manejar, completamente, las bases de datos por medio de sus capacidades relacionales.
2. Regla de información. Toda la información contenida dentro de una base de datos se representa de manera explícita a nivel lógico y exactamente de una sola manera, como valores en una tabla.
3. Regla de acceso garantizado. Se garantiza que todos y cada uno de los datos (valor atómico) en una base de datos relacional puede ser leído recurriendo a una combinación del nombre de la tabla, valor de la llave primaria y el nombre de la columna.
4. El manejo sistemático de los valores nulos. En un DBMS totalmente relacional soporta los valores nulos para representar información faltante o no aplicable de una forma consistente, independiente del tipo de dato.
5. Catálogo dinámico en línea basado en un modelo relacional. La descripción de la base de datos se representa en el nivel lógico de la misma forma que los datos ordinarios, de tal suerte que los usuarios autorizados puedan aplicar el mismo lenguaje relacional para consultarla, que aquél que emplean para con sus datos habituales.
6. Regla del sub-lenguaje de dato completo. Se debe contar con un sub-lenguaje que defina los datos, los manipule y que controle la seguridad de los datos.

7. Regla de actualización de vistas. Todas las vistas que sean teóricamente actualizables deberán ser actualizadas por medio del sistema.
8. Inserción, actualización y eliminación de alto nivel. La posibilidad de manejar una relación base o una relación derivada como un solo operador se aplica a la lectura, inserción, modificación y eliminación de datos.
9. Independencia física de los datos. Los programas de aplicación y la actividad en terminales no deberán ser afectados por cambios en el almacenamiento físico de los datos o en el método de acceso.
10. Independencia lógica de los datos. Los programas de aplicación y las actividades en terminales no deberán ser afectados por cambios de cualquier tipo que preserven la información y que teóricamente permitan la no afectación de las tablas base.
11. Independencia de la integridad. Las restricciones de integridad de una base de datos deberán ser definidas en el mismo sublenguaje de datos relacional y deberán almacenarse en el catálogo, no en los programas de aplicación.
12. Independencia de la distribución. Un DBMS relacional tiene independencia de distribución.
13. Regla de la no subversión. Si un sistema relacional tiene un lenguaje de bajo nivel, ese bajo nivel no puede ser utilizado para suprimir las reglas de integridad y las restricciones expresadas en el lenguaje relacional de nivel superior.

Base de datos relacionales

Una base de datos relacional es una base de datos en donde todos los datos visibles al usuario están organizados estrictamente como tablas de valores, y en donde todas las operaciones de la base de datos operan sobre estas tablas.

Características más importantes de las Bases de Datos Relacionales:

- Cada tabla contiene un solo tipo de registro.
- Los campos no tienen un orden específico, de izquierda a derecha.
- Los registros no tienen un orden específico de arriba hacia abajo.

- Cada campo tiene un solo valor.
- Los registros poseen un campo identificador único, llamado llave primaria.

Así, todos los datos en una base de datos relacional se representan de una y sólo una manera, a saber, por su valor explícito (ésta se denomina en ocasiones “principio básico del modelo relacional”). En particular, las conexiones lógicas dentro de una relación y entre las relaciones se representan mediante esos valores; no existen “ligas” o apuntadores visibles para el usuario, ni ordenamientos visibles para el usuario, ni grupos repetitivos visibles para el usuario, etc.

1.5 LENGUAJE DE CONSULTA SQL (STRUCTURED QUERY LANGUAGE).

SQL (Structured Query Language) es un lenguaje de consultas para base de datos, siendo adoptado como estándar en el año 1989. Las ventajas de adopción del ANSI SQL, es que los diversos RDBMS tienen que acoplarse al estándar, permitiendo así una mayor compatibilidad entre ellos. Algunos fabricantes, aunque tratan de acoplarse al estándar ANSI SQL, cada uno de ellos implementa funcionalidades extras que le dan un valor agregado a su producto, por lo cual se pueden notar ciertas diferencias entre los distintos RDBMS.

Este lenguaje se basa en el álgebra relacional que consiste en una colección de operaciones sobre relaciones donde cada operación toma una o más relaciones como su operando y produce otra relación como su resultado. Dado que el resultado de una operación del álgebra relacional es una relación, ésta a su vez puede ser sujeto de posteriores operaciones algebraicas.

El álgebra relacional se basa en la teoría de conjuntos, relaciones y en el álgebra de conjuntos.

Adicionalmente al conjunto básico de operadores como: unión, diferencia, producto cartesiano e intersección; incorpora operadores específicos de base de datos tales como proyección, selección y join.

Las bases de datos relacionales están basadas en el concepto matemático de relaciones entre conjuntos. Así las operaciones que se pueden efectuar entre relaciones son tanto las comunes a los conjuntos, unión, intersección, diferencia, producto cartesiano; como las específicas de las relaciones, selección, proyección, etc.

Si q , r y s son relaciones con todos los dominios iguales, esto es, con el mismo esquema, se les puede aplicar las operaciones típicas de conjuntos.

Unión Es la relación sobre los mismos dominios que contiene las eneadas que están en R, en S o en ambas. Esto es, son todos los elementos que se encuentran en el conjunto S y en el R. Construye una relación formada por todas las tuplas que aparecen en cualquiera de las dos relaciones especificadas o la suma de sus elementos como lo muestra la Fig. 11.

R			S			R ∪ S		
A	B	C	A	B	C	A	B	C
a1	b1	c1	a1	b1	c1	a1	b1	c1
a2	b2	c1	a2	b2	c1	a1	b2	c1
a2	b1	c2	a2	b2	c2	a2	b1	c2
a2	b2	c2	a2	b2	c2	a2	b2	c2

Figura 11. Unión de conjuntos

Intersección r/s ó $r \cap s$: Es la relación que contiene las eneadas que están en r y en s. Esto es, construye una relación formada por aquellas tuplas que aparezcan en las dos relaciones especificadas (fig. 12).

r			s			$r \cap s$		
A	B	C	A	B	C	A	B	C
a1	b1	c1	a1	b1	c1	a1	b1	c1
a2	b2	c1	a2	b2	c1	a2	b2	c1
a2	b1	c2	a2	b2	c2			

Figura 12. Intersección de conjuntos

Diferencia $r - s$: Es la relación con las eneadas que están en r pero no en s. Corresponde a obtener los elementos del conjunto r que no se encuentran en el conjunto s (fig.13).

r			s			$r - s$		
A	B	C	A	B	C	A	B	C
a1	b1	c1	a1	b1	c1	a2	b1	c2
a2	b2	c1	a2	b2	c1			
a2	b1	c2	a2	b2	c2	$s - r$		
						A	B	C
						a2	b2	c2

Figura 13. Diferencia en relación de conjuntos

Producto cartesiano $r \times s$: Obtiene todas las eneadas que se construyen concatenando cada eneada de r con otra de s . En este caso los dominios de r y s no tienen que ser los mismos. A partir de dos relaciones especificadas, construye una relación que contiene todas las combinaciones posibles de tuplas, una de cada una de las dos, esto es, los pares ordenados (fig. 14)

r			s			A	B	C	D	E	F
A	B	C	D	E	F	a1	b1	c1	d1	e1	f1
a1	b1	c1	d1	e1	f1	a1	b1	c1	d2	e2	f1
a2	b2	c1	d2	e2	f1	a1	b1	c1	d2	e2	f2
a2	b2	c1	d2	e2	f1	a2	b2	c1	d1	e1	f1
a2	b1	c2	d2	e2	f2	a2	b2	c1	d2	e2	f2
						a2	b1	c2	d1	e1	f1
						a2	b1	c2	d2	e2	f1
						a2	b1	c2	d2	e2	f2

Figura 14. Producto cartesiano en relación de conjuntos

Proyección La proyección selecciona y genera un subconjunto con los atributos indicados de una tabla. También es conocida como operación vertical (fig. 15).

CLAVE	RFC	NOMBRE	DEPTO	SUELDO
125	LOPA650303	PEDRO LOPEZ	1	\$ 250.00
236	PERM640506	MARIA PEREZ	1	\$ 350.00
554	GOMA700808	ALBERTO GOMEZ	2	\$ 350.00
698	ALLO601212	OSCAR ALVAREZ	3	\$ 300.00
741	SUOP701231	PABLO SUAREZ	2	\$ 500.00
254	TENN680409	MONICA TELLEZ	3	\$ 350.00

NOMBRE	DEPTO	SUELDO
PEDRO LOPEZ	1	\$ 250.00
MARIA PEREZ	1	\$ 350.00
ALBERTO GOMEZ	2	\$ 350.00
OSCAR ALVAREZ	3	\$ 300.00
PABLO SUAREZ	2	\$ 500.00
MONICA TELLEZ	3	\$ 350.00

Figura 15. Proyección en relación de conjuntos

Selección La selección toma y genera un subconjunto con los renglones indicados de una tabla. También es conocida como operación horizontal (fig. 16).

CLAVE	RFC	NOMBRE	DEPTO	SUELDO
125	LOPA650303	PEDRO LOPEZ	1	\$ 250.00
236	PERM640506	MARIA PEREZ	1	\$ 350.00
554	GOMA700808	ALBERTO GOMEZ	2	\$ 350.00
698	ALLO601212	OSCAR ALVAREZ	3	\$ 300.00
741	SUOP701231	PABLO SUAREZ	2	\$ 500.00
254	TENN680409	MONICA TELLEZ	3	\$ 350.00

CLAVE	RFC	NOMBRE	DEPTO	SUELDO
125	LOPA650303	PEDRO LOPEZ	1	\$ 250.00
236	PERM640506	MARIA PEREZ	1	\$ 350.00
554	GOMA700808	ALBERTO GOMEZ	2	\$ 350.00

Figura 16. Selección en relación de conjuntos

Join. La operación join es en esencia un producto cartesiano, donde se seleccionan las columnas que satisfagan las condiciones indicadas. Es la operación más común en las bases de datos relacionales (fig. 17).

CLAVE	NOMBRE	DEPTO
125	JUAN MENDEZ	1
365	MARIA NAJERA	2
698	JESUS MORALES	1

DEPTO	NOMBRE DEPTO
1	GERENCIA
2	CONTABILIDAD

CLAVE	NOMBRE	DEPTO	NOMBRE DEPTO
125	JUAN MENDEZ	1	GERENCIA
365	MARIA NAJERA	2	CONTABILIDAD
698	JESUS MORALES	1	GERENCIA

Figura 17. Join en relación de conjuntos

Los componentes de SQL son:

- DDL (Data Definition Language; Lenguaje de Definición de Datos). Permite crear, modificar y eliminar la estructura de la base de datos.
- DML (Data Manipulation Language; Lenguaje de Manipulación de Datos). Permite consultar, insertar, modificar, y eliminar datos de las tablas.
- DCL (Data Control Language; Lenguaje de Control de Datos). Permite establecer privilegios de acceso a los datos.

Definición de datos

Dentro de este modulo, se crea, modifica y elimina la estructura de la base de datos como son: tablas, índices, triggers, store procedure, etc.

Reglas

Las reglas son condiciones que deben cumplir los datos para que sea válida.

El exceso de reglas disminuye el rendimiento del RDBMS en los procesos de inserción y modificación de la información.

En algunos DBMS como SYBASE y SQL SERVER las reglas se pueden crear como objetos independientes que se vinculan a distintas tablas. En cambio ORACLE y POSTGRESQLSQL no son objetos independientes, sino que son restricciones que afecta a una sola tabla.

Defaults

Los defaults establecen que valor será registrado de manera predeterminada para una columna, en caso de no especificar al momento de introducir los datos. Los defaults dependiendo del DBMS como SYBASE y SQL SERVER pueden ser objetos independientes, mientras que en ORACLE, POSTGRESQLSQL y MYSQL los defaults están ligados a un solo campo.

Llaves e índices

Las llaves son identificadoras que diferencian entre un registro y otro además de mejorar el proceso de consulta de los datos. Existen las llaves primarias que son identificadores únicos dentro de una tabla; también están las llaves foráneas que son las columnas que hacen referencia a la llave primaria de otra tabla y a través de estas se pueden establecer relaciones entre tablas.

Llave primaria: Es aquel atributo que identifica de manera única a un registro.

Esto es, no debe haber dos tuplas que tengan el mismo valor, por lo tanto, con sólo conocer el valor de la llave primaria para una determinada tupla será suficiente para identificarlo de manera única.

Llave candidata: Es el atributo o conjunto de atributos que podrían servir como llaves primarias.

Una llave candidata debe cumplir dos condiciones:

- Unicidad: no pueden existir dos tuplas con el mismo valor en todos los atributos que forman la llave candidata.
- Minimidad: no existe ningún subconjunto de la llave que cumpla la regla de unicidad.

Un índice es una estructura de almacenamiento físico que permite recuperar datos de manera eficiente.

Llave secundaria o alterna: Son aquellas llaves candidatas que no se eligieron como llave primaria, es decir, tienen todas las características para ser llaves primarias, pero que por alguna razón no fueron tomadas como tal debido quizás a que hubo otra que cumplía mejor con ese objetivo.

Llave foránea: Es una llave primaria en otra relación, éstas representan las asociaciones entre las diferentes entidades, es decir, son llaves que están siendo compartidas por dos tablas para formar una relación entre ellas.

Manipulación de datos

La mayoría del trabajo con SQL gira en torno a cuatro comandos:

- Select. Permite recuperar información de la tabla
- Insert. Permite agregar información a la tabla.
- Delete. Permite eliminar información de la tabla.
- Update. Permite actualizar información que existe en la tabla.

La independencia de los datos Una de las principales ventajas que provee una base de datos es la independencia entre los datos y los tratamientos que se hacen de ellos ya que en los sistemas orientados a procesos los datos eran sumamente dependientes de los programas.

Como tal, la independencia de los datos se refiere a la protección contra los programas de aplicación que puedan originar modificaciones cuando se altera la organización física o lógica de la base de datos existen 2 niveles de independencia de datos:

- Independencia física de datos
- Independencia lógica de datos

Independencia física de los datos

Es la capacidad de modificar el esquema físico sin provocar que se vuelvan a escribir los programas de aplicación. Dicho de otra forma, el esquema conceptual no es afectado por cambios al esquema físico de datos.

Es decir se refiere al ocultamiento de los detalles sobre las estructuras de almacenamiento a las aplicaciones de usuario, o sea la descripción física de datos puede cambiar sin afectar a las aplicaciones de usuario.

Si el DBMS modifica su organización interna de ficheros, no pasa nada con el esquema conceptual.

Ejemplos:

- Los datos pueden ser movidos de un disco a otro.
- La organización de los datos puede cambiar.
- Cuando hay un cambio de versión.
- Al migrar a otro DBMS.

Independencia lógica de los datos

Capacidad de modificar el esquema conceptual sin provocar que se vuelvan a escribir los programas de aplicación.

Algunos elementos del esquema externo no son afectados por cambios al esquema conceptual.

Ejemplos:

- Modificar columnas sólo afecta las vistas que incluyen esas columnas.
- Agregar más columnas no afecta las vistas.
- La creación de una nueva relación.
- El reordenamiento lógico de algunos atributos.

La integridad referencial

El término de integridad referencial se enmarca en la segunda regla de integridad y se aplica a las llaves foráneas:

“Si en una relación hay alguna llave foránea, sus valores deben coincidir con valores de la llave primaria a la que hace referencia, o bien, deben ser completamente nulos”.

Lo que en realidad trata de decir el texto anterior es que las llaves foráneas no pueden dejar de tener correspondencia con la llave primaria de la tabla externa.

El modelo entidad-relación

El modelo Entidad-Relación es una técnica de diseño de base de datos gráfica, que incorpora información relativa a los datos y las relaciones existentes entre ellos.

Entre sus principales características esta:

- Sólo representa la existencia de los datos sin expresar lo que se hace con ellos.
- Es independiente del DBMS y del sistema operativo.
- La inclusión de todos los datos sin considerar las aplicaciones que tendrá.

Los elementos que componen este modelo son:

Entidad: Se puede definir como entidad a cualquier objeto, real o abstracto, que existe en un contexto determinado o que puede llegar a existir y del cual deseamos guardar información, por ejemplo, un profesor, un alumno o bien una materia.

Las entidades las podemos clasificar en:

- Relación Fuertes: Son aquellas entidades que existen por si mismas, es decir, la existencia de un ejemplar de la entidad no depende de la existencia de otros ejemplares en otra entidad, por ejemplo, la entidad "PROFESOR" (fig. 18).



Figura 18 Ejemplo de entidad fuerte

- Relación Débiles: Son aquellas entidades en las que su existencia depende de la existencia de ejemplares en otras entidades, por ejemplo, la existencia de la entidad "PROFESOR" depende de la existencia de la entidad "ESCUELA" (fig. 19).



Figura 19. Ejemplo de entidad débil

Atributos

Las entidades se componen de atributos que son cada una de las propiedades o características que tienen las entidades. Cada ejemplar de una misma entidad posee los mismos atributos, tanto en nombre como en número, diferenciándose cada uno de los ejemplares por los valores que toman dichos atributos.

Cada atributo tiene un conjunto de valores asociados denominado dominio.

Dominios

Se define dominio como un conjunto de valores que puede tomar un determinado atributo dentro de una entidad

Llaves

El modelo Entidad -Relación exige que cada entidad tenga un identificador, se trata de un atributo o conjunto de atributos que identifican de forma única a cada uno de los ejemplares de la entidad.

Relación Un ejemplo de identificador es el atributo "RFC" (fig.20), que en la entidad "PROFESOR", identifica de forma única a cada uno de los profesores. Estos identificadores reciben el nombre de Llave Primaria o PrimaryKey (PK). Como ya se había mencionado antes, puede ser que existan más identificadores, a estos atributos se les conoce como Identificadores Candidatos (IC).



Figura 20. Ejemplo de l identificador RFC

- Interrelaciones: Es la asociación o vinculación entre entidades. Las interrelaciones se clasifican en débiles y regulares según el tipo de entidad que estén asociando, las relaciones débiles se clasifican en:
 - Interrelación en existencia: Se da cuando los ejemplares de la entidad débil no puede existir si desaparece el ejemplar de la entidad regular de la que depende.
 - Interrelación en identificación: Cuando además de ser una relación en existencia, los ejemplares de la entidad no se pueden identificar por si mismo y exige añadir el identificador principal de la entidad regular del cual depende para ser identificado.

Cardinalidad de las Relaciones

El diseño de relaciones entre las tablas de una base de datos puede ser la siguiente:

- **Relaciones de uno a uno:** una instancia de la entidad A se relaciona con una y solamente una de la entidad B.
- **Relaciones de uno a muchos:** cada instancia de la entidad A se relaciona con varias instancias de la entidad B.
- **Relaciones de muchos a muchos:** cualquier instancia de la entidad A se relaciona con cualquier instancia de la entidad B.

El Diagrama Entidad-Relación (DER) se utiliza como una herramienta de comunicación entre los analistas y diseñadores de sistemas y los usuarios finales durante las fases de análisis de requerimientos y de diseño conceptual debido a que es simple y fácil de entender.

El DER se basa en una percepción de un mundo real (abstracción), que consiste en un conjunto de objetos básicos llamados entidades y relaciones.

En un diagrama entidad-relación (DER) las entidad se representan con un rectángulo; un sustantivo en español corresponde al nombre de la entidad en el DER. Tienen propiedades o atributos (fig. 21).

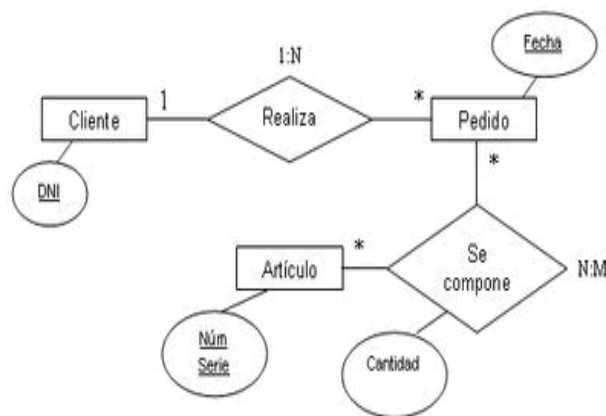


Figura 21. Modelo Entidad/relación

Relación. Representa la asociación o correspondencia entre 2 entidades o de una entidad consigo misma; es binaria, es decir, puede leerse en dos direcciones o sentidos. Asocia una entidad de un conjunto a una o varias entidades de otro. En un DER las relaciones se representan con líneas conectando las entidades relacionadas; normalmente un verbo corresponde a la relación.

Normalización

Es el proceso que consiste en agrupar a los campos de datos en un conjunto de tablas que representan a las entidades, sus características y sus relaciones de forma adecuada.

El objetivo de la normalización es asegurar que el modelo conceptual de la base de datos funcionará.

Las formas normales fueron propuestas por Codd entre los años 1971 y 1972, estas formas normales son una serie de restricciones para evitar anomalías en las tablas.

Las formas normales son:

- Primera forma normal. Una relación esta en primera forma normal si y solo si todos los dominios de la misma contienen valores atómicos, es decir no hay grupos repetitivos.
- Segunda forma normal. Una relación esta en segunda forma normal si y solo si, está en 1FN y además, cada atributo que no está en la clave primaria es completamente dependiente de la clave primaria.
- Tercera forma normal. Una relación esta en tercera forma normal si y solo si, cumple con la 1FN, 2FN y no deben de existir campos, que a partir de ellos pueda conocer otros datos de otra tabla.

Las formas normales son una serie de restricciones que se definen sobre las estructuras relacionales para evitar anomalías al efectuar adiciones, eliminaciones o actualizaciones de tuplas (fig.22).

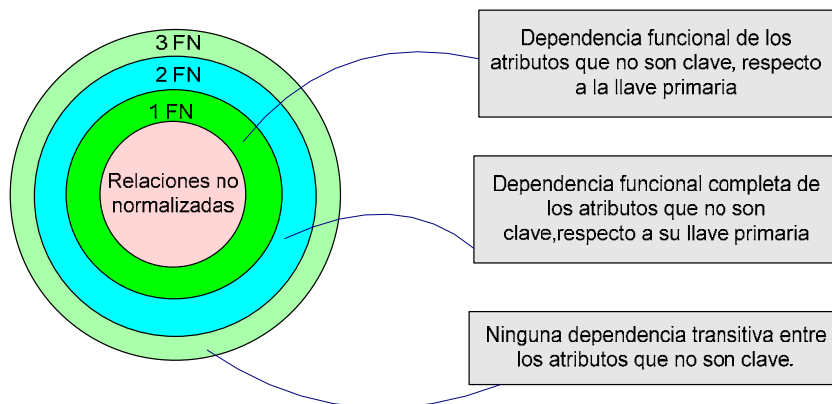


Figura 22. Formas normales

Esquema de seguridad en el RDBMS

Dentro del RDBMS se puede encontrar un acceso multicapas (fig.23) que consiste en lo siguiente:

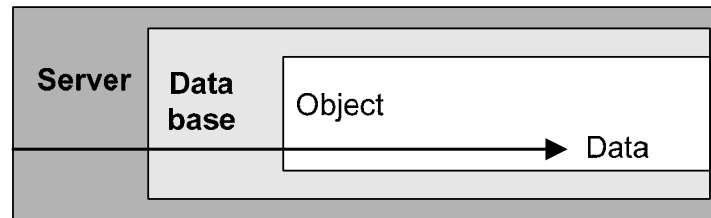


Figura 23. Esquema de seguridad

Seguridad a nivel servidor. En donde el usuario final debe tener una cuenta válida dentro de la capa del servidor RDBMS.

- Seguridad a nivel de Base de Datos. El usuario final debe ser un usuario válido dentro de la capa de la base de datos.
- Seguridad a Nivel de Objetos y comandos. El usuario final deberá tener permisos dentro de la capa de los datos.

Ejemplos de RDBMS

Los RDBMS más comúnmente utilizados, tanto comerciales como de licencia libre son:

Por Licencia:

- Microsoft SQL Server
- Sybase
- Oracle
- Informix
- BD2

De software libre:

- PostgreSQL
- MySQL

PostgreSQL

Características principales:

- Base de datos de distribución libre.
- Velocidad.
- Confiabilidad.
- Flexibilidad.
- Bajo costo de operación.
- Conformación a estándares ANSI.
- Estrategia de almacenamiento MVCC para grandes volúmenes.

Soporta replicación de bases de datos.

- Interfases nativas para ODBC, JDBC, C, C++, PHP, Perl, TCL, XML.
- Soporta SSL nativo.
- Plataformas que soporta:
- Corre bajo cualquier plataforma UNIX.

Usuarios en una base de datos

- Programador de aplicaciones.
- Usuario final, el cual tiene acceso a los datos de la base a través de algunas aplicaciones desarrolladas.
- El Administrador de la Base de Datos.

Roles o funciones

Los roles, sirven como medio para conceder privilegios, sobre todo un sistema a un usuario que lo requiera, como:

- Administrador de la Base de Datos (DBA).
- Dueño de la Base de Datos (DBO).
- Oficial del Sistema de Seguridad (SSO).
- Operador.

Características principales de PostgreSQL:

- Base de datos de distribución libre.
- Velocidad.
- Confiabilidad.
- Flexibilidad.
- Bajo costo de operación.
- Conformación a estándares ANSI.
- Estrategia de almacenamiento MVCC para grandes volúmenes.
- Soporta replicación de bases de datos.
- Interfases nativas para ODBC, JDBC, C, C++, PHP, Perl, TCL, XML.
- Soporta SSL nativo.

Plataformas que soporta:

Corre bajo cualquier plataforma UNIX.

PostgreSQL es un potente sistema de base de datos relacional libre (open source, su código fuente está disponible) liberado bajo licencia BSD. Tiene más de 15 años de activo desarrollo y arquitectura probada que se ha ganado una muy buena reputación por su confiabilidad e integridad de datos.

Funciona en todos los sistemas operativos importantes, incluyendo Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64), y Windows.

El desarrollo de PostgreSQL es realizado por un equipo de desarrolladores (voluntarios en su mayoría) dispersos alrededor del mundo y comunicados vía Internet. Este es un proyecto de la comunidad y no es controlado por ninguna compañía.

Las principales mejoras en PostgreSQL incluyen:

Los bloqueos de tabla han sido sustituidos por el control de concurrencia multi-versión, el cual permite a los accesos de sólo lectura continuar leyendo datos consistentes durante la actualización de registros, y permite copias de seguridad en caliente desde `pg_dump` mientras la base de datos permanece disponible para consultas.

Se han implementado importantes características del motor de datos, incluyendo subconsultas, valores por defecto, restricciones a valores en los campos (constraints) y disparadores (triggers).

Se han añadido funcionalidades en línea con el estándar SQL92, incluyendo claves primarias, identificadores entrecomillados, forzado de tipos cadenas literales, conversión de tipos y entrada de enteros binarios y hexadecimales.

Los tipos internos han sido mejorados, incluyendo nuevos tipos de fecha/hora de rango amplio y soporte para tipos geométricos adicionales.

La velocidad del código del motor de datos ha sido incrementada aproximadamente en un 20-40%, y su tiempo de arranque ha bajado el 80% desde que la versión 6.0 fue lanzada.

I.6 PROGRAMACIÓN DE CLIENTES

Debido a que el objetivo de la base que alberga los datos, es mantener los datos almacenados, organizados, relacionados entre si, para después ser procesados y compartidos entre los diferentes usuarios autorizados, a los cuales le son útiles los datos, que una vez procesados son información; se propone estos lenguajes de programación cliente, que proporcionan además de sus elementos básicos del lenguaje, un conjunto de elementos que nos permiten interaccionar con los RDBMS para obtener o proporcionar datos a la base de datos, formulando así la interacción del usuario, en cualquier parte del mundo a través de un programa cliente, con la base de datos en un lugar completamente distinto al del usuario.

I.7 PHP

PHP es un lenguaje de script interpretado en el lado del servidor utilizado para la generación de páginas Web dinámicas, similar al ASP de Microsoft o el JSP de Sun, embebido en páginas HTML y ejecutado en el servidor.

La mayor parte de su sintaxis ha sido tomada de C, Java y Perl con algunas características específicas de sí mismo. La meta del lenguaje es permitir rápidamente a los desarrolladores la generación dinámica de páginas. No es un lenguaje de marcas como podría ser HTML, XML o WML. Está más cercano a JavaScript o a C, para aquellos que conocen estos lenguajes.

Permite también embeber pequeños fragmentos de código dentro de las páginas HTML y realizar determinadas acciones de una forma fácil y eficaz sin tener que generar programas íntegramente en un lenguaje distinto al HTML.

Funcionamiento

A diferencia de Java o JavaScript que se ejecutan en el navegador PHP se ejecuta en el servidor por eso nos permite acceder a los recursos que tenga el servidor como por ejemplo podría ser una base de datos. El programa PHP es ejecutado en el servidor y el resultado es enviado al navegador. El resultado es normalmente una página HTML pero también podría ser una página WML (Wap).

Al ser PHP un lenguaje que se ejecuta en el servidor no es necesario que su navegador lo soporte, es independiente del navegador, sin embargo, para que sus páginas PHP funcionen el servidor donde están alojadas debe soportar PHP (fig. 24).



Figura 24. Funcionamiento del lenguaje PHP con el servidor

Características

Al ser un lenguaje libre dispone de una gran cantidad de características que lo convierten en la herramienta ideal para la creación de páginas web dinámicas:

Soporte para una gran cantidad de bases de datos: MySQL, PostgreSQL, Oracle, MS SQL Server, Sybase mSQL, Informix, entre otras.

Integración con varias bibliotecas externas, permite generar documentos en PDF (documentos de Acrobat Reader) hasta analizar código XML.

Ofrece una solución simple y universal para las paginaciones dinámicas del Web de fácil programación.

Perceptiblemente más fácil de mantener y poner al día que el código desarrollado en otros lenguajes.

Soportado por una gran comunidad de desarrolladores, como producto de código abierto, PHP goza de la ayuda de un gran grupo de programadores, permitiendo que los fallos de funcionamiento se encuentren y reparen rápidamente.

El código se pone al día continuamente con mejoras y extensiones de lenguaje para ampliar las capacidades de PHP.

Con PHP se puede hacer cualquier cosa que podemos realizar con un script CGI, como el procesamiento de información en formularios, foros de discusión, manipulación de cookies y páginas dinámicas.

Un sitio con páginas webs dinámicas es aquel que permite interactuar con el visitante, de modo que cada usuario que visita la página vea la información modificada para propósitos particulares.

Seguridad

PHP es un potente lenguaje y el intérprete, tanto incluido en el servidor Web como módulo o ejecutado como un binario CGI, puede acceder a ficheros, ejecutar comandos y abrir comunicaciones de red en el servidor. Todas estas características hacen que lo que se ejecute en el servidor Web sea seguro por defecto.

PHP ha sido diseñado específicamente para ser un lenguaje más seguro para escribir programas CGI, Perl o C y con la correcta selección de las opciones de configuración de tiempo de compilación y ejecución se consigue la exacta combinación de libertad y seguridad que se necesita. Ya que existen diferentes modos de utilizar PHP, existe también una multitud de opciones de configuración que permiten controlar su funcionamiento. Una gran selección de opciones garantiza que se pueda usar PHP para diferentes aplicaciones, pero también significa que existen combinaciones de estas opciones y configuraciones del servidor que producen instalaciones inseguras.

I.8 ADMINISTRACIÓN DE BASES DE DATOS

La administración de datos consiste en la planificación y coordinación requerida para definir datos de manera congruente en toda empresa o compañía. Una persona o grupo debe tener la responsabilidad de determinar que datos deben reunirse, como deben guardarse y fomentar modos de utilizar dichos datos. Esta persona o grupo es responsable por la integridad de los datos.

La administración de una base de datos consiste en los aspectos técnicos de creación y ejecución de la base de datos. Las tareas básicas son el monitoreo de desempeño, el respaldo y la recuperación, así como la asignación y control de la seguridad. Los administradores de bases de datos están capacitados en los detalles de instalación configuración y operación del DBMS.

La seguridad de las bases de datos es un grupo secundario dentro de los temas de seguridad para la computadora. Sin embargo debido al objetivo de compartir datos, la seguridad es un problema crucial en la administración de una base de datos. Además todos los desarrolladores de aplicaciones y administradores de bases de datos deben estudiar algunos giros interesantes dentro de la seguridad para la base de datos.

A pesar del poder y flexibilidad de los sistemas de bases de datos, las aplicaciones construidas en diferentes momentos por diferentes personas no comparten datos de manera automática. La clave para integrar los datos es colocar a alguien a cargo de las fuentes de datos de la compañía y el administrador de datos (DA) cubre este puesto.

La función principal del DA consiste en proporcionar el control centralizado sobre los datos de toda la organización, establece los estándares de definición de datos para garantizar que todas las aplicaciones utilicen formatos consistentes y convenciones de asignación de nombres, coordina las

aplicaciones y los equipos para garantizar que los datos proyectos individuales puedan integrarse en un sistema de información a nivel corporativo.

Tareas del administrador

Las tareas que se desempeñan como administrador de base de datos se numeran las siguientes:

- Instalar el servidor de base de datos
- Configurar el servidor de base de datos
- Crear las bases de datos
- Administración de usuarios
- Proporcionar recursos a los usuarios
- Realizar respaldos de las bases de datos
- Llevar a cabo monitoreo
- Afinar el servidor
- Implementar seguridad

Entre las tareas que desempeña el administrador de base de datos esta la de calcular el tamaño de la base de datos y teóricamente se hacer de la siguiente forma:

1. Buscar todas aquellas tablas de la base de datos, que impactan en el tamaño de la base de datos, es decir, todas aquellas tablas que no son catálogo.
2. Verificar el tamaño por registro de esas tablas que no son catálogo.
3. Calcular aproximadamente cuantos registros por tabla se incluirán en el arranque del sistema.
4. Multiplicar, la cantidad de registros aproximados que se incluirán en una determinada tabla, por el tamaño del registro.
5. Sumar todos los resultados.

6. Por ajustes, del total anterior sacar el 30% y sumarlo al resultado total anteriormente obtenido.
7. Pensando en los índices, habrá que obtener del total anterior un 20% y sumarlo al total anteriormente obtenido.
8. Identificar más o menos cuanto crece el sistema y cada cuando.
9. Hacer los mismos pasos del 1 al 5 y después sumarlo al último resultado obtenido.
10. Hacer un aproximado de 5 años y finalmente obtener el 30% del total anterior y sumarlo al total, esto para el Log de transacciones, en caso de que la base de datos sea altamente transaccional.

Personal con el que interactúa el DBA

- **Con usuarios en general:** para especificar los requisitos que debe cumplir la BD, estudiar las necesidades y la mejor manera de satisfacerlas.
- **Con directivos:** que fijarán los objetivos de la BD para que ésta responda a los objetivos generales de la empresa.
- **Con analistas y programadores,** a los que el DBA deberá proporcionar información acerca de la estructura de los datos, especificaciones para las aplicaciones que vayan a desarrollarse, etc.
- **Con operadores,** con ellos, el DBA deberá analizar los procedimientos de recuperación a aplicar en la explotación de la BD ante fallos del sistema, normas para copias de seguridad, arranque, etc.
- **Con suministradores,** para corregir problemas que puedan producirse en el sistema de base de datos, además de estar al corriente de nuevas herramientas, interfaces, programas, etc que estén desarrollándose y que sean de interés en el entorno de la BD.
- **Con personal de formación,** a fin de preparar los planes adecuados de formación sobre bases de datos a todos los niveles de la empresa.

Usuarios del sistema y control de acceso

Otra parte importante en la administración de la base de datos es el control de usuarios.

Los DBMS deben garantizar al menos tres niveles de acceso a usuarios, ya que garantiza la seguridad del servidor.

El primero nivel es un identificador para tener acceso al DBMS, el segundo a la base de datos y tercero a los objetos de la base.

Configuración del servidor

Una vez puesto en marcha el DBMS se pueden configurar ciertos parámetros para un mejor rendimiento de nuestra DBMS y las bases de datos, pero habrá que tomar en cuenta que el cambio en algunos de los parámetros, requerirá que se reinicie el servidor, por lo que habrá que buscar el momento adecuado para hacerlo.

Administración de las cuentas de usuarios y grupos

Una importante tarea del administrador del sistema es configurar las cuentas de usuario proporcionando los recursos y servicios necesarios.

Las cuentas de usuario se conforman de un nombre de usuario y una contraseña principalmente.

En un sistema Linux la información de las cuentas de usuarios se almacenan en los siguientes archivos: /etc/passwd, /etc/shadow y /etc/grup.

Otra forma de administrar a los usuarios es por grupos, en donde cada grupo representa diferentes niveles y tipos de usuarios.

Administración de software

Todo el software en un sistema Linux está dividido en paquetes RPM los cuales pueden ser instalados, actualizados o eliminados.

RPM mantiene una base de datos de los paquetes instalados y de sus archivos.

RPM tiene cinco modos de operación básicos: instalación, desinstalación, actualización, consulta y verificación.

Instalación de software utilizando código fuente

Este es el método universal para todas las distribuciones de Linux.

Es necesario que lea la documentación que acompaña a dicho paquete y seguir las instrucciones proporcionadas por el autor. Por lo general son necesarios al menos tres pasos:

```
./configure --prefix=/usr
```

```
make
```

```
make install
```

Respaldos

Creación y recuperación de respaldos

Podemos crear y recuperar respaldos utilizando las herramientas gzip, zip, gunzip y tar.

Diagnostico del sistema.

Podemos hacer uso de una gran gama de herramientas que vienen incluidas en la distribución de Linux para poder diagnosticar cuales son los problema que se presenta en el servidor.

Monitoreo de procesos

Podemos hacer uso de los siguientes comandos para ver las tareas que se estan ejecutando en nuestro sistema y la cantidad de recursos que ocupa cada uno de ellos.

ps Sirve para listar los procesos que están corriendo en el servidor.

```
ps -fea
```

```
ps -ue usuario
```

```
ps
```

Espacio en disco

df Sirve para determinar cuando espacio estamos utilizando en todas las particiones.

```
df -h
```

```
df -k
```

du Sirve para determinar cuando espacio tiene un archivo o directorio.

```
du -sh archivo
```

```
du -sk archivo
```

Monitoreo de la red

Ifconfig permite configurar las interfaces de red y mostrar información sobre ellas.

nmap Realiza un mapeo de puertos a cualquier servidor, para determinar que sistema operativo esta utilizando, o que puertos tiene abiertos.

ping Permite enviar paquetes de información a través de la red para determinar si se tiene acceso a una maquina remota.

traceroute Permite trazar una ruta de hosts por donde pasa un paquete para llegar al host destino seleccionado.

Bitácoras del sistema

Como administrador de una máquina Linux, revisa los archivos de registro; sirven para detectar posibles fallos de seguridad en nuestra máquina.

Linux. Los archivos de registro se encuentran en /var/log:

- messages: mensajes de seguridad y autenticación.
- cron: mensajes generados por el demonio cron.
- secure: mensajes de seguridad de acceso (conexiones, wrappers.).
- access_.log: mensajes de acceso al servidor WWW, si lo hubiese.
- Error_log: mensajes de error durante el acceso al servidor WWW, si lo hubiese.
- utmp: contiene información sobre los usuarios que se encuentran. Actualmente en el sistema. La información la consultamos con comandos como who, w, users.
- wtmp: contiene información sobre las entradas y salidas de los usuarios al sistema. La información la consultamos con el comando last.

1.9 SEGURIDAD EN LAS BASES DE DATOS

Debido a la importancia que tienen los datos para la subsistencia de las organizaciones, es por tal motivo, que sin negar el compartir los datos entre los diferentes usuarios, se requiere de reglas, procedimientos y herramientas para la protección y uso de los datos. Es por eso que en este modulo nos proporcionan los elementos para identificar aquello que se quiera proteger, identificando cuales son los posibles hoyos con los que podemos contar y una vez identificados estos, como los podemos solucionar.

Seguridad

Seguridad es un estado, donde se esta libre de riesgos. Sin embargo, la seguridad informática lo que nos brinda son mecanismos para disminuir los riesgos tanto de hardware, software y datos.

Rutina de implementación de seguridad

La forma de comenzar a implementar seguridad es la siguiente:

Análisis de riesgos. Identificando las amenazas preguntándonos ¿de que me tengo que proteger?

Tanto en:

- hardware (incendios, inundaciones, temblores, etc)
- software (virus, troyanos, gusanos, etc)
- datos
- humanos (hacker, crackers, lamers, etc)

Identificando elementos críticos del sistema. Seria aquello de nuestro sistema que queremos proteger como:

- Base de datos
- Programas
- Red
- Equipos
- Personas clave

Definición de mecanismos de seguridad. Identificando con que herramientas se va a proteger según nuestros elementos críticos, en principio se tendrían que definir:

- Políticas de seguridad
- Procedimientos de seguridad, definiendo como es que se deben de hacer las cosas
- Mecanismos técnicos como firewalls

Mantener canales de información como son:

- Páginas Web
- Listas de correo tanto de las aplicaciones y del sistema operativo

Proporcionar solo los servicios necesarios:

- Segmentando los servicios
- Determinar los puertos abiertos

Proteger los puertos que tengan que estar abiertos con:

- Mecanismos que ofrezca el propio sistema

CAPÍTULO II ANÁLISIS Y DESARROLLO

II.1 FUNCIONES DE LA COORDINACIÓN DE DESARROLLO INSTITUCIONAL

Las necesidades constantes de desarrollo y mejoramiento de las actividades administrativas que realiza la empresa obligan a métodos automatizados de manejo de información para un mejor desempeño de funciones.

Funciones de la empresa

La empresa TecnoGens esta dedicada a la capacitación en tecnologías de información, ofreciendo la oportunidad a alumnos de distintas instituciones y áreas de conocimiento que puedan capacitarse de manera presencial por medio de cursos por ello es la encargada de realizar apertura de grupos, formulación de horarios, asignación de profesores así como del material necesario para cada curso, además de eso la empresa brinda servicios de instalación y mantenimiento de redes y desarrollo de sistemas.

Actualmente cuenta con más de una base de datos ya que una de ellas lleva el control del sistema escolar y en otra a lo relacionado a los materiales (tutoriales, manuales, videos, etc.) que son apoyo para impartir los cursos que ofrecen en estos momentos, además cuenta con otras bases de datos para el desarrollo de sistemas y el mantenimiento de redes, pero esta herramienta debe cumplir con el objetivo de administrar las bases de datos situadas en el manejador de PostgreSQL.

El departamento de cómputo debe cumplir con las siguientes funciones:

- Generar, mantener y administrar las bases de datos.
- Coordinar el Proceso de auto evaluación de la empresa
- Asesorar y brindar apoyo al área de control escolar para la acreditación de los cursos.
- Organizar actividades y coordinar acciones tendientes a aumentar la capacidad y el compromiso del conjunto a fin de garantizar los propósitos de la empresa.
- Promover mejoramiento del nivel pedagógico y didáctico de enseñanza supervisando el análisis de los problemas.
- Brindar servicio y asesoramiento

- Organizar los cursos de capacitación.

II.2 PLANTEAMIENTO DEL PROBLEMA

La empresa TecnoGens realiza tareas repetitivas para la creación de usuarios de base de datos y de grupos de trabajo, tareas como asignar privilegios, realizar consultas, reportes, entre otros, que en ocasiones se requiere del DBA que es el encargado de la administración de las bases de datos y éste va solo una vez por semana o cuando se requiera ya que no hay recursos suficientes para un contrato de base. Los operarios del sistema que son los que tienen ciertos privilegios en el, se les hacen tedioso realizar consultas en la interfaz básica del manejador Postgresql además de que muchas operaciones que realizan (consultas) se hacen de manera repetitiva varias veces al día y consultas que en ocasiones deben esperar a que el DBA vaya sin saber que son consultas sencillas, en este momento se han cometido errores al eliminar objetos de la base de datos, y lo gerentes y encargados de la coordinación que requieren conocer la plataforma les cuesta trabajo conocer con que usuarios y privilegios cuentan.

II.3 REQUERIMIENTOS DEL SISTEMA

- *Instalar el software requerido para el mejor funcionamiento de la aplicación*

Software

- Se requiere software libre **CentOS** (Acronym de Community Enterprise Operating System). El cual es un clon a nivel binario de la distribución Red Hat Enterprise Linux (Ver Anexo página 123).
- Base de datos instalados en el sistema de gestión de bases de datos PostgreSQL

Hardware

- Unidad Central de Proceso CPU

i386		
<ul style="list-style-type: none"> • Intel i486 • Celeron, Pentium, Pentium Pro, Pentium II y Pentium IV • AMD Athlon, Athlon XP y MP 		
s390	ia64	alpha
<ul style="list-style-type: none"> • IBM S/390 	<ul style="list-style-type: none"> • Intel Itanium 2 	DEC Alpha
s390	x86_64	Memoria RAM sin el ambiente gráfico. 512 Mb como mínimo
<ul style="list-style-type: none"> • Servidores IBM Z-Series 	<ul style="list-style-type: none"> • AMD Athlon 64, Opteron y Xeon EM64T 	

Tabla 2. Espacio mínimo requerido para CPU

- Disco Duro

Tipo de Instalación	Espacio mínimo requerido
Personal Desktop	2.3 Gb
Workstation	3.0 Gb
Server	1.1 a 1.5 Gb
Custom	0.62 a 6.9 Gb

Tabla 3. Espacio mínimo requerido para la instalación de Linux

- **Crear la conexión a un servidor web**
 - La configuración principal de Apache se realiza en el archivo *httpd.conf*, el cual CentOS pone en la ruta */etc/httpd/conf/*
 - En estos tipos de archivos de configuración, todas las líneas que comienzan con el signo # corresponden a comentarios dentro del archivo.

Este archivo contiene varios parámetros e indica todos los módulos cargados, pero ya que ocuparemos en general los parámetros de default solo debemos revisar el *DocumentRoot* que es el directorio que define en donde quedarán alojadas las páginas en el servidor Linux. En este caso CentOS las ubica en ***/var/www/html***

Una forma de verificar que Apache y PostgreSQL se inicializan con el sistema operativo, es revisando el archivo *init.d* que se encuentra en ***/etc/*** En este archivo están listados todos los servicios que arrancan con el sistema, por lo cual debe encontrarse listado *httpd* (Apache) y *postgresql*. Para revisar este archivo se puede hacer desde el editor preferido, por ejemplo vi, emacs o gedit (los dos últimos para ambientes gráficos).

- ***Crear cuentas de usuario y asignar el perfil.***

Se requiere un grupo de trabajo de usuarios de la base de datos TECNOSAE en el servidor Linux

Es necesario trabajar con las cuentas de los usuarios del sistema. Y lo primero que hay que hacer es crear un grupo de trabajo para identificarlo de manera sencilla como el grupo de trabajo de usuarios que tienen acceso a la herramienta web

- **Cuentas de usuario**

El siguiente paso es la creación de una cuenta como usuarios del servidor que es igual al nombre de usuario y contraseña que utilicen al encender sus equipos y entrar al sistema operativo. Es muy importante establecer ciertas políticas de seguridad al asignarle a cada usuario una cuenta de acceso al servidor y una contraseña. En la medida que los accesos a éste estén protegidos será la medida de protección de nuestros datos.

- **Crear usuario en la base de datos**

Todos los usuarios tienen permiso para conectarse al servidor y leer y escribir en la base de datos *TECNODB*, deberá crearles una cuenta en la base de datos *TECNODB* que puede ser idéntica a la que les creo para el sistema operativo, según sea su conveniencia. Generalmente es recomendable no obligar a los usuarios a que se aprendan muchas contraseñas.

Para dar una cuenta de acceso a la herramienta web hay que entrar al sistema con cuenta de administrador o con permisos para crear usuarios le damos clic al botón de usuario y asignamos el nombre de usuario (login) y password el cual deberá tener como mínimo 8 caracteres y deberá ser asignado a un grupo de trabajo, desde el momento que se asigna a un grupo, el usuario con dicha cuenta podrá crear otros usuarios y bases de datos, siempre que así lo indique el tipo de cuenta (filtro). Esta plataforma nos mostrará los datos que hemos agregado y podremos agregar un nuevo registro con el login.

Es de considerarse el campo Filtro en donde deberá indicar con uno o dos números el área donde se encuentra el usuario para efectos de las opciones que podrá ver y modificar.

Entre los usuarios se encuentran

- Administradores (Administrador)
- Analistas y programadores (operador)
- Esporádicos (usuario)

Registre el filtro tomando en cuenta la tabla No. 4:

Perfil	Tipo de usuario	Permisos
1	Administrador Total (super usuario)	Tiene permiso de acceso a todas las opciones del sistema y a la alimentación de catálogos
2	Operador	Tiene permisos para consultas y permiso sobre algunos objetos del sistema
3	Usuario	Tiene permiso de consulta
4	Administrador del sistema	Tiene permiso de acceso a todas las opciones del sistema

Tabla 4. Filtros utilizados en TecnoGens para el acceso a la información

- **Creación de Objetos y manipulación de los mismos**
 - Se requiere la creación de una tabla llamada perfil en la cual se almacenará la información de *perfil de acceso* y la descripción de cada uno de ellos como lo muestra la tabla 4 , además se creará una tabla llamada *usuariosweb* en donde tendrá los datos personales de los usuarios, la cual contendrá la referencia del perfil de acceso.
- **Particiones**

	Espacio mínimo requerido	Espacio Recomendado
/boot	100Mb	150Mb
/	500 Mb	2048Mb
/usr	Mínimo 1.7	4048 Mb
/var	500 Mb	2048 Mb
/tmp	500 Mb	1.5 la cantidad de la RAM
/home	Tan grande como sea posible	
swap	2xRAM	2xRAM

Tabla 5 de espacio para instalar Linux en particiones

II.4 PLANTEAMIENTO DE LA SOLUCIÓN

El sistema propuesto es una herramienta Web que se acopla a las necesidades de la empresa por lo tanto cambia significativamente la manera en que es entregada la información a la gente que labora actualmente en ese lugar. El objetivo principal es satisfacer los requerimientos de información internos de la empresa para una mejor gestión, con eficiencia y facilidad de acceso.

Esta herramienta brinda la pauta para la gestión y control de las bases de cursos y eventos, programas académicos, localizador de información, materiales educativos y noticias que conforman

la empresa TecnoGens. Este sistema nos permite controlar el ingreso de datos en forma sencilla dividiendo el entorno de trabajo en procesamiento y producción, así mismo permite el control de base de datos de tablas como de los usuarios y con la posibilidad de exportar los datos para tener respaldo de la información.

Además de suministrar al usuario las herramientas que le permitan manipular, en términos abstractos, los datos, o sea, de forma que no le sea necesario conocer el modo de almacenamiento de los datos en la computadora, ni el método de acceso empleado.

Para administrar la base de datos, se utiliza la interfaz Web TecnoGens PostgreSQL Admin, que es bastante potente, amigable y una de las más extendidas. TecnoGens PostgreSQL Admin es un conjunto de programas desarrollados en PHP. Es una herramienta muy completa que permite acceder a todas las funciones típicas de la base de datos PostgreSQL a través de una interfaz Web muy intuitiva. Y para que ésta funcione, debemos tener configurado el soporte PHP e instalado PostgreSQL.

Actualmente, la práctica totalidad de las instalaciones se están realizando con SGBD relacionales, instalados bien en modo centralizado, bien en modo cliente/servidor y, en pocas ocasiones, arquitecturas distribuidas. Esta es sin duda la solución más conveniente casi siempre y la que proporcionará un mayor número de posibilidades de elección.

La característica propia del sistema es que aparte de la administración de base de datos también se sugiere que realice la administración que catálogos de la dependencia.

Análisis del sistema con diagrama de Casos de Uso en UML

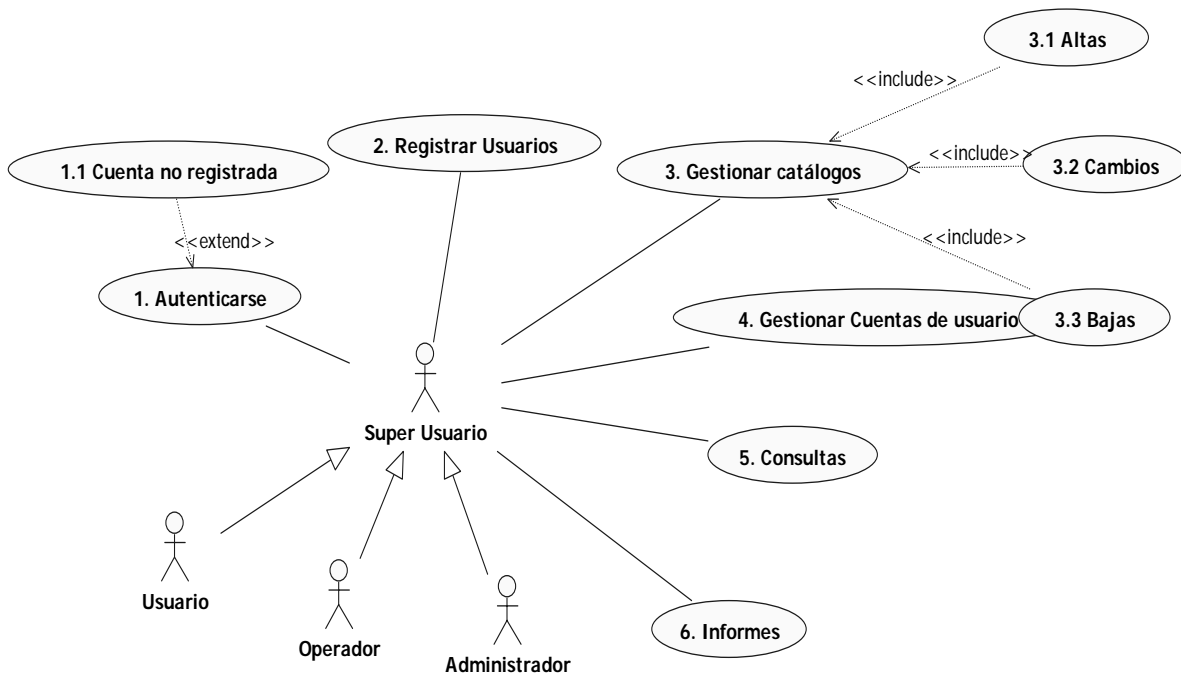


Figura. 25 Esquema general del sistema PostgreSQL Admin

Escenarios del sistema

Es de mucha importancia crear escenarios múltiples para satisfacer circunstancias excepcionales y caminos de proceso alternativos por lo que a continuación se muestran algunos ejemplos de escenarios de mayor importancia en el sistema:

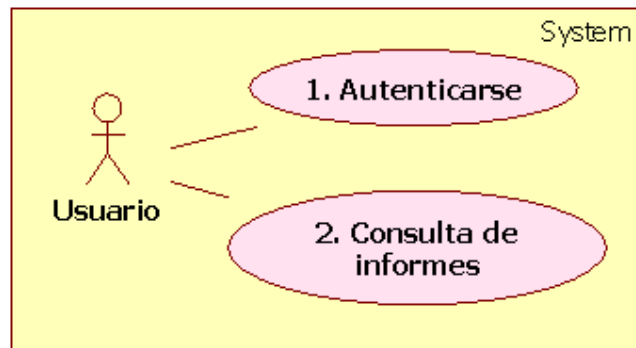


Figura 26. Escenario Usuario

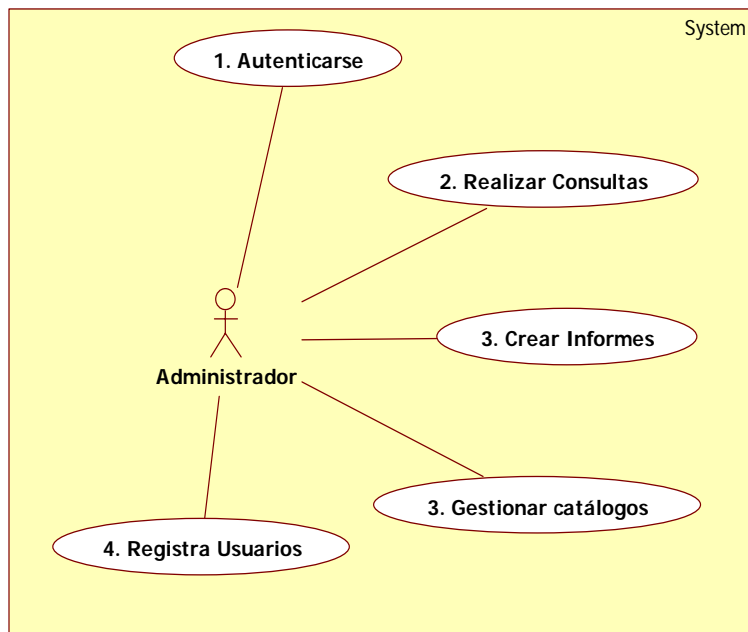


Figura 27. Escenario Administrador

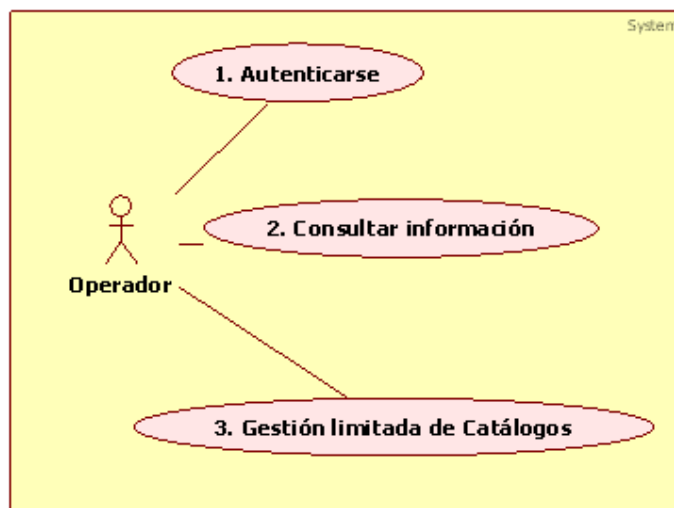


Figura 28. Escenario Operador

A continuación se muestran las especificaciones de casos de uso para el sistema POSTGRESQL ADMIN:

CASO DE USO	Autenticación.				
Id. del CU	1	Prioridad	Alta / Media / Baja	Estado	elaboración/Propuesto/Validado
Actores participantes	Todos los actores tendrán que autenticarse para poder hacer uso del sistema.				
Breve Descripción	Describe la forma que deben acceder los usuarios al sistema.				
Pre-condiciones	El usuario tendrá que estar previamente registrado.				
Flujo Principal	<p>Flujo principal</p> <ul style="list-style-type: none"> ○ Inicio El usuario ingresa al sitio ○ Ingresar datos El usuario ingresara su login y su password los cuales están 				

	<p>conformados de la siguiente manera:</p> <p><i>Login:</i> puede ser el nombre del usuario o una palabra que esté compuesta por letras y números.</p> <p><i>Contraseña:</i> debe estar compuesto con letras, números y símbolos.</p> <ul style="list-style-type: none"> o Validar datos El sistema comparara con los registro en la base de datos si el usuario esta registrado (<i>Excepción I</i>), entonces el sistema tomara la clave del rol que va desempeñar durante su acceso. o Acceso al sistema El sistema le dará acceso solo aquellos módulos que tiene permisos.
Flujos Alternos	No hay en este caso de uso.
Flujos de excepción	<p>Flujo Excepción I: El usuario no se registro</p> <p>Si el usuario no está registrado, el sistema mostrará un mensaje de error. El usuario deberá notificar al Súper usuario para que le registre en el sistema.</p>
Post-condiciones	Los registros de la base de dato se verán afectados conforme al rol que desempeña el usuario.

CASO DE USO	Registro de usuarios				
Id. del CU	2	Prioridad	<u>Alta</u> / Media / Baja	Estado	<u>elaboración</u> /Propuesto/Validado
Actores participantes	<p>Súper usuario: es la persona que goza de todos los privilegios en el sistema, así como se hace cargo de la administración del mismo y el único que puede dar de alta una cuenta.</p> <p>Usuario, es la persona que podrá introducir datos básicos al sistema para iniciar su registro al mismo.</p>				
Breve Descripción	Describe la manera de cómo se registra a un usuario en la base de datos para tener acceso posteriormente al sistema.				
Pre-condiciones	Que envíen el oficio con la copia del correo con login y password para activar la nueva cuenta.				
Flujo Principal	<ul style="list-style-type: none"> ○ Inicio El Súper usuario ingresará al módulo de registro. ○ Mostrar formulario El sistema desplegará un formulario donde se capturara los datos del nuevo usuario. Los datos que se proporcionará son los siguientes: <ul style="list-style-type: none"> • <i>Nombre:</i> de tipo alfanumérico, no mayor a 20 caracteres y será obligatorio. 				

	<ul style="list-style-type: none"> • <i>Apellidos</i>: de tipo alfanumérico, no mayor a 50 caracteres y será obligatorio. • <i>Departamento</i>: es el nombre del área a la que pertenece y será obligatorio. • <i>Correo electrónico</i>: de tipo alfanumérico, no mayor a 35 caracteres de los cuales no serán permitidos caracteres especiales, será seguido de un @ y el nombre del servidor y será obligatorio. • <i>Login</i>: de tipo alfanumérico y será de tipo obligatorio. • Contraseña: de tipo alfanumérico y será de tipo obligatorio. • Rol: de tipo numérico y será obligatorio, este campo definirá los privilegios que tendrá el usuario para el sistema. <ul style="list-style-type: none"> ○ Validar datos El sistema comparará con los registros en la base de datos si los datos cumplen con el estándar definido (<i>Excepción I</i>) y si el usuario ya existe en la base de datos (<i>Flujo Alternativo I</i>). ○ Envío de datos El sistema insertará los datos a la base de datos en sus respectivos campos y finalmente envía una respuesta en pantalla al Súper usuario. ○ Despliegue de respuesta El sistema confirmará al Súper usuario que la inserción de datos fue satisfactoria.
Flujos Alternos	<p>Flujo Alternativo I: El usuario ya existe</p> <p>Si el usuario no está registrado, el sistema le notificará al Súper usuario que ya existe el registro, el Súper usuario simplemente le cambiará la contraseña para que siga activa la cuenta, ir a envío de datos.</p>
Flujos de excepción	<p>Flujo Excepcional I: Los datos no son correctos</p> <p>El sistema notificará al Súper usuario que los datos que capturo no están dentro de los estándares y deberá corregir aquellos datos que no estén bien.</p>
Post-condiciones	<p>El usuario podrá autenticarse.</p>

CASO DE USO		Gestión de catálogos.			
Id. del CU	3	Prioridad	Alta / Media / Baja	Estado	elaboración/Propuesto/Validado
Actores participantes	Solo los siguientes usuarios con sus respectivos privilegios podrán gestionar los catálogos: <ul style="list-style-type: none"> • Súper Usuario • Administrador • Operador 				
Breve Descripción	Describe el manejo de las tablas de la base de datos, conforme a su inserción, actualización y eliminación de registros.				
Pre-condiciones	Tener acceso al sistema.				
Flujo Principal	<ul style="list-style-type: none"> ○ Alta Su funcionamiento se explica en el <i>caso de uso 3.1</i> ○ Actualizar Su funcionamiento se explica en el <i>caso de uso 3.2</i> ○ Baja Su funcionamiento se explica en el <i>caso de uso 3.3</i> 				

Flujos Alternos	No hay en este caso de uso.
Flujos de excepción	No hay en este caso de uso.
Post-condiciones	Los registros de la base de dato se verán afectados conforme al rol que desempeña el usuario.

CASO DE USO	Altas.				
Id. del CU	3.1	Prioridad	<u>Alta</u> / Media / Baja	Estado	<u>elaboración</u> /Propuesto/Validado
Actores participantes	Solo los siguientes usuarios con sus respectivos privilegios podrán gestionar los catálogos: Súper Usuario, Administrador, Operador				
Breve Descripción	Describe la inserción de datos en las distintas tablas de la base de datos.				
Pre-condiciones	Tener acceso al sistema y tener los privilegios necesarios para modificar.				

Flujo Principal	<ul style="list-style-type: none"> ○ Inicio El usuario selecciona la tabla a la que desea ingresar los datos. ○ Seleccionar numero de datos El usuario entra al módulo de insertar y selecciona el número de datos que desea insertar a la tabla previamente seleccionada. ○ Mostrar formulario El sistema muestra el formulario repetido con respecto al número de registros que el usuario desea ingresar, así también muestra el nombre y el tipo de los campos de la tabla. ○ Crear sentencia SQL El usuario capturar los datos en los campos de la tabla, cuando le de ejecutar el sistema generar la sentencia SQL para insertar los datos. ○ Fin El sistema valida los datos ingresados y dará respuesta al usuario en pantalla de que la inserción fue satisfactoria (<i>Excepción I</i>).
Flujos Alternos	No hay en este caso de uso
Flujos de excepción	<p>Flujo Excepción I : Datos inválidos</p> <p>El sistema notificará al usuario que hubo un error al insertar los datos ya que el tipo de los campos no corresponde al de los datos, el sistema permitirá capturar de nuevo los datos.</p>
Post-Condiciones	Los registros de la base de dato se verán afectados conforme al rol que desempeña el usuario.

CASO DE USO	Cambios.				
Id. del CU	3.2	Prioridad	Alta / Media / Baja	Estado	elaboración/Propuesto/Validado
Actores participantes	Solo los siguientes usuarios con sus respectivos privilegios podrán gestionar los catálogos: Súper Usuario, Administrador, Operador.				

Breve Descripción	Describe la forma de hacer cambios en los registros existentes de una tabla de la base de datos.
Pre-condiciones	Tener acceso al sistema y tener los privilegios necesarios para modificar.
Flujo Principal	<ul style="list-style-type: none"> ○ Inicio El usuario selecciona la tabla a la que desea ingresar los datos. ○ Seleccionar el registro El usuario entra al módulo de cambios y selecciona los registros que quiere cambiar. ○ Mostrar formulario El sistema muestra el formulario lleno con los datos de los campos de la tabla, de tal manera que el usuario podrá cambiar los datos. ○ Crear sentencia SQL El usuario enviara los datos y el sistema creara la sentencia SQL de actualización. ○ Fin El sistema valida los datos ingresados y dará respuesta al usuario en pantalla de la actualización que fue satisfactoria (<i>Excepción I</i>).
Flujos Alternos	No hay en este caso de uso.
Flujos de excepción	Flujo Excepción I: Datos inválidos El sistema notificará al usuario que hubo un error al actualizar los datos ya que el tipo de los campos no corresponde al de los datos, el sistema permitirá capturar de nuevo los datos.
Post-condiciones	Los registros de la base de dato se verán afectados conforme al rol que desempeña el usuario.

CASO DE USO	Bajas.				
Id. del CU	3.3	Prioridad	<u>Alta</u> / Media / Baja	Estado	<u>elaboración</u> /Propuesto/Validado
Actores participantes	Solo los siguientes usuarios con sus respectivos privilegios podrán gestionar los catálogos: <ul style="list-style-type: none"> • Súper Usuario • Administrador 				
Breve Descripción	Describe la forma de borrar registros de una tabla de la base de datos.				
Pre-condiciones	Tener acceso al sistema y tener los privilegios necesarios para modificar.				

<p>Flujo Principal</p>	<ul style="list-style-type: none"> ○ Inicio El usuario selecciona la tabla en la que desea borrar uno o varios registros. ○ Seleccionar registros El usuario entra al módulo de Borrar y selecciona el o los registros que desea borrar. El sistema le preguntara si desea borrarlos o cancela la operación (<i>Excepción I</i>). ○ Crear sentencia SQL Cuando el usuario confirma la operación el sistema genera la sentencia SQL para borrar. ○ Respuesta El sistema válido que las llaves primarias de los datos no estén ligadas con otras tablas (<i>Excepción II</i>) y finalmente le da una respuesta al usuario en pantalla de que la operación fue satisfactoria.
<p>Flujos Alternos</p>	<p>No hay en este caso de uso.</p>
<p>Flujos de excepción</p>	<p>Flujo Excepción I: Cancelar borrar registros El usuario al seleccionar cancelar la operación de borrar registro y no ejecuta ninguna sentencia de SQL y le permite al usuario seleccionar otro registro o realizar otra operación.</p> <p>Flujo Excepción II: La llave primaria está ligada a otra El sistema notifica al usuario que el registro que desea borrar afectara a otros registros ligados a la tabla y le mostrara a que tabla en específico afectaría, si desea borrar ese registro, primero deberá borrar los registros dependientes y posteriormente el registro que se desea borrar.</p>
<p>Post-condiciones</p>	<p>Los registros de la base de dato se verán afectados conforme al rol que desempeña el usuario.</p>

CASO DE USO	Gestión de cuentas				
Id. del CU	4	Prioridad	<u>Alta</u> / Media / Baja	Estado	elaboración/ <u>Propuesto</u> /Validado
Actores participantes	Súper Usuario				
Breve Descripción	Describe la creación de una cuenta y carpeta dentro del servidor.				
Pre-condiciones	Tener acceso al servidor.				

<p>Flujo Principal</p>	<ul style="list-style-type: none"> ○ Inicio Ingresar como usuario root al servidor ○ Crear cuenta El Súper usuario abrirá una ventana de consola y tecleara lo siguiente: # useradd nombre -d /home/nombre #passwd nombre ó creará un nuevo usuario desde la interfaz web ○ Validar datos El servidor comparara si hay algún usuario dado de alta con el mismo nombre (<i>Excepción I</i>). ○ Notificación El servidor notificará que la cuenta esta creada y se podrá corroborar si esta creada la carpeta en el directorio de /home, se le entregará la cuenta al usuario. 				
<p>Flujos Alternos</p>	<p>No hay en este caso de uso.</p>				
<p>Flujos de excepción</p>	<p>Flujo Excepción I: El usuario ya existe El servidor le desplegará al Súper usuario en pantalla que ya existe una cuenta con ese nombre, el Súper usuario tendrá que poner una variante a la cuenta para que se pueda crearse.</p>				
<p>Post-condiciones</p>	<p>Se enviara un oficio de la cuenta que se debe crear.</p>				
<p>CASO DE USO</p>	<p>Consultas</p>				
<p>Id. del CU</p>	<p>5</p>	<p>Prioridad</p>	<p>Alta / Media / Baja</p>	<p>Estado</p>	<p>elaboración/Propuesto/<u>Validado</u></p>
<p>Actores participantes</p>	<p>Todos los usuarios.</p>				
<p>Breve Descripción</p>	<p>Describe la creación de consultas a los catálogos.</p>				
<p>Pre-condiciones</p>	<p>Saber que información se requiere y haberse autenticado.</p>				

<p>Flujo Principal</p>	<ul style="list-style-type: none"> ○ Inicio Ingresar al modulo de consulta. ○ Crear una consulta El usuario declarara por medio de lenguaje SQL, lo datos que desea conocer de las distintas tablas que albergan los registros. ○ Validar consulta Se ejecuta la consulta y la base de datos interpreta la instrucción. ○ Muestra resultado. El sistema muestra los registros arroja la base de datos (<i>Excepción I</i>) (<i>Excepción II</i>).
<p>Flujos Alternos</p>	<p>No hay en este caso de uso.</p>
<p>Flujos de excepción</p>	<p>Flujo Excepción I: No hay registros. El sistema muestra un mensaje al usuario de que la consulta no arroja registros con los parámetros que especifico.</p>

CASO DE USO	Informes				
<p>Id. del CU</p>	<p>6</p>	<p>Prioridad</p>	<p>Alta / Media / Baja</p>	<p>Estado</p>	<p>elaboración/Propuesto/Validado</p>
<p>Actores participantes</p>	<p>Todos los usuarios.</p>				
<p>Breve Descripción</p>	<p>Describe la consulta de informes con respecto al acceso de los usuarios al sistema y sus acciones dentro del mismo.</p>				
<p>Pre-condiciones</p>	<p>Saber que información se requiere y haberse autenticado.</p>				

Flujo Principal	<ul style="list-style-type: none"> ○ Inicio Ingresar al modulo de informes. ○ Seleccionar tipo de informe El sistema le presentara una serie de opciones para que el usuario seleccione el tipo de informe que desea ver como: <ul style="list-style-type: none"> • Acceso de Usuarios • Insertar registros • Actualizar registros • Borrar registros • Ejecución de consultas. Además que podrá seleccionar algunos parámetros como: <ul style="list-style-type: none"> • Fecha • Hora • Usuario ○ Mostrar informe El sistema mostrará en un formato la selección, además de que paginara los resultados desde el más reciente hasta el más viejo (<i>Excepción I</i>).
Flujos Alternos	No hay en este caso de uso.
Flujos de excepción	<p>Flujo Excepción I: No hay registros.</p> <p>El sistema muestra un mensaje al del que la consulta para el informe con los parámetros que seleccionó no arroja resultados.</p>

Diagramas de Actividades

Un diagrama de actividades es fundamentalmente un diagrama de flujo que nos muestra el flujo de control entre las actividades a continuación se muestran varios diagramas de actividades que fueron el punto clave para la programación del sistema (figs. 29 - 38).

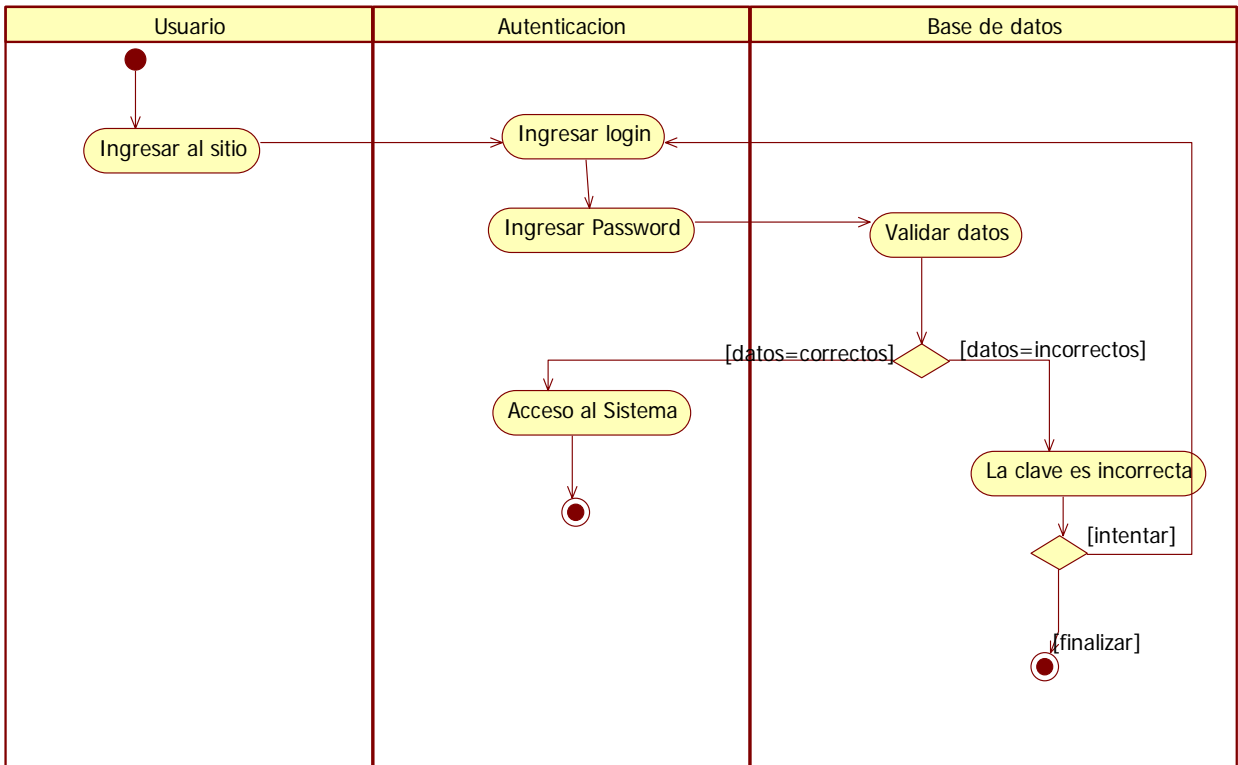


Figura 29 Diagrama de actividad, autenticar usuario

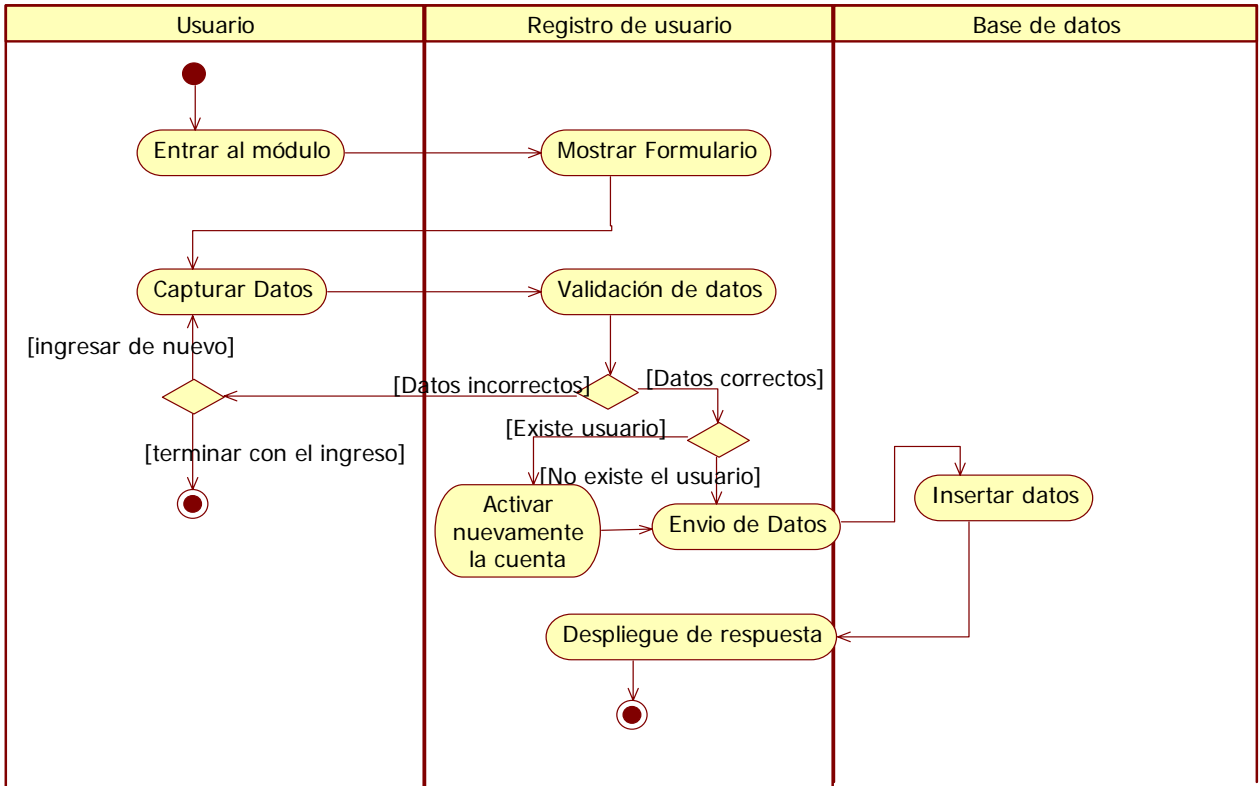


Figura 30. Diagrama de actividad, registrar usuario

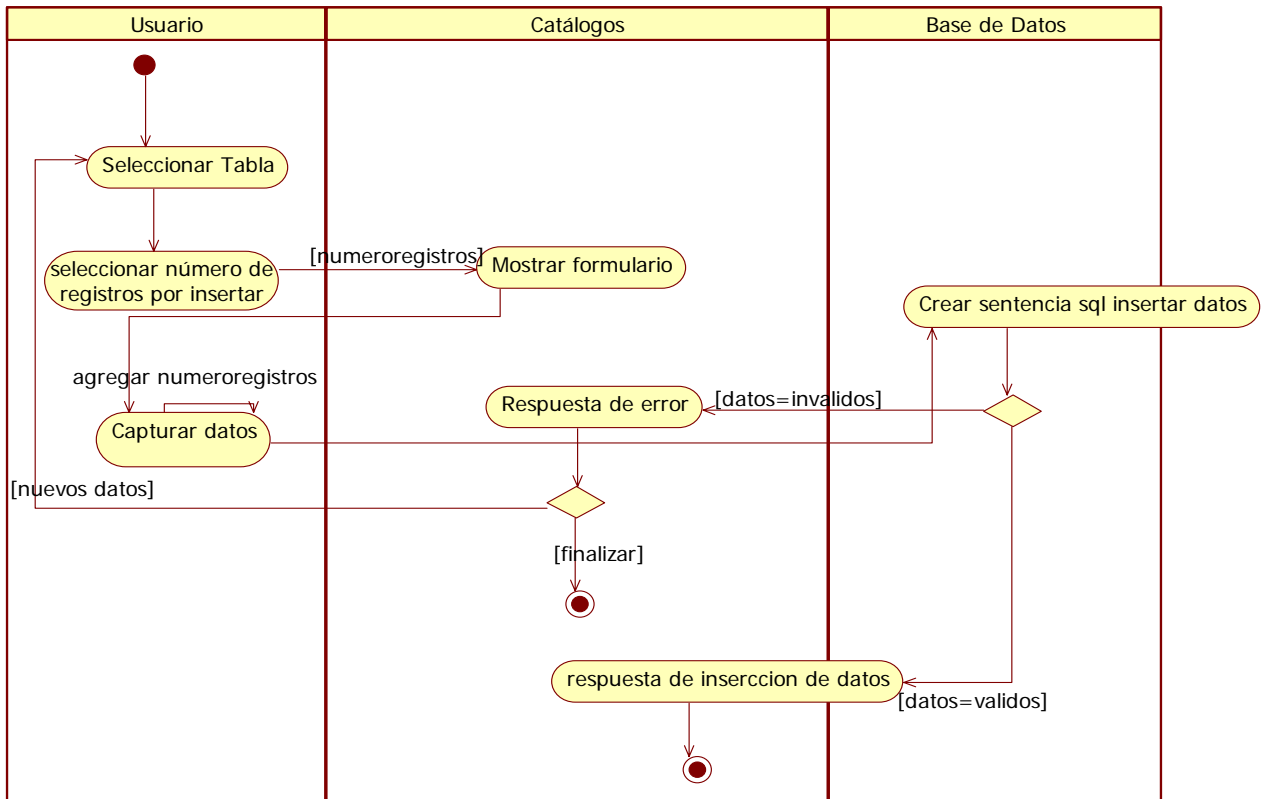


Figura 31. Diagrama de actividades, crear registro

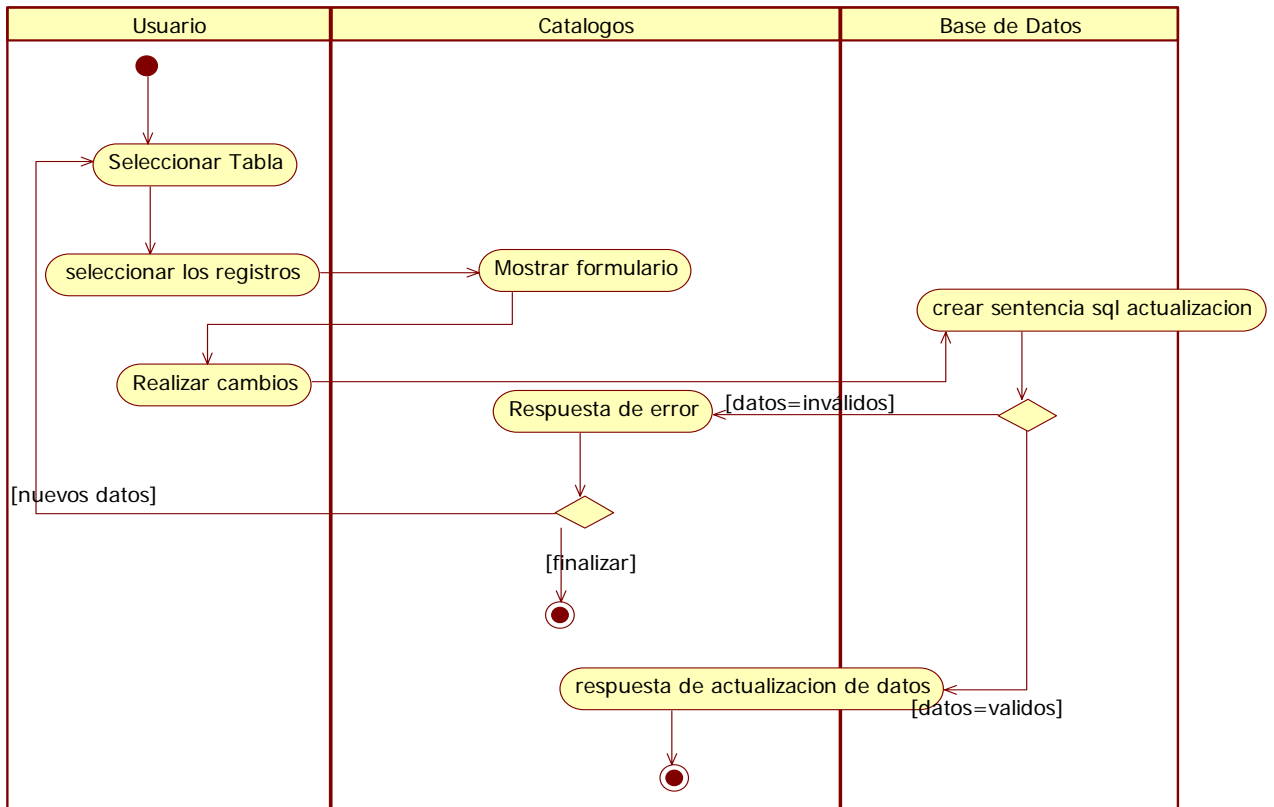


Figura 32. Diagrama de actividades, actualizar

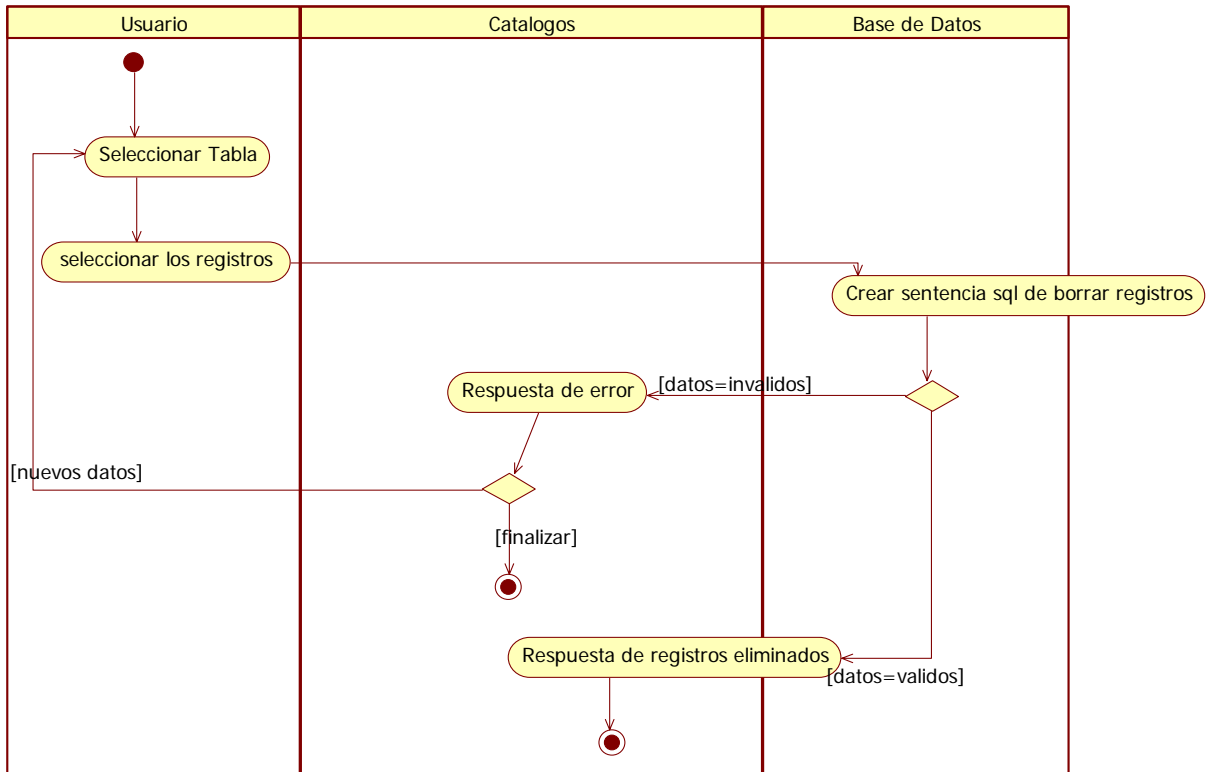


Figura 33 Diagrama de actividades, Borrar

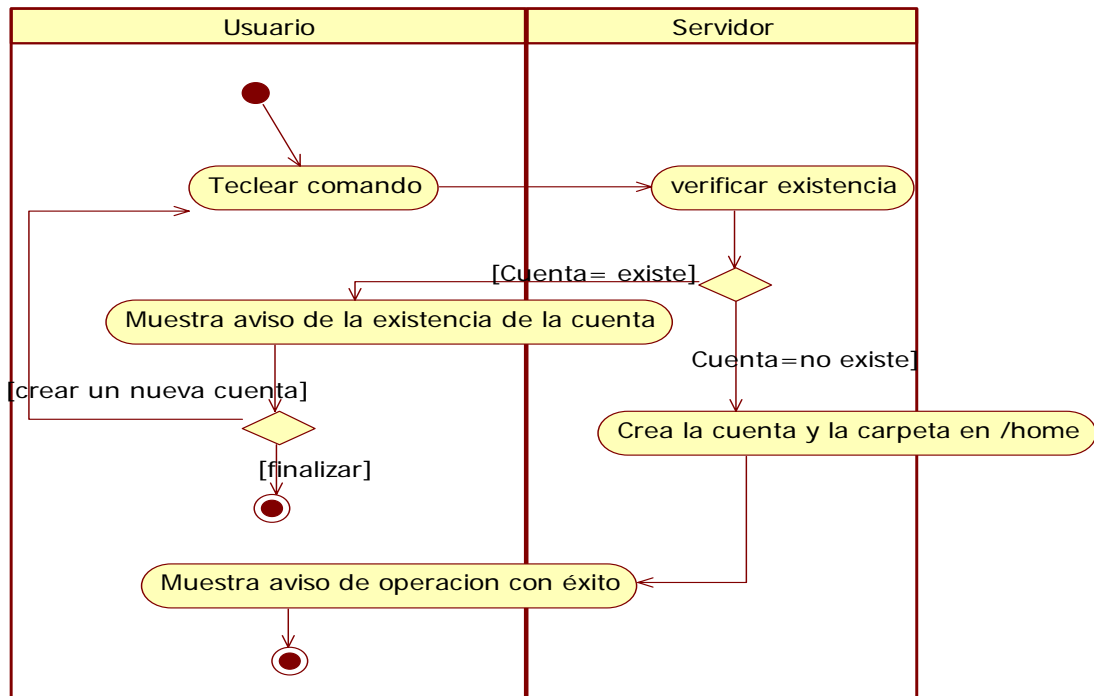


Figura 34. Diagrama de actividades, gestión de cuentas

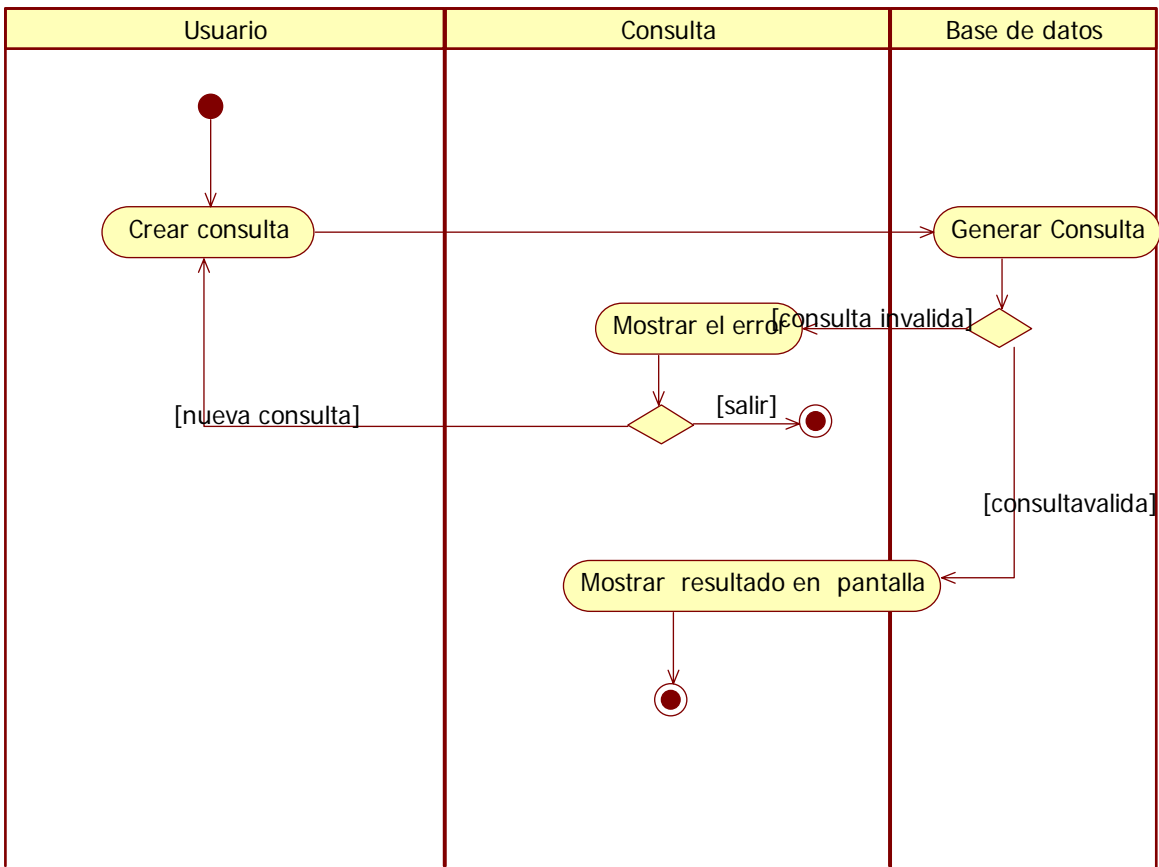


Figura 35. Diagrama de actividades, consultas

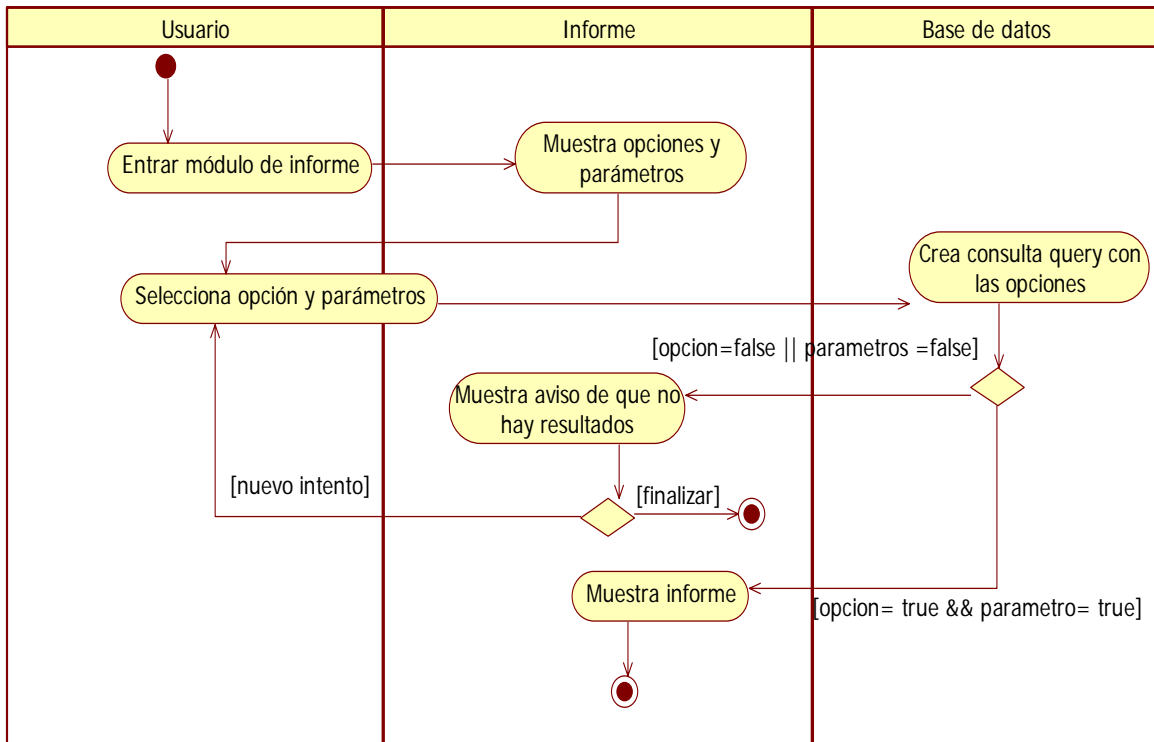


Figura 36. Diagrama de actividades, Informes

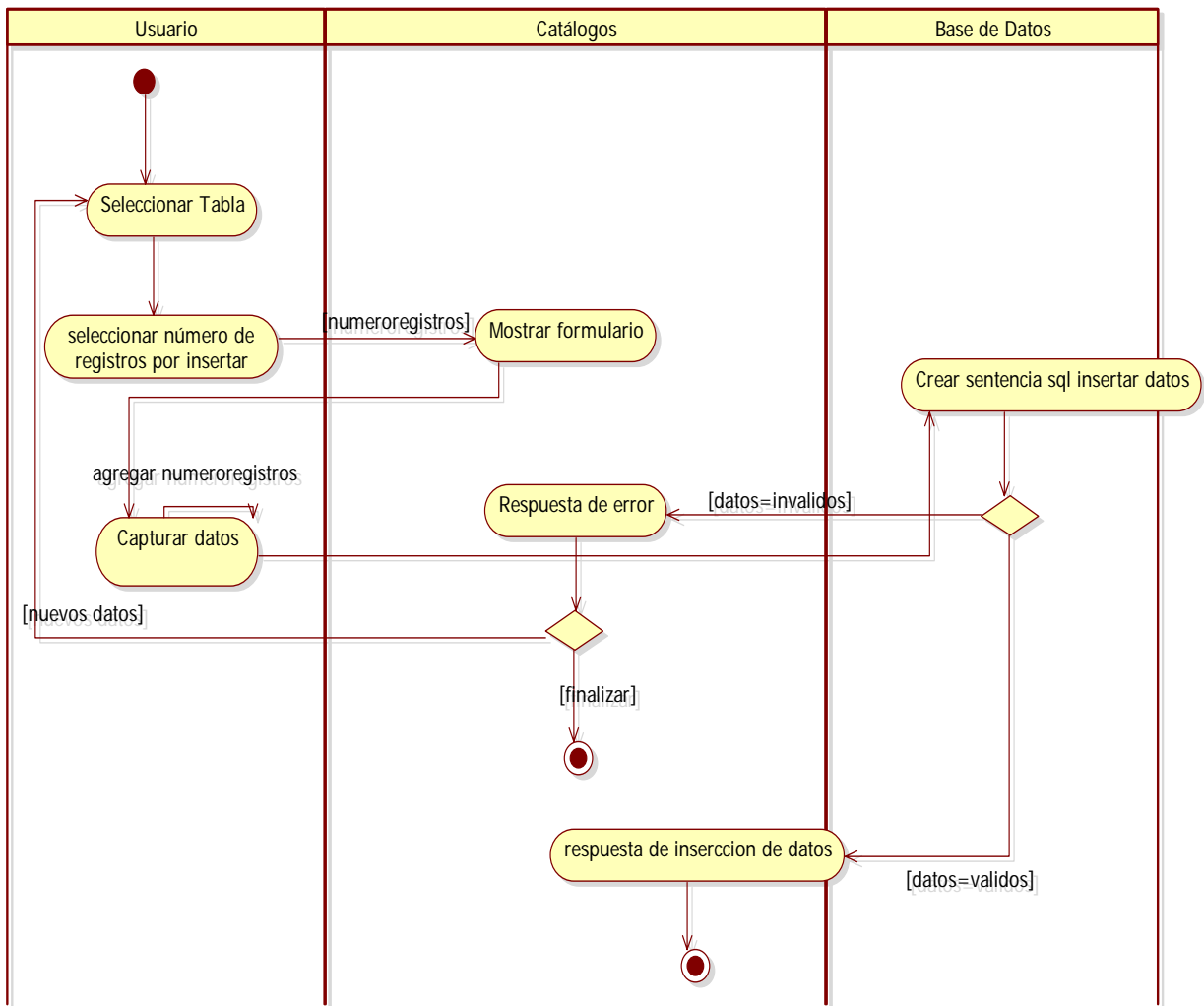


Figura 37. Diagrama de actividades, altas al sistema

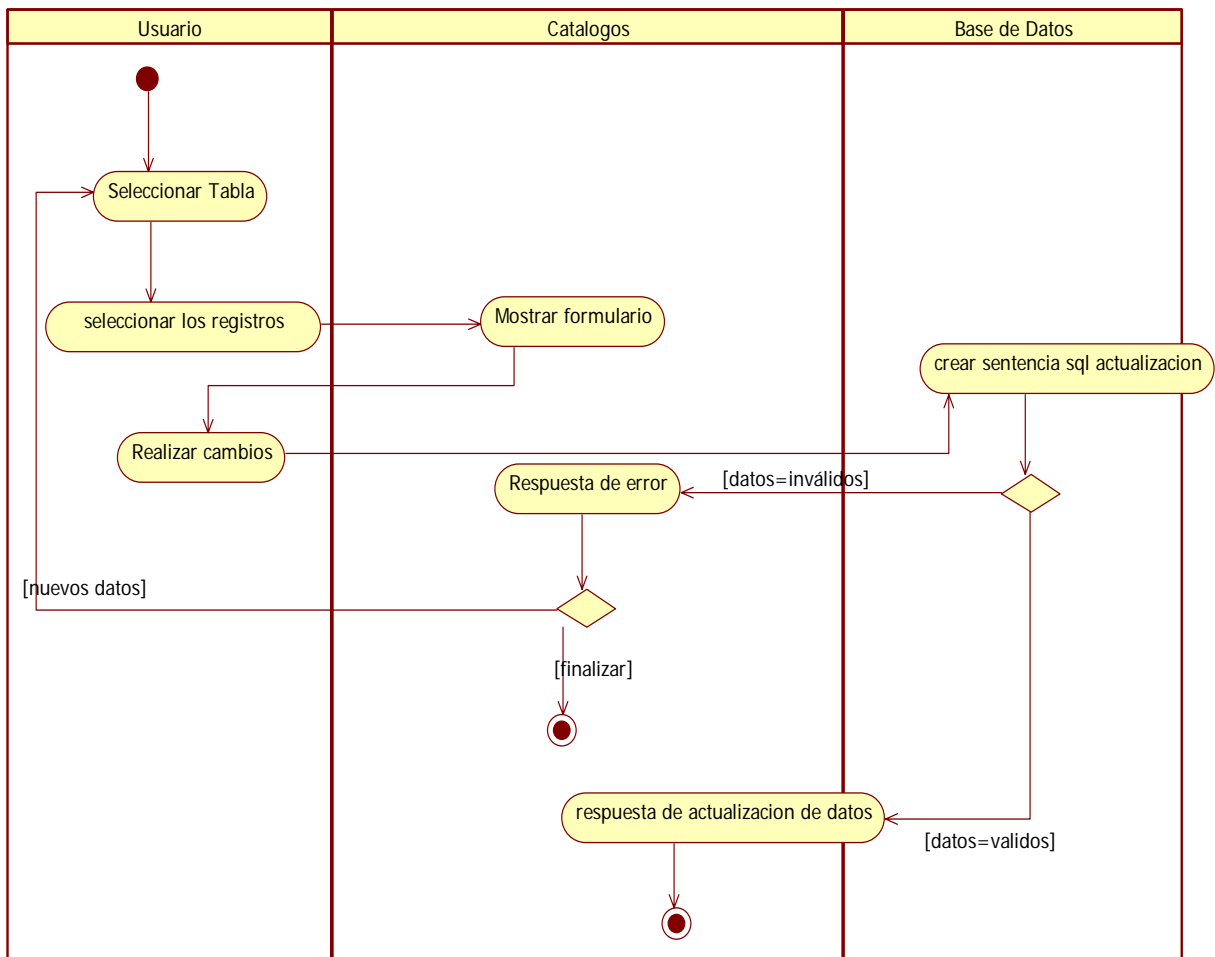


Figura 38. Diagrama de actividades, cambios

El diagrama de despliegue se utiliza para visualizar los aspectos estáticos de los nodos físicos y sus relaciones y para especificar sus detalles para la construcción (fig. 39).

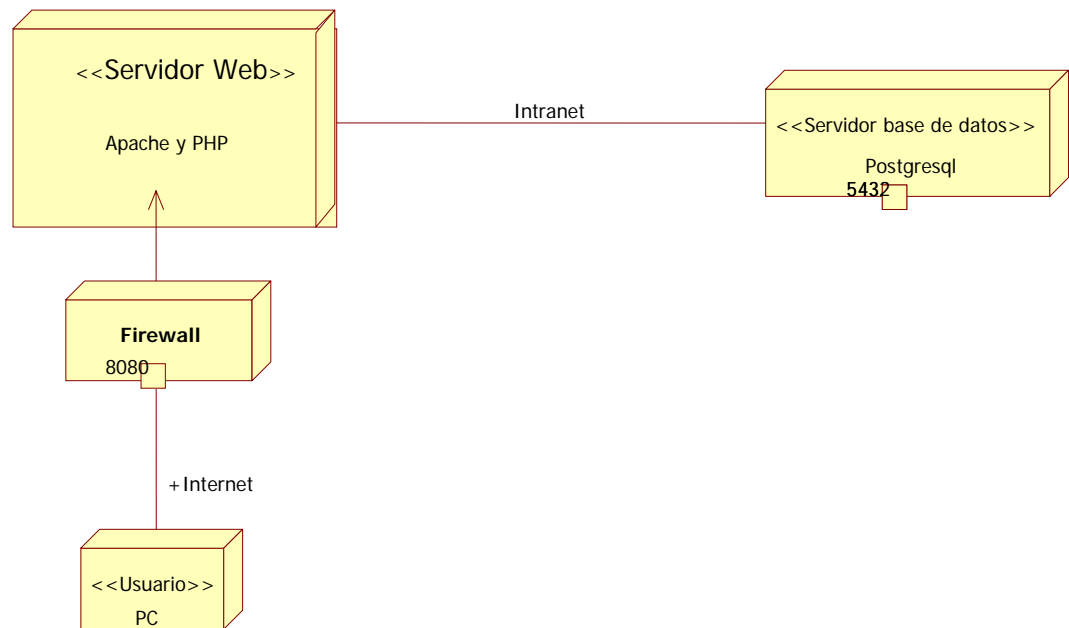


Figura 39. Diagrama de despliegue del Servidor

El diagrama de componentes nos sirve para modelar el código fuente y sus versiones ejecutables, es parecido al diagrama de despliegue ya que también se utilizan para visualizar los aspectos estáticos de los artefactos físicos y sus relaciones, el diagrama siguiente muestra a grandes rasgos las páginas de código del sistema pero cabe mencionar que no se especifican de todo el sistema solo una parte esto por seguridad de la empresa (fig.40).

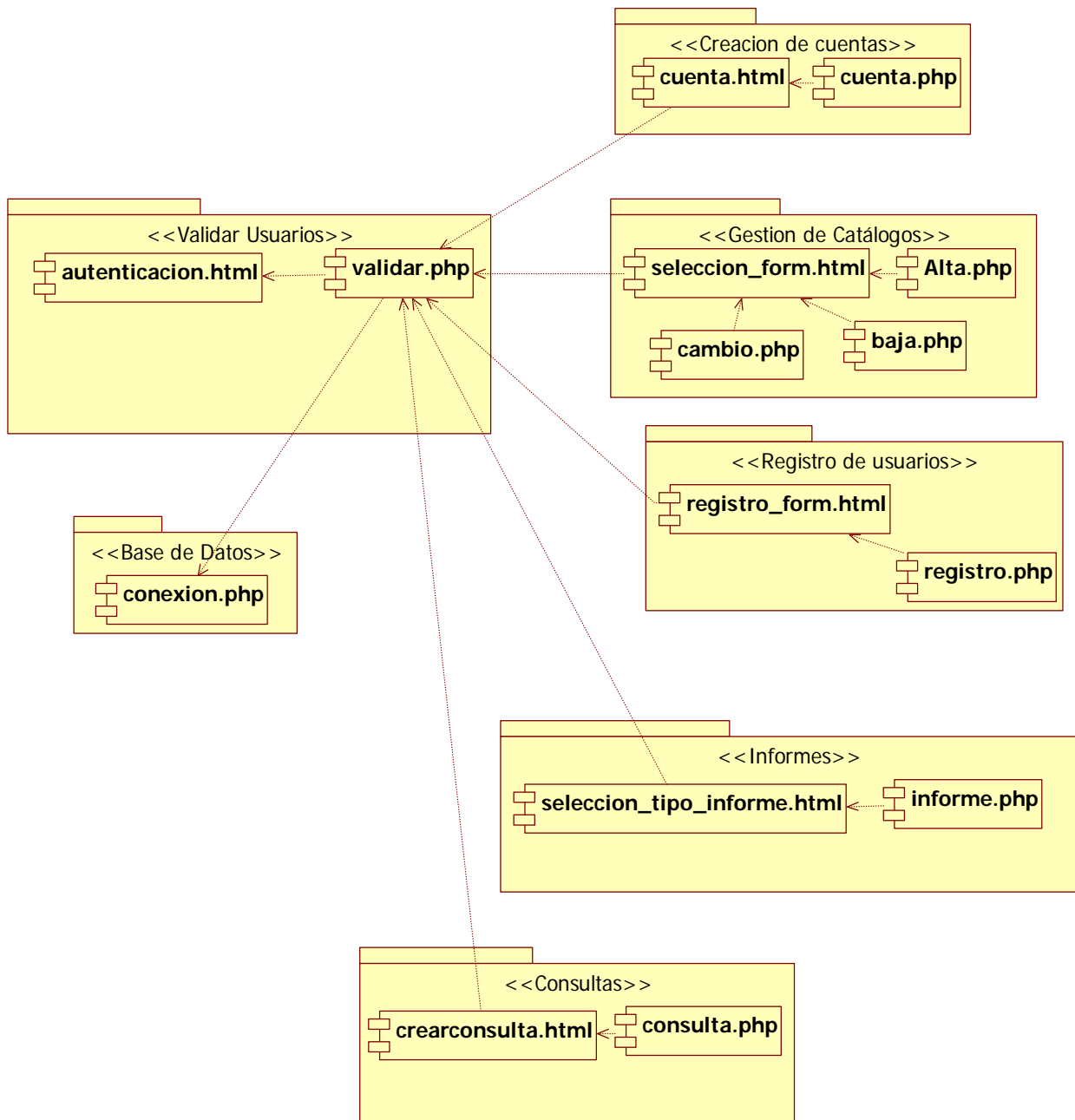


Figura 40. Diagrama de componentes del sistema

CAPÍTULO III. IMPLANTACIÓN Y PRUEBAS

III.1 PRESENTACIÓN AL USUARIO

Para ingresar al modulo de gestión de Bases de Datos TecnoGens PostgrtesSQL Admin (Fig. 41) se debe:

1. Digitar el nombre de usuario o ID que le ha sido asignada al usuario previamente registrado.
2. Digitar su password

3. Dar click en el botón



Figura 41. Ingreso a la herramienta de Administración PostgreSQL Admin

Si aún no tiene cuenta para ingresar al sistema entonces mandara y un mensaje como se muestra a continuación (fig.42):

**No esta autorizado para ver esta pagina,
para acceder a esta página es necesario un previo registro
y además una activación de cuenta por parte del Administrador.**

[Regresar](#)

Figura 42. Mensaje de error para usuario no autorizado a la herramienta PostgreSQL Admin

Por lo tanto se dará clic en registrarse (fig.43) y abrirá una pantalla como se muestra en la figura 44



INICIO DE SESIÓN

USUARIO

PASSWORD 

- ¿Olvidaste tu password?
- Registrarse

Figura. 43 Inicio de Sesión



Genes de la Tecnología 

REGISTRO

Clave de acceso minimo 8 y máximo 10 caracteres

Nombre

Apellido paterno

Apellido materno

Correo electrónico

Copyright ©2008 TecnoGenes.com.mx Inicio Bases de Datos Usuarios Gatos SQL Acerca de...

Figura 44. Registro rápido por medio de la herramienta de Administración PostgreSQL Admin

El sistema indicará los datos ingresados y pedirá que acepte o retorne al paso anterior con la opción corregir datos (fig. 45).

Figura 45. Comprobación de datos de registro en la herramienta de Administración PostgreSQL Admin

Al dar clic en el botón Enviar datos aparecerá una pantalla como se muestra a continuación con las indicaciones para continuar con el registro, en donde especifica Ha completado su registro al sistema. Que deberá esperar el correo electrónico con la clave y contraseña para continuar con el procedimiento de acceso al sistema TecnoGens PostgreSQL Admin, una vez recibida la cuenta el administrador podrá activarla el día que la institución así lo requiera (fig.46).



Figura 46. Registro completo por medio de la herramienta de Administración PostgreSQL Admin

III.2 ADMINISTRACIÓN DE BASE DE DATOS

Al ingresar a la herramienta se inicia una sesión para llevar mayor control sobre los objetos y usuarios, la primer pagina que encontramos en el sistema es la de administración de base de datos en la cual se puede crear y consultar las bases de datos existentes de la empresa TecnoGens teniendo en cuenta que de acuerdo al perfil se puede manipular la información, mostrando el propietario de la base de datos, su codificación (encoding), el tamaño, así como las opciones para ver sus tablas o eliminar la base de datos (Fig. 47).

Bases de Datos

Por favor indique el nombre de la nueva base de datos.

Figura 47. Crear base de datos en la herramienta PostgreSQL Admin

Si por error damos clic en crear sin haber escrito nombre a la nueva base de datos, entonces el sistema nos abrirá una ventana emergente como se muestra a continuación (fig. 48):

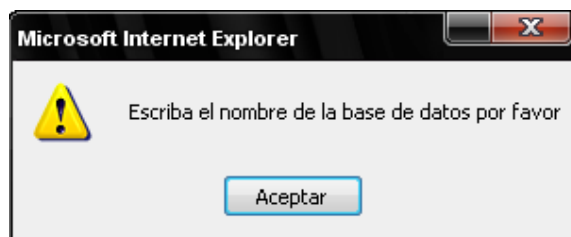


Figura 48. Crear base de datos en la herramienta PostgreSQL Admin

Al capturar el nombre de la base de datos, seleccionar el propietario y su tipo de codificación entonces podemos dar clic en crear y aparecerá un mensaje de aceptación que se ha creado nuestra base de datos (fig. 49).

Bases de Datos

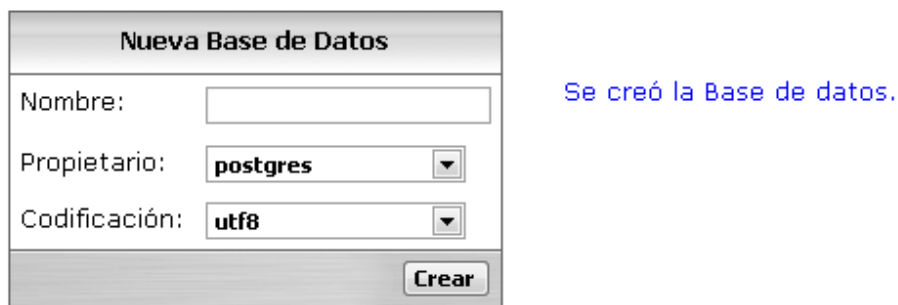


Figura 49. Mensaje de aceptación al crear una base de datos

El manejo de errores lo toma del manejador solo que el sistema lo pone de color rojo para hacer más llamativo el mensaje de error, si la base de datos que deseas crear ya existe entonces manda el siguiente mensaje de error (figs. 50,51).

Bases de Datos

ERROR: database "tecnopolisdb" already exists

Figura 50. Mensaje de error de base de datos duplicada en el sistema

Bases de Datos

No.	Nombre Base de Datos	Propietario	Codificación	Tamaño	Tablas	Eliminar
1	bdpsqladmin	upsqladmin	win1252	4648 kB	ver	eliminar
2	tecnodb	jazmin	utf8	4424 kB	ver	eliminar
3	tecnopolis	soporte	utf8	4528 kB	ver	eliminar
Total 3 Bases de Datos						

Figura 51. Crear y Consultar base de datos en la herramienta PostgreSQL Admin para usuarios con perfil 1 (Superusuario).

Al dar clic en la opción eliminar Base de datos El sistema validará mandando un mensaje de advertencia como se muestra a continuación (fig. 52):

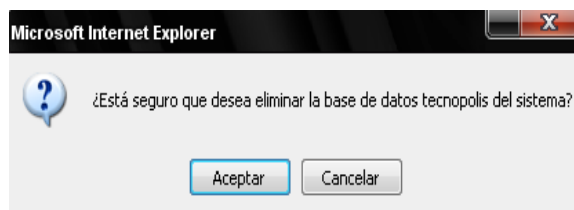


Figura 52. Mensaje de validación, Eliminar Base de datos

Si se da clic en Aceptar mostrará el siguiente mensaje “*Se eliminó la Base de datos*”, como se muestra en la siguiente figura (fig. 53)

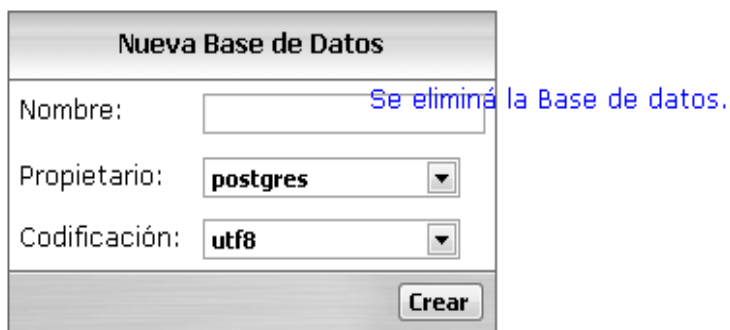


Figura 53. Eliminar Base de datos

Al haber seleccionado la liga ver debajo del título tablas, ésta abre otra página la cuál muestra las tablas de usuario con que cuenta esa base de datos, con la opción de insertar campos, ver sus propiedades o esquema de la tabla así como la opción de eliminar dicha tabla (figs. 54, 55).

Tablas de Usuario

Nueva Tabla

Nombre:

Campos:

Propietario: opertecnogens ▼

Tablas de usuario en tecnopolis

No.	Nombre Tabla	Propietario	Tamaño	Insertar	Propiedades	Eliminar
1	alumno	upsqldadmin	16 kB	insertar	ver	eliminar
2	beca	upsqldadmin	24 kB	insertar	ver	eliminar
3	area	upsqldadmin	24 kB	insertar	ver	eliminar
4	curso	upsqldadmin	16 kB	insertar	ver	eliminar
5	cuota	upsqldadmin	8192 bytes	insertar	ver	eliminar
Total 5 Tablas en tecnopolis						

Figura 54. Tablas de usuario en una base de datos en la herramienta PostgreSQL Admin

Tablas de Usuario

Indique los campos para la nueva tabla: progacademico

No.	Campo	Tipo de Dato	Longitud	Not Null	PK	FK
1	<input style="width: 100%;" type="text"/>	text	<input style="width: 50%;" type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	<input style="width: 100%;" type="text"/>	bool char date float4 int4 money numeric text time timestamp varchar	<input style="width: 50%;" type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="button" value="Crear"/>						

Copyright ©2008 TecnoGens.com

[Bases de Datos](#)
 [Usuarios](#)
 [Gupos](#)
 [SQL](#)
 [Acerca de...](#)

Figura 55. Características de los campos para una nueva tabla

Si la tabla no contiene ningún valor aparecerá un mensaje la tabla esta vacía (fig. 56) de lo contrario aparecerán los datos con que cuenta actualmente la tabla (fig. 57).

Tablas de Usuario

Base de Datos: tecnodb

Tabla: area

Proporcione los valores a insertar para los campos.

Insertar Registro		
txtclavearea	<input type="text"/>	(character varying)
txtnombrearea	<input type="text"/>	(character varying)
posicion	<input type="text"/>	(integer)
vigencia	<input type="text"/>	(character varying)
boolvigente	<input type="text"/>	(boolean) true/false
<input type="button" value="Insertar"/>		

La tabla está vacía.



Figura 56. Insertar registros en una tabla vacía.

Base de Datos: bdpsqladmin

Tabla: perfil

Proporcione los valores a insertar para los campos.

Insertar Registro		
idperfil	<input type="text"/>	(numeric)
txtnombreperfil	<input type="text"/>	(character varying)
txtdescripcionperfil	<input type="text"/>	(character varying)
<input type="button" value="Insertar"/>		

Registros que contiene actualmente la tabla.

idperfil	txtnombreperfil	txtdescripcionperfil
1	Administrador	Todos los privilegios concedidos al usuario de la interfaz gráfica psql Admin
2	Operador	los privilegios de consulta y manipulacion de algunos objetos de la interfaz gráfica tsqldadmin
3	Usuario	los privilegios de consulta de la interfaz gráfica tsqldadmin
Total: 3 Registros		

[Modificar Campos](#)



Figura 57. Insertar registros y consulta de los datos existentes en esa tabla.

Si requerimos cambiar algún campo de nuestra tabla damos clic a la liga Modificar campos y ésta nos enviara a la siguiente ventana con las opciones de actualizar y borrar (fig. 58).

Tablas de Usuario

Base de Datos: tecnopolisdb
Tabla: programa

Registros que contiene actualmente la tabla.

No. Campo	Campo	Actualizar	Nombre	Curso
1	ACA	Académico	Actualizar	eliminar
2	INF	Infantil	Actualizar	eliminar

Total: 2 Registros

Figura 58 Registros de la tabla

Si se requiere actualizar la información del lado del usuario específicamente en un campo se da clic en la liga actualizar y le enviará a la siguiente página (fig. 59):

Tablas de Usuario

Base de Datos: tecnopolisdb
Tabla: programa

claveprograma
ACA
nombreprograma
Académico
clavesubprograma
A
siglas
Ac
representante
López
cargorepresentante
Jefe
slogan
Ninguno

Actualizar



Figura 59. Actualización de catálogos de usuario, tabla programa

Al dar clic en propiedades se puede ver el esquema de la tabla y el nombre de la base de datos a la que pertenece, así como sus constraints que es una regla que se utiliza para mantener la integridad de los datos (fig. 60):

Tablas de Usuario

Base de Datos: bdpsqladmin
Tabla: usuariosweb

No.	Nombre de Columna	Nulo	Tipo de Dato	Precision	longitud
1	idusuario	no	integer	32	
2	txtlogin	no	character varying		20
3	txtpassword	no	character varying		20
4	txtnombreusuario	no	character varying		70
5	txtapellidopaternalusuario	no	character varying		70
6	txtapellidomaternousuario	si	character varying		70
7	txtcorreoelectronico	no	character varying		50
8	idperfil	no	numeric	2	
Total 8 Campos					

Constraints:

pk_usuariosweb, PRIMARY KEY(idusuario)
fk_usuariosweb_perfil, FOREIGN KEY(idperfil)

Figura 60. Esquema de la tabla usuariosweb en la herramienta TecnoGens PostgreSQL Admin

Al eliminar alguna tabla con solo dar clic en eliminar mostrara una ventana emergente con el siguiente mensaje (fig. 61):

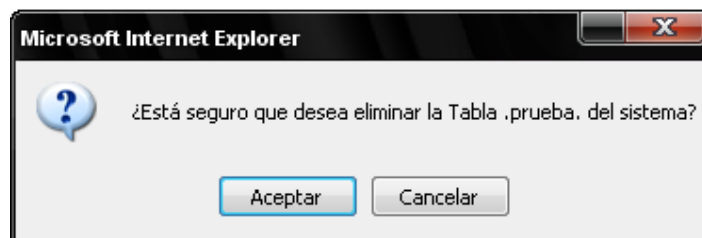


Figura 61. El mensaje con el nombre de la tabla que desea eliminar

III.3 ADMINISTRACIÓN DE USUARIOS

Una de las más básicas tareas de un administrador de base de datos es identificar los usuarios. Cada usuario que conecta en la base de datos debe de tener una cuenta.

Cuando un usuario conecta con una base de datos verifica que este usuario y la contraseña introducida almacenada en la base de datos, sea correcta. Las contraseñas se guardan en la base de datos.

En PostgreSQL Admin el administrador de base de datos debe asignar una cuenta para entrar al sistema posteriormente ese usuario puede crear mas usuarios, si dicho usuario cuenta con el perfil de administrador (1 o 4), entonces podrá crear usuarios asignándole un nombre de usuario y contraseña, asociarlo a un grupo de trabajo, así como teniendo la posibilidad de crear bases de datos (BD) y que permita la creación de otros usuarios, además de permitir consultar a los usuarios que están en el servidor, saber si están asociados a un grupo, así como asignar permisos para modificar las propiedades de password autenticándose y con la posibilidad de eliminar a ese usuario del sistema como se muestra en la (fig. 62).

Administración de Usuarios

No.	Nombre Usuario	Grupo(s)	Crea BD	Crea Usuarios	Privilegios	Modificar	Eliminar
1	postgres		Si	Si	privilegios	modificar	eliminar
2	upsqladmin	gpsqladmin	No	Si	privilegios	modificar	eliminar
Total 2 Usuarios							

Figura 62. Administración de usuarios

A un usuario se le puede otorgar una serie de privilegios. Un privilegio permite a un usuario acceder a ciertos objetos o realizar ciertas acciones:

- Privilegios sobre Objetos (Object privileges) a permisos sobre vistas, tablas, procedimientos.
- Privilegios del Sistema (System privileges) a permisos sobre “niveles de la base de datos” como pueden ser conexión a la base de datos, creación de usuarios, limitar cuentas.
- Privilegios sobre Roles (Role privileges) a muchos permisos son otorgados mediante roles agrupando un conjunto de privilegios.

Para otorgar privilegios se utilizo la sentencia **GRANT**, para quitar un privilegio o permiso a un usuario la sentencia **REVOKE**

Se pueden ver los privilegios (select, insert, update, rule, trigger) con que cuenta el usuario en dicha base de datos y el tipo de objeto (tablas, BD, vistas, procedimientos, etc.) al que se ejecuta dicho privilegio (fig.62). A si mismo tiene la posibilidad de modificar esos privilegios (figs.63, 64).

Privilegios en:

Usuario:

Base de Datos:

Tipo de Objeto:

El usuario jazmin tiene privilegios sobre los siguientes objetos.

tabla	select	insert	delete	update	references	rule	trigger	propietario	modificar
material	•	•		•				•	modificar
revisor	•	•	•	•	•	•	•	•	modificar
revisortipo	•	•	•	•	•	•	•	•	modificar
tipo	•	•	•	•	•	•	•	•	modificar
Total:	4 Registros								

Figura 63. Privilegios de un usuario en particular en una base de datos y un tipo de objeto

Administración de Usuarios

Modificar Privilegios		
Select:	<input checked="" type="radio"/> Si	<input type="radio"/> No
Insert:	<input checked="" type="radio"/> Si	<input type="radio"/> No
Delete:	<input checked="" type="radio"/> Si	<input type="radio"/> No
Update:	<input checked="" type="radio"/> Si	<input type="radio"/> No
References:	<input checked="" type="radio"/> Si	<input type="radio"/> No
Rule:	<input checked="" type="radio"/> Si	<input type="radio"/> No
Trigger:	<input checked="" type="radio"/> Si	<input type="radio"/> No
Modificar		

Modificar los privilegios del usuario upsqladmin en bdpsqladmin.usuariosweb

Figura 64. Modificar privilegios

De lo contrario si un usuario no tuviera privilegios sobre algún objeto del sistema entonces mandaría un mensaje como el siguiente (figura 65):

Administración de Usuarios

Privilegios en:	
Usuario:	<input type="text" value="userprueba"/>
Base de Datos:	<input type="text" value="pruebadb"/>
Tipo de Objeto:	<input type="text" value="Tablas"/>
ver	

El usuario userprueba no tiene privilegios sobre objetos de este tipo.

Figura 65. Mensaje sobre privilegios de objetos en el sistema

Para el cambio de password de un usuario solo podrá tener acceso a esta función el perfil de SuperUsuario y podrá cambiarlo por fines de seguridad (fig. 66).

Administración de Usuarios

Modificar Usuario		
Nombre:	postgres	Modificar las propiedades del login.
Contraseña:	●●●●●●	
Repetir Contraseña:	●●●●●●	
Nuevo Login:	●●●●●●●●	
Modificar		

```
ALTER USER postgres WITH PASSWORD postgres2
```

Figura 66. Modificar propiedades de login

Además podrán cambiar su contraseña los integrantes de la interfaz los cuales solo podrán cambiar el password de su propia cuenta sin derecho a cambiar el de otro usuario.

III.4 ADMINISTRACIÓN DE GRUPOS

Los grupos sirven para simplificar la asignación de privilegios. Los privilegios ordinarios deben ser asignados a un único usuario, uno cada vez. Esto puede resultar tedioso si los usuarios a los que hay que asignar los mismos accesos a una gran variedad de objetos de base de datos.

Los grupos son creados para evitar este problema. Un grupo requiere un nombre, y puede ser creado vacío (sin usuarios). Una vez creado, los usuarios que se pretende compartan permisos comunes son añadidos todos al grupo, y quedan asociados al grupo por su número de miembro. Los permisos en los objetos de base de datos son entonces asignados al grupo, en vez de a cada uno de los miembros

del grupo. Para un sistema con muchos usuarios y bases de datos, los grupos hacen que la asignación de permisos sea más cómoda para el administrador.

Dentro de la herramienta TecnoGens PostgresSQL admin., la administración de grupos se basa en los permisos para crear BD y usuarios, esta herramienta permite crear grupos así como ver a que usuarios tienen asignado ese grupo y con la opción de modificar sus características (fig.67), así como la posibilidad de eliminar el grupo (fig. 67).

Nuevo Grupo

Nombre:

Crear BD: Si No

Crear Usuarios: Si No

No.	Nombre Grupo	Usuarios(s)	Crea BD	Crea Usuarios	Modificar	Eliminar
1	gpsqladmin	upsqldadmin	No	Si	modificar	eliminar
Total 1 Usuarios						

Figura 67. Creación de nuevos grupos y características

Administración de Grupos

Modificar Grupo	
Nombre:	gpsqladmin
Crear BD:	<input checked="" type="radio"/> Si <input type="radio"/> No
Crear Usuarios:	<input checked="" type="radio"/> Si <input type="radio"/> No
 <input type="button" value="Modificar"/>	

Aregar Usuarios al Grupo	
Usuario:	<input type="text" value="opertecnogens"/>
<input type="button" value="Agregar"/>	

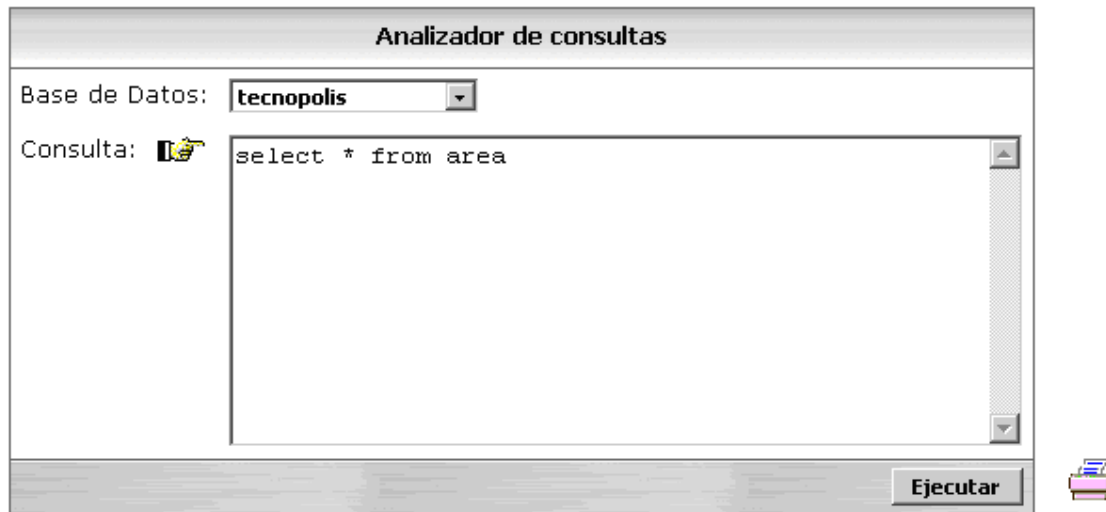
Usuarios que pertenecen a gpsqladmin

No.	Nombre Usuario	Eliminar del grupo
1	jazmin	eliminar
2	soporte	eliminar
3	upsqladmin	eliminar
Total 3 Usuarios en el Grupo		

Figura 68 Administración de Grupos

PostgreSQL Admin cuenta con una consola de consulta SQL que es muy accesible la cual está insertada en el editor, soporta sentencias DML y DDL y proporciona información sobre la base de datos.(Fig.69).

Consola de consultas SQL



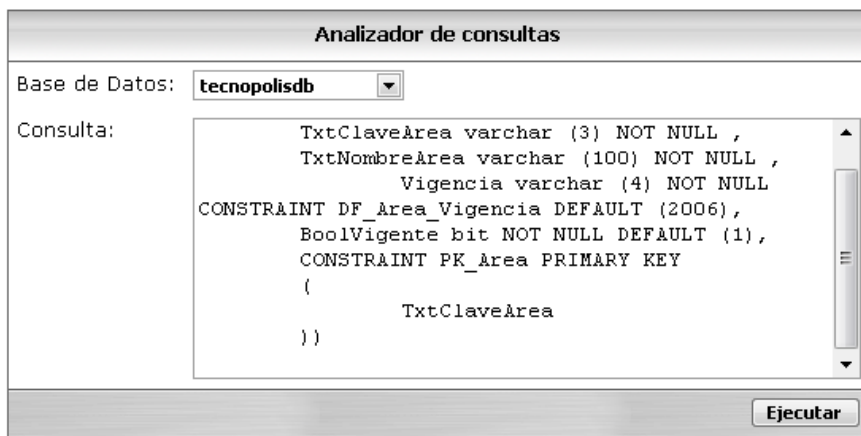
Se ejecutó la consulta.

txtclavearea	txtnombrearea	posicion	vigencia	boolvigente
BD	Bases de datos	1	2008	1
LP	Lenguajes de programación	2	2008	1
SI	Sistemas de información	3	2008	1
SO	Sistemas operativos	4	2008	1
SC	Seguridad en cómputo	5	2008	1
TLL	Talleres de prácticas	6	2008	1
CN	Cómputo para niños	7	2008	1
Total:	7 Registros			

Figura 69. Consola de consulta SQL


En cuanto al manejo de errores los toma del manejador el cuál esta en idioma inglés, lo que se esperaba es realizar la traducción de errores (básicos) pero por tiempo no se completó. Pero al dar una sentencia mal escrita o mal estructurada el sistema mandara un mensaje de error y no ejecutará la sentencia escrita por el usuario (Fig. 70).


Consola de consultas SQL



ERROR: column "boolvigente" is of type bit but default expression is of type integer HINT: You will need to rewrite or cast the expression.

Figura 70. Manejo de errores para la consola SQL

Para exportar datos se puede hacer de la siguiente forma dar clic en la imagen  esta opción permite la impresión de información en este caso del analizador de consulta, dicha información se verá como lo muestra la siguiente figura 71.

Otra manera de exportar información es la que se encuentre en la imagen  como es el caso de la figura 59, en donde nos permite exportar información relacionado a las tablas de usuario a un archivo de Excel.



Consola de consultas SQL

Analizador de consultas

Base de Datos:

Consulta:

```
CREATE TABLE Curso (  
    ClaveCurso varchar(255) NOT NULL,  
    ClaveCuota float NULL,  
    ClaveArea varchar(255) NULL,  
    Curso varchar(255) NULL,  
    ClasificacionCurso varchar(255) NULL,  
    Horas float NULL,  
    Semanas float NULL,  
    Constancia varchar(255) NULL,  
    Vigencia varchar(255) NULL,
```

Figura 71. Exportar datos de una inserción de tabla

CONCLUSIONES

En términos generales se concluye que el sistema cumple con los objetivos establecidos en un principio, es un sistema que nos sirve como apoyo para el proceso de administración de las bases de datos instaladas en el manejador de postgresql, este sistema ayudó a la empresa TecnoGens a mostrar su información de manera más resumida y con una interfaz vistosa que les permite ver sus usuarios y así asignar privilegios sobre los objetos de las bases de datos permite la administración de grupos y además permite ejecutar sentencias SQL por medio de dicha interfaz, dentro de las ventajas que podemos considerar es que la interfaz es fácil de manejar y tienen un ambiente agradable en cuanto al diseño y esta adecuada para la administración de catálogos como son cuotas, profesores, cursos, materiales, entre otros.

Si hablamos de seguridad podría concluir que la interfaz es segura ya que no solo se deben tener cuenta para el servidor sino también para la interfaz y el único que puede cambiar esa cuenta de servidor es el administrador eso es de gran importancia ya que en realidad el administrador sigue llevando el control total pero dejando un poco mas de privilegios cuando el este ausente.

Este sistema está limitado en cuanto a que no muestra un diagrama de relación entre tablas aunque si se puede asignar una llave primaria y/o foránea.

En este momento solo se puede realizar el respaldo de tu información exportando los datos que se introdujeron al crear un objeto o al generar una consulta por lo que el administrador lo debe hacer de forma manual, pero este sistema no es para que se quede a este nivel, el sistema contiene la documentación necesaria para irse ampliando y adecuando cada vez más a las necesidades de la empresa.

Esta herramienta fue de gran importancia para la empresa es óptima, segura y reduce favorablemente los procesos administrativos, es de apoyo para administradores de las bases de datos, administradores de catálogos y los directivos.

Esta herramienta asegura el cumplimiento de estándares obligatorios en su organización, ayuda a implementar proyectos de mejora continua, Optimiza la utilización de los recursos por ejemplo de materiales para los cursos de actualización, reduce los tiempos en los ciclos de los procesos.

BIBLIOGRAFÍA

1. C. Batini, S. Ceri, S.B. Navathe (1994)
Diseño Conceptual de Bases de Datos. Un enfoque de entidades-interrelaciones
Addison-Wesley / Díaz de Santos
2. T. Connolly, C. Begg, A. Strachan (1996)
Database Systems. A Practical Approach to Design, Implementation and Management
Addison-Wesley
Segunda Edición en 1998.
3. C.J. Date (1993)
Introducción a los Sistemas de Bases de Datos
Volumen I, Quinta Edición
Addison-Wesley Iberoamericana
Sexta Edición en 1995 (en inglés, por Addison-Wesley)
4. El lenguaje unificado de modelado UML
Grady Booch, James Rumbaugh, Ivar Jacobson
Pearson Addison Wesley
Segunda edición
5. Aplique SQL
Groff, James R. y Weinberg Paul N. Osborne
McGraw-Hill, 1991
6. Organización de las Bases de Datos
Martin, James.
Prentice-Hall Hispanoamericana, S. A., 1991

7. R. Elmasri, S.B. Navathe (1997)
Sistemas de Bases de Datos. Conceptos fundamentales
Segunda Edición
Addison-Wesley Iberoamericana

8. Apuntes de Diplomado de Administración de bases de datos novena generación
Centro de extensión en cómputo COAPA, UNAM

9. Material de Cursos en línea Análisis y diseño Orientado a Objetos con UML
Dirección General de Servicios de Cómputo Académico, UNAM

REFERENCIA ELECTRÓNICA

- Tutorial de PostgreSQL
El equipo de desarrollo de PostgreSQL
<http://es.tldp.org/Postgresql-es/web/navegable/tutorial/tutorial.html>
Es marca registrada 1996-9 por el Postgres Global Development Group.

- <http://www.postgresql.org/download/linux>
Copyright © 1996 – 2009 PostgreSQL Global Development Group

- <http://www.postgresql.org/>
Copyright © 1996 – 2009 PostgreSQL Global Development Group

TECNOGENS

ANEXOS



Guía de desarrollo de Casos de Uso

CASO DE USO	[Nombre del Caso de Uso y en su caso el módulo al que pertenece, debe tener consistencia con el nombre del Caso en el diagrama de Casos de Uso]				
Id. del CU	[No. del caso de uso] ¹	Prioridad	Alta / Media / Baja [Subrayar el correspondiente] ²	Estado	En elaboración/Propuesto/Validado [Subrayar el correspondiente] ³
Actores participantes	[Enunciar el actor(es) (cualquier cosa que interactúe con el sistema) que intervienen directamente en el Caso de Uso, se le deberá identificar claramente y redactar un breve descripción sobre el papel que desempeña el mismo.]				
Breve Descripción	[Describir concisa y brevemente el conjunto de escenarios que estructuran el Caso de Uso]				
Pre-condiciones	[Enunciar los requisitos necesarios, tanto del proceso como del sistema para que el Caso de Uso se lleve a cabo, es decir los requisitos para que este caso de uso se ponga en marcha] Del proceso ■ ■ Del sistema ■ ■				

¹ Este ID corresponde al que aparece en el diagrama general de Casos de Uso

² Indica la importancia del Caso de Uso con respecto a todos los demás que comprende el sistema. Se determina por las características del sistema y por las prácticas con el cliente y usuario

³ Indica si se está desarrollando, si ya está terminada la propuesta del Caso, o si ya ha sido validado por los usuarios.

Flujo Principal	[Escribir la secuencia(s) de acción(es) que se dan normalmente en el caso de uso de manera general]
Flujos Alternos	[Indicar las secuencias de acciones que detallan las alternativas del Caso, es decir, los que se pueden desprender de los flujos principales, mencionándolos con el mayor detalle todos los elementos que integran cada uno de estos flujos alternos, recordando que estos flujos serán interpretados por otras personas y se requiere que se expliquen de la manera más específica, detallada y sin ambigüedades.]
Flujos de excepción	[Señalar todos los eventos extraordinarios que pudieran surgir del Caso de Uso]
Post-condiciones	[Son los requisitos que deben cumplirse posteriormente, una vez terminado el Caso de Uso]
Requerimientos No Funcionales	[Las restricciones impuestas al software tienen que ver con características que de una u otra forma puedan limitar el sistema, como por ejemplo, el rendimiento (en tiempo y espacio), interfaces de usuario, fiabilidad (robustez del sistema, disponibilidad de equipo), mantenimiento, seguridad, portabilidad, estándares, etc.]
Notas	[Algún comentario que complemente la información del Caso de Uso y que no haya sido mencionado con anterioridad]

Instalación CentOS

- Arranque de CD-ROM : Primeramente debe asegurarse que su BIOS se encuentre configurado para leer el CD-ROM al momento de inicializarse su PC o Servidor, esto es necesario para que se ejecute el "shell" de instalación CentOS en lugar del sistema operativo que tiene instalado en su disco duro.

NOTA: El acceso al BIOS varía dependiendo del Hardware que utilice, sin embargo, las opciones más comunes son las teclas ESC, F2 o DEL.

- Arranque de CentOS : Inicialice su sistema con el Disco 1 de CentOS colocado en su lector de CD-ROM. Si configuró correctamente su BIOS para leer CD-ROM's al arranque.
- Proceso inicial y detección de Hardware: Estando en la pantalla anterior, simplemente esperando u oprimiendo Enter, iniciará el proceso para detección de Hardware y proceso de instalación a través de una consola gráfica , esta secuencia puede durar entre 10 o 15 segundos, mientras observa mensajes descriptivos sobre su sistema. Si desea realizar su proceso de instalación mediante linea de comandos, entonces deberá introducir la secuencia linux text.
- Verificación de medios (CD-ROM's): Posteriormente será presentado con la opción de realizar una prueba de integridad sobre los CD-ROM's de instalación CentOS, esta prueba dura entre 10 y 15 minutos para los 4 CD's de instalación. Si no desea realizar esta prueba, seleccione la opción Skip.
- Bienvenida: Es desplegado un mensaje de bienvenida de CentOS con instrucciones de navegación, simplemente oprima Next para continuar con el proceso.
- Lenguaje del Sistema: La siguiente selección que debe hacer es sobre el idioma que desea para su proceso de instalación y sistema, puede elegir castellano (Spanish):
- Configuración de Teclado: Seleccione el tipo de teclado en base a su sistema; español si posee un teclado con tecla eñe y caracteres latinos, o bien, estadounidense de lo contrario.
- Tipo de Instalación (Actualización, Escritorio personal, Estación de Trabajo, Servidor, Personalizada): Seguido, será realizado un proceso de auto-detección para determinar si posee una instalación pre-existente de CentOS, de encontrarse será presentado con una

opción de actualización, de lo contrario, podrá elegir entre 4 modalidades -- Escritorio Personal, Estación de Trabajo, Servidor o Personalizada -- cada una de éstas presenta una breve descripción de su funcionamiento. Para el caso de esta guía, se asumirá una configuración de Servidor, ya que CentOS se caracteriza por su estabilidad en esta área.

Partición de Disco Duro: Posteriormente, debe realizar el particionamiento de su disco duro, CentOS le ofrece dos alternativas para llevar a cabo este proceso. (NOTA: Es recomendable lea el proceso de Pre-Instalación en caso que no esté familiarizado con el concepto de particionar discos duros).

- Particionamiento Automático: Como su nombre lo implica, CentOS realiza el particionamiento del disco duro a dimensiones pre-determinadas, sin embargo, esto implica generalmente que debe borrar toda la información existente en su disco duro.
- Partición manual con Disk Druid: Para usuarios con amplio conocimiento del proceso de partición, pueden optar por hacer su propia distribución de espacio con esta opción.

Se recomienda seleccionar *Particionamiento Automático* puesto que implica el método más directo y sencillo para instalar CentOS, no obstante, si utiliza la opción de *Disk Druid* tome en cuenta los factores de redimensionamiento que utilice para sus particiones. La principal ventaja de utilizar la opción automática, se debe a que las particiones son creadas en el orden y tamaño recomendado por CentOS para operación óptima.

- Administrador de Arranque ("Boot Loader") : Posteriormente debe confirmar la instalación del administrador de arranque ("Boot Loader") GRUB; si CentOS será el único sistema operativo instalado en su equipo, este paso no debe ser de mayor trascendencia. Sin embargo, si posee más de un disco duro, o bien, además de CentOS existirá otro sistema operativo, esta configuración tiene implicaciones en la manera que es inicializado su sistema.
- Configuración de Red: Seguido debe indicar los parámetros para acceso a red, ya sea manualmente con información de nodos IP y DNS, o bien, indicando una configuración automática vía DHCP.
- Configuración Cortafuegos ("Firewall"): Aquí debe especificar si desea instalar un mecanismo de "Firewall" para proteger su sistema. De ser así, también tiene la opción de

habilitar determinados servicios para que éstos no sean afectados por el "Firewall", tales como: SSH, Servidores Web, Servidores de Correo y FTP. Idiomas adicionales para Sistema: Esta consola le permitirá definir idiomas alternos para su sistema, además del seleccionado en primeras instancias.

- Zona Horaria del Sistema: Aquí debe definir la zona horaria para su instalación. Definición de usuario raíz (root): Posteriormente debe indicar una contraseña para el usuario raíz ("root") del sistema, como su nombre lo indica, éste será el usuario maestro de la instalación y tendrá control absoluto de acceso sobre toda aplicación en CentOS.
- Selección de Aplicaciones/ Paquetes : En esta consola tiene la opción de elegir la serie de aplicaciones que serán instalados, por "default" se encontraran seleccionados una serie de paquetes que corresponden a una configuración típica de servidor, esto debido a que con anterioridad fue seleccionado este tipo de instalación. No obstante, puede agregar aplicaciones a su discreción, tales como un ambiente gráfico u otra función necesaria para cumplir con sus requerimientos.
- Selección de Aplicaciones/ Paquetes: Seleccionadas las aplicaciones, al oprimir el botón Siguiente iniciará la instalación de aplicaciones, dependiendo de su Hardware, este paso puede demorar entre 20 o 40 minutos. Asegúrese también de tener cerca de su PC los discos de instalación y no dejarla desatendida, ya que durante este proceso necesitará colocar los distintos CD-ROM's conforme los requiera su instalación.
- Ultimo paso : Ha terminado de instalar satisfactoriamente CentOS Linux. Ahora sólo debe reinicializar su sistema para entrar a su ambiente Linux.